



The Deep Inferential Aspects of the CoRE Calculus

Serge Autexier

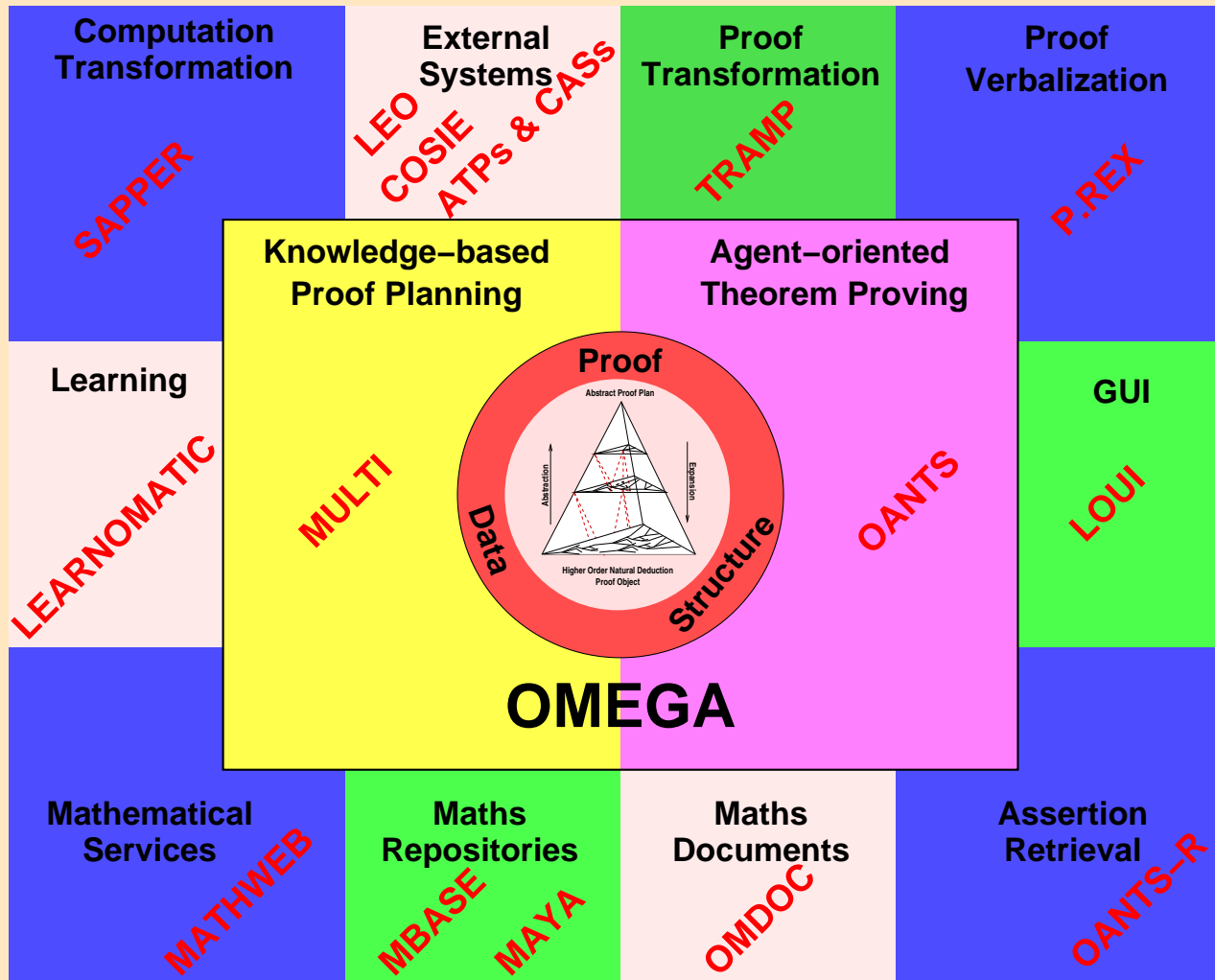
`serge@ags.uni-sb.de`

DFKI GmbH & CS Department, Saarland University, Saarbrücken, Germany

ICCL-Autumn Workshop, December 14-15 2005

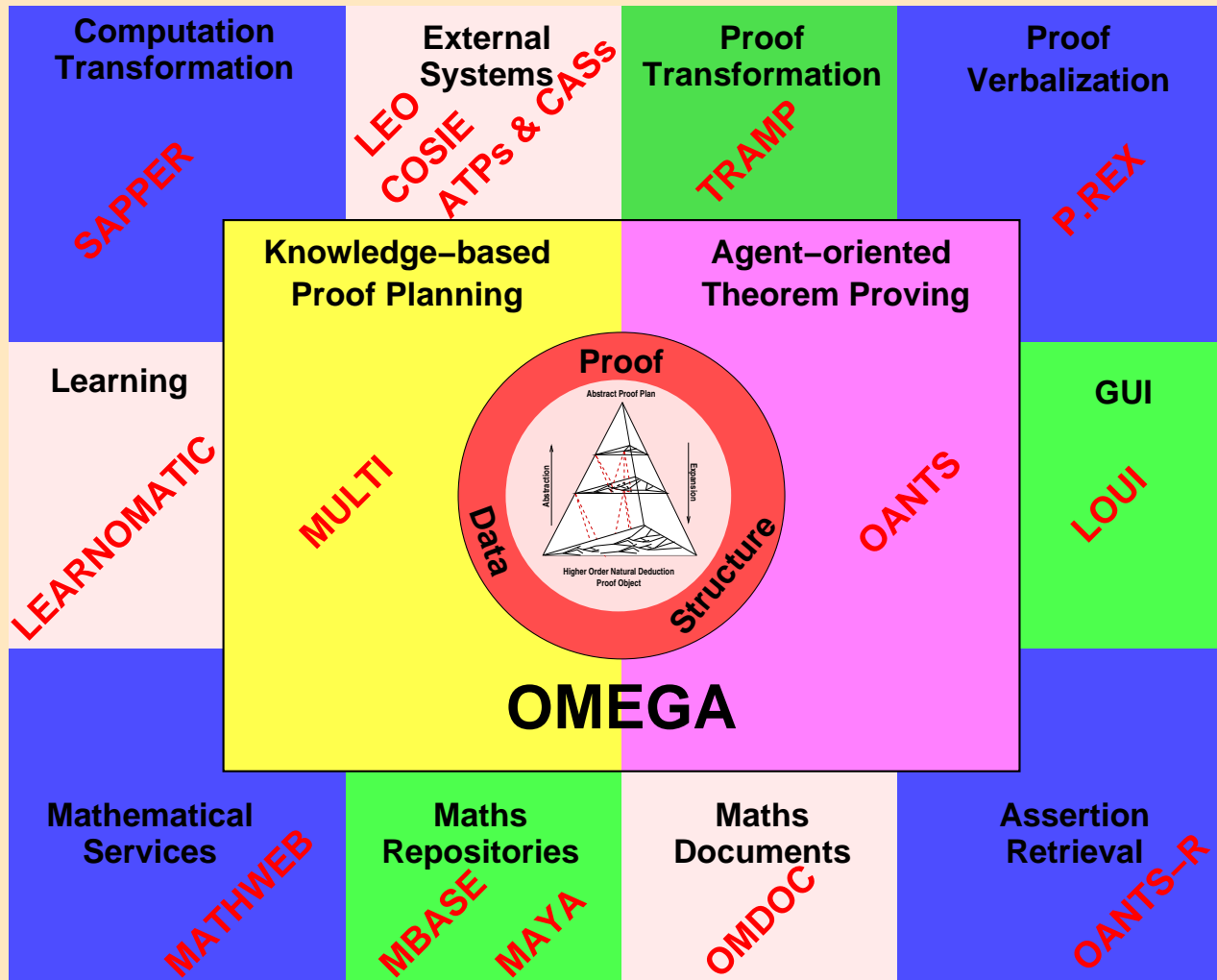
Dresden, Germany

Major Aspects of the Old Ω MEGA



- Aims at being a proof assistant for mathematics
- Proof planning, Agent-based proof search
- External theorem provers and CASs
- Database for mathematical theories
- Proof explanation in natural language

Major Aspects of the Old Ω MEGA



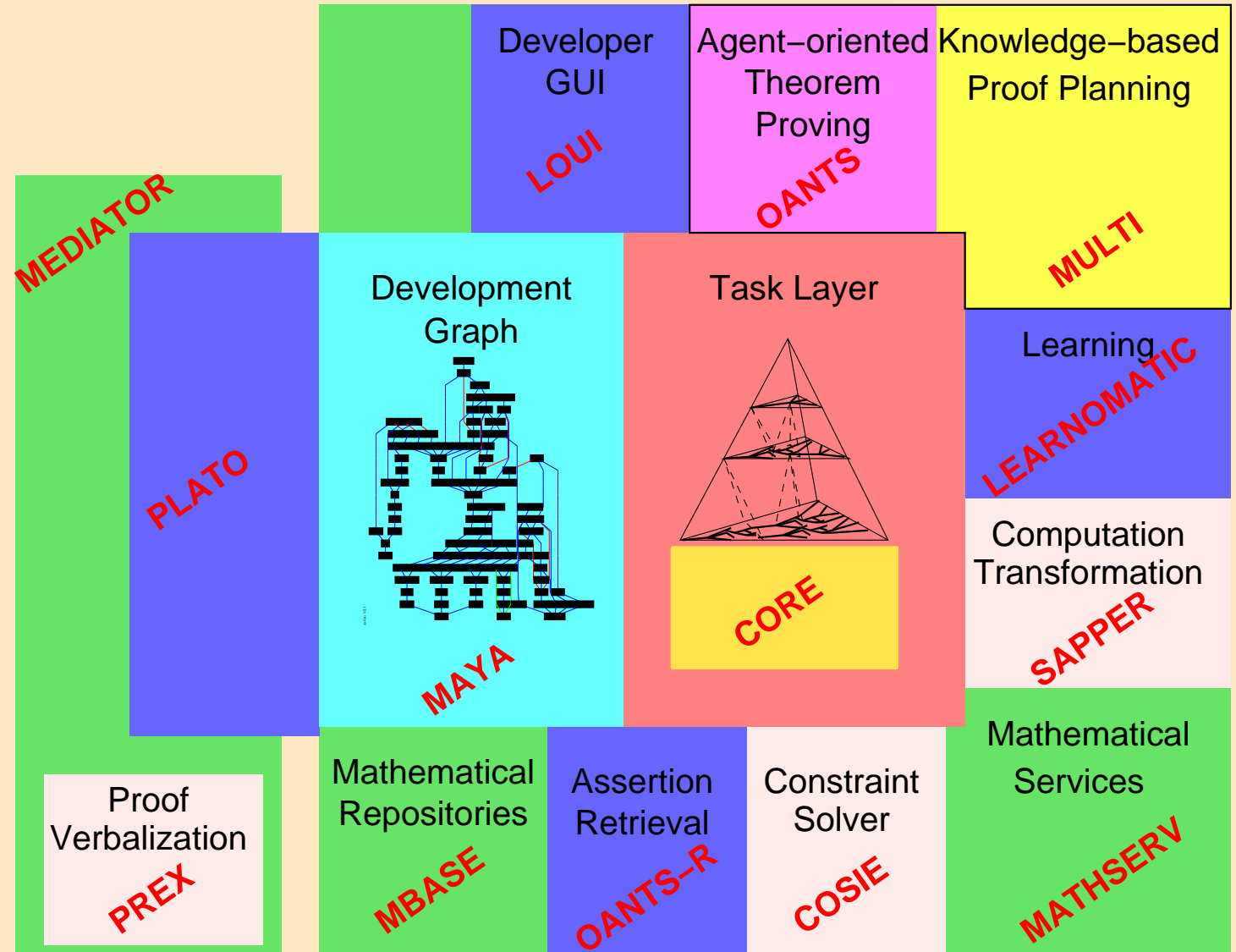
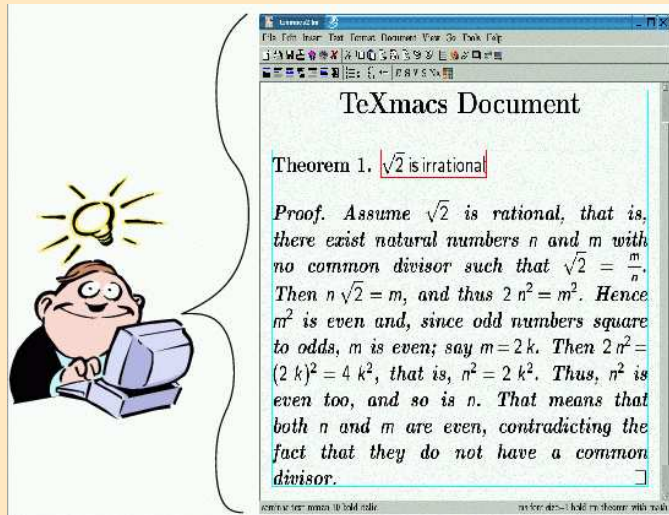
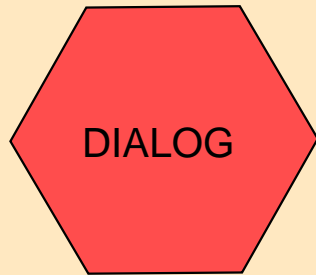
- Base Calculus: Natural Deduction
- Hierarchical proof datastructure (PDS) tailored to
 - ▶ base ND calculus
 - ▶ proof planning methods
 - ▶ tactics
 - ▶ Heavily overcrowded
- Graphical user interface *LOUI*

Why and What did we Change?



- Natural Deduction Calculus too cumbersome (not natural enough)
 - ▶ Want direct reasoning with the assertions (interactive & automatic)
 - ▶ Natural Deduction not flexible enough
- The proof datastructure was completely overcrowded
 - ▶ New generic proof datastructure
 - ▶ Definition of the task-layer as uniform reasoning platform
- GUI *L Ω UI* nice, but did not meet requirements of targeted users, say mathematicians
 - ▶ Plug Ω MEGA into tools anyway used by users, e.g. TEXMACS
 - ▶ Ω MEGA as reasoning service provider (analogy: grammar checkers)

New Architecture



Context



- Interactive proof assistant system Ω MEGA
- Human user assisted by ATPs (including PP)
- Communication between the user and the theorem proving system is crucial
(close to CML, “mathural” proof development style)
- ... Especially if human user is not expert in formal logics
- ... How do they do proofs?

A Textbook Proof

Theorem. For any natural numbers n it holds $\sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2$.

Proof.

The proof is by induction over n . For $n = 0$ the statements holds trivially by definitions. In the induction step by definition we have

$$(n + 1)^3 + \sum_{i=1}^n i^3 = ((n + 1) + \sum_{i=1}^n i)^2$$

$((n + 1) + \sum_{i=1}^n i)^2$ is equivalent $(n + 1)^2 + 2(n + 1) \sum_{i=1}^n i + (\sum_{i=1}^n i)^2$, and by induction hypothesis we obtain $(n + 1)^3 = (n + 1)^2 + 2(n + 1) \sum_{i=1}^n i$.

...

Sequent Calculus Proof



$$\begin{array}{c}
 \vdots \\
 \hline
 n = n' + 1 \wedge \sum_{i=1}^{n'} i^3 = (\sum_{i=1}^{n'} i)^2 \vdash \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2 \\
 \hline
 \vdash (n = n' + 1 \wedge \sum_{i=1}^{n'} i^3 = (\sum_{i=1}^{n'} i)^2) \\
 \Rightarrow \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2 \\
 \hline
 \vdash \forall n, n'. (n = n' + 1 \wedge \sum_{i=1}^{n'} i^3 = (\sum_{i=1}^{n'} i)^2) \\
 \Rightarrow \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2 \\
 \hline
 \vdash \forall n, n'. (n = n' + 1 \wedge \sum_{i=1}^{n'} i^3 = (\sum_{i=1}^{n'} i)^2) \\
 \Rightarrow \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2, \forall n. \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2 \\
 \hline
 (\forall n, n'. (n = n' + 1 \wedge \sum_{i=1}^{n'} i^3 = (\sum_{i=1}^{n'} i)^2)) \\
 \Rightarrow \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2 \Rightarrow \forall n. \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2 \\
 \vdash \forall n. \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2 \\
 \hline
 (\forall n. n = 0 \Rightarrow \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2) \\
 \Rightarrow ((\forall n, n'. (n = n' + 1 \wedge \sum_{i=1}^{n'} i^3 = (\sum_{i=1}^{n'} i)^2)) \\
 \Rightarrow \forall n. \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2) \vdash \forall n. \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2 \\
 \hline
 \forall \mathbf{P}. (\forall n. n = 0 \Rightarrow \mathbf{P}(n)) \\
 \Rightarrow ((\forall n, n'. (n = n' + 1 \wedge \mathbf{P}(n'))) \Rightarrow \forall n. \mathbf{P}(n)) \vdash \forall n. \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2
 \end{array}$$

$$\begin{array}{c}
 \vdots \\
 \hline
 n = 0 \vdash \sum_{i=1}^0 i^3 = (\sum_{i=1}^0 i)^2, \\
 \forall n. \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2 \\
 \hline
 n = 0 \vdash \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2, \\
 \forall n. \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2 \\
 \hline
 \vdash n = 0 \Rightarrow \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2, \\
 \forall n. \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2 \\
 \hline
 \vdash \forall n. n = 0 \Rightarrow \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2, \\
 \forall n. \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2 \\
 \hline
 \vdash \forall n. n = 0 \Rightarrow \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2, \\
 \forall n. \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2
 \end{array}$$

Analysis

- Human users perform proofs by applying definitions, lemmas, theorems (*assertions*), information contained in the formula to (sub-)formulas
- [Huang94] introduced the *assertion level* as basis for NL proof presentation
- **Example:** $\forall S_1, S_2 : Set. S_1 \subseteq S_2 \Leftrightarrow \forall x. x \in S_1 \Rightarrow x \in S_2.$

This assertion allows us to derive

1. $a \in S'_2$ from $a \in S'_1$ and $S'_1 \subseteq S'_2$
 2. $S'_1 \not\subseteq S'_2$ from $a \in S'_1$ and $a \notin S'_2$
 3. $\forall x. x \in S'_1 \Rightarrow x \in S'_2$ from $S'_1 \subseteq S'_2$
- **Drawbacks:**
 - ▶ Proof search is not at the assertion level
 - ▶ No active support what *rules* follow from an assertion

Goals & Requirements



- Have a calculus where assertion application is a primitive rule
- Requires:
 - ▶ Means to efficiently obtain possible rules from an assertion
 - ▶ Include information contained in subformulas
 - ▶ Means to apply these rules on subformulas
- Build a calculus around a primitive assertion application rule

Idea

To prove $\forall n. \sum_{i=1}^n i^3 = \left(\sum_{i=1}^n i\right)^2$

(0) we apply $\forall P. (\forall n. n = 0 \Rightarrow P(n)) \Rightarrow ((\forall n, n'. (n = n' + 1 \wedge P(n')) \Rightarrow P(n)) \Rightarrow \forall n. P(n))$



to obtain $\forall n. n = 0 \Rightarrow \sum_{i=1}^n i^3 = \left(\sum_{i=1}^n i\right)^2 \wedge$
 $\forall n, n'. n = n' + 1 \wedge \sum_{i=1}^{n'} i^3 = \left(\sum_{i=1}^{n'} i\right)^2 \Rightarrow \sum_{i=1}^n i^3 = \left(\sum_{i=1}^n i\right)^2$

(0) then apply $n = 0$

to obtain $\forall n. n = 0 \Rightarrow \sum_{i=1}^0 i^3 = \left(\sum_{i=1}^0 i\right)^2 \wedge$
 $\forall n, n'. n = n' + 1 \wedge \sum_{i=1}^{n'} i^3 = \left(\sum_{i=1}^{n'} i\right)^2 \Rightarrow \sum_{i=1}^n i^3 = \left(\sum_{i=1}^n i\right)^2$

Everything we need is in the formulas,
 but how to make that information explicit?



Sequent Calculus Proof

Let $Prop(n) := \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2$

$\frac{}{\vdash \forall n, n'. (n = n' + 1 \wedge Prop(n'))} \text{Ax}$ $\frac{}{\vdash \forall n. Prop(n)} \text{Ax}$ $\frac{}{\Rightarrow \forall n. Prop(n), \forall n. Prop(n)}$	$\frac{n = 0 \vdash Prop(0), \forall n. Prop(n)}{n = 0 \vdash Prop(n), \forall n. Prop(n)} \text{Rew}$ $\frac{}{\vdash n = 0 \Rightarrow Prop(n), \forall n. Prop(n)} \Rightarrow_R \alpha$ $\frac{}{\vdash \forall n. n = 0 \Rightarrow Prop(n), \forall n. Prop(n)} \forall_R \delta$
$\frac{(\forall n, n'. (n = n' + 1 \wedge Prop(n')) \Rightarrow \forall n. Prop(n))}{\vdash \forall n. Prop(n)}$	$\frac{}{\vdash \forall n. n = 0 \Rightarrow Prop(n), \forall n. Prop(n)} \Rightarrow_L \beta$
$\frac{(\forall n. n = 0 \Rightarrow Prop(n)) \Rightarrow (\forall n, n'. (n = n' + 1 \wedge Prop(n')) \Rightarrow \forall n. Prop(n))}{\vdash \forall n. Prop(n)} \text{Ax}$	
$\frac{\forall P. (\forall n. n = 0 \Rightarrow P(n)) \Rightarrow (\forall n, n'. (n = n' + 1 \wedge P(n')) \Rightarrow \forall n. P(n))}{\vdash \forall n. Prop(n)} \forall_L P \leftarrow Prop \gamma$	



Sequent Calculus Proof

Let $Prop(n) := \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2$

$\frac{}{\vdash \forall n, n'. (n = n' + 1 \wedge Prop(n'))} \text{Ax}$ $\frac{}{\vdash \forall n. Prop(n)} \text{Ax}$ $\frac{}{\Rightarrow \forall n. Prop(n), \forall n. Prop(n)} \text{Ax}$ <hr/> $\frac{}{\vdash \forall n, n'. (n = n' + 1 \wedge Prop(n'))} \text{Ax}$ $\frac{}{\Rightarrow \forall n. Prop(n)} \text{Ax}$ $\frac{}{\vdash \forall n. Prop(n)} \text{Ax}$ <hr/> $\frac{}{\vdash \forall n. n = 0 \Rightarrow Prop(n), \forall n. Prop(n)} \text{Ax}$ <hr/> $\frac{}{\forall P. (\forall n. n = 0 \Rightarrow P(n))} \text{Ax}$ $\Rightarrow ((\forall n, n'. (n = n' + 1 \wedge P(n'))) \Rightarrow \forall n. P(n)) \vdash \forall n. Prop(n)$	$\frac{n = 0 \vdash Prop(0), \forall n. Prop(n)}{n = 0 \vdash Prop(n), \forall n. Prop(n)} \text{Rew}$ $\frac{}{\vdash n = 0 \Rightarrow Prop(n), \forall n. Prop(n)} \Rightarrow_R \alpha$ $\frac{}{\vdash \forall n. n = 0 \Rightarrow Prop(n), \forall n. Prop(n)} \forall_R \delta$ <hr/> $\frac{}{\vdash \forall n. n = 0 \Rightarrow Prop(n), \forall n. Prop(n)} \Rightarrow_L \beta$ <hr/> $\frac{}{\forall_L P \leftarrow Prop \gamma}$
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------



Sequent Calculus Proof

Let $Prop(n) := \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2$

$\frac{}{\vdash \forall n, n'. (n = n' + 1 \wedge Prop(n'))} \text{Ax}$ $\frac{}{\vdash \forall n. Prop(n)} \text{Ax}$ $\frac{}{\Rightarrow \forall n. Prop(n), \forall n. Prop(n)} \text{Ax}$	$\frac{n = 0 \vdash Prop(0), \forall n. Prop(n)}{n = 0 \vdash Prop(n), \forall n. Prop(n)} \text{Rew}$ $\frac{}{\vdash n = 0 \Rightarrow Prop(n), \forall n. Prop(n)} \Rightarrow_R \alpha$ $\frac{}{\vdash \forall n. n = 0 \Rightarrow Prop(n), \forall n. Prop(n)} \forall_R \delta$
$\frac{}{(\forall n, n'. (n = n' + 1 \wedge Prop(n')))} \Rightarrow \forall n. Prop(n)$ $\frac{}{\vdash \forall n. Prop(n)}$	$\frac{}{\vdash \forall n. n = 0 \Rightarrow Prop(n), \forall n. Prop(n)} \Rightarrow_L \beta$
$\frac{}{(\forall n. n = 0 \Rightarrow Prop(n))} \Rightarrow (\forall n, n'. (n = n' + 1 \wedge Prop(n')))$ $\Rightarrow \forall n. Prop(n) \vdash \forall n. Prop(n)$	$\frac{}{\Rightarrow_L \beta}$
$\forall P. (\forall n. n = 0 \Rightarrow P(n))$ $\Rightarrow ((\forall n, n'. (n = n' + 1 \wedge P(n'))) \Rightarrow \forall n. P(n)) \vdash \forall n. Prop(n)$	$\frac{}{\forall_L P \leftarrow Prop \gamma}$

What Formulas can tell us...



- Formulas are not just a string of symbols. . .

What Formulas can tell us...

- Formulas are not just a string of symbols. . .
- Formulas are trees, subformulas occur negatively or positively

What Formulas can tell us...

- Formulas are not just a string of symbols. . .
- Formulas are trees, subformulas occur negatively or positively
- The occurring connectives have a close correspondence to structural properties of the SK proof

What Formulas can tell us...

- Formulas are not just a string of symbols. . .
- Formulas are trees, subformulas occur negatively or positively
- The occurring connectives have a close correspondence to structural properties of the SK proof
 - ▶ positive \wedge and negative \vee to splits in the SK proof
 \implies their subformulas occur in different sequents
different logical context

What Formulas can tell us...

- Formulas are not just a string of symbols. . .
- Formulas are trees, subformulas occur negatively or positively
- The occurring connectives have a close correspondence to structural properties of the SK proof
 - ▶ positive \wedge and negative \vee to splits in the SK proof
 \implies their subformulas occur in different sequents
different logical context
 - ▶ negative \wedge and positive \vee don't
 \implies their subformulas occur in the same sequent
same logical context

What Formulas can tell us...

- Formulas are not just a string of symbols. . .
- Formulas are trees, subformulas occur negatively or positively
- The occurring connectives have a close correspondence to structural properties of the SK proof
 - ▶ positive \wedge and negative \vee to splits in the SK proof
 \implies their subformulas occur in different sequents
different logical context
 - ▶ negative \wedge and positive \vee don't
 \implies their subformulas occur in the same sequent
same logical context
 - ▶ negative \exists quantifiers and positive \forall quantifiers have an *Eigenvariable* condition

What Formulas can tell us...

- Formulas are not just a string of symbols. . .
- Formulas are trees, subformulas occur negatively or positively
- The occurring connectives have a close correspondence to structural properties of the SK proof
 - ▶ positive \wedge and negative \vee to splits in the SK proof
 \implies their subformulas occur in different sequents
different logical context
 - ▶ negative \wedge and positive \vee don't
 \implies their subformulas occur in the same sequent
same logical context
 - ▶ negative \exists quantifiers and positive \forall quantifiers have an *Eigenvariable* condition
 - ▶ positive \exists quantifiers and negative \forall quantifiers can be freely instantiated

What tell us...

Logic of symbols...

Quantifiers occur negatively or positively

There is a close correspondence to structural

Polarities

Uniform Types

[Smullyan68, Wallen90]



- ▶ positive \wedge and negative \vee to splits in the SK proof
 \implies their subformulas occur in different sequents
different logical context
- ▶ negative \wedge and positive \vee don't
 \implies their subformulas occur in the same sequent
same logical context
- ▶ negative \exists quantifiers and positive \forall quantifiers have an *Eigenvariable* condition
- ▶ positive \exists quantifiers and negative \forall quantifiers can be freely instantiated

Polarities & Uniform Types

- Assign polarities to formulas

$$\underbrace{\varphi_1, \dots, \varphi_n}_- \vdash \underbrace{\psi_1, \dots, \psi_m}_+$$

Signed formulas φ^- , ψ^+

- Categorize signed formulas by uniform types

α	α_0	α_1
$(\varphi \vee \psi)^+$	φ^+	ψ^+
$(\varphi \Rightarrow \psi)^+$	φ^-	ψ^+
$(\varphi \wedge \psi)^-$	φ^-	ψ^-
$(\neg\varphi)^+$	φ^-	—
$(\neg\varphi)^-$	φ^+	—

β	β_0	β_1
$(\varphi \wedge \psi)^+$	φ^+	ψ^+
$(\varphi \vee \psi)^-$	φ^-	ψ^-
$(\varphi \Rightarrow \psi)^-$	φ^+	ψ^-

γ	$\gamma_0(c)$
$(\forall x.\varphi)^-$	$(\varphi[x/t])^-$
$(\exists x.\varphi)^+$	$(\varphi[x/t])^+$

δ	$\delta_0(c)$
$(\forall x.\varphi)^+$	$(\varphi[x/c])^+$
$(\exists x.\varphi)^-$	$(\varphi[x/c])^-$

Signed Formulas

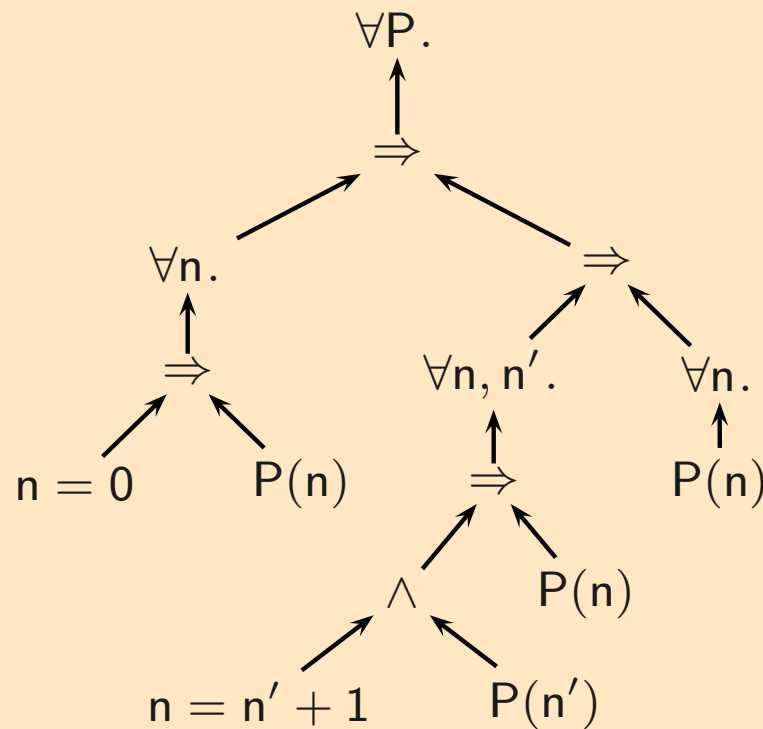


$$\forall P. (\forall n. n = 0 \Rightarrow P(n)) \Rightarrow ((\forall n, n'. (n = n' + 1 \wedge P(n')) \Rightarrow P(n)) \Rightarrow \forall n. P(n)) \vdash \forall n. \sum_{i=1}^n i^3 = \left(\sum_{i=1}^n i \right)^2$$

Signed Formulas



$$\forall P. (\forall n. n = 0 \Rightarrow P(n)) \Rightarrow ((\forall n, n'. (n = n' + 1 \wedge P(n')) \Rightarrow P(n)) \Rightarrow \forall n. P(n)) \vdash \forall n. \sum_{i=1}^n i^3 = \left(\sum_{i=1}^n i\right)^2$$

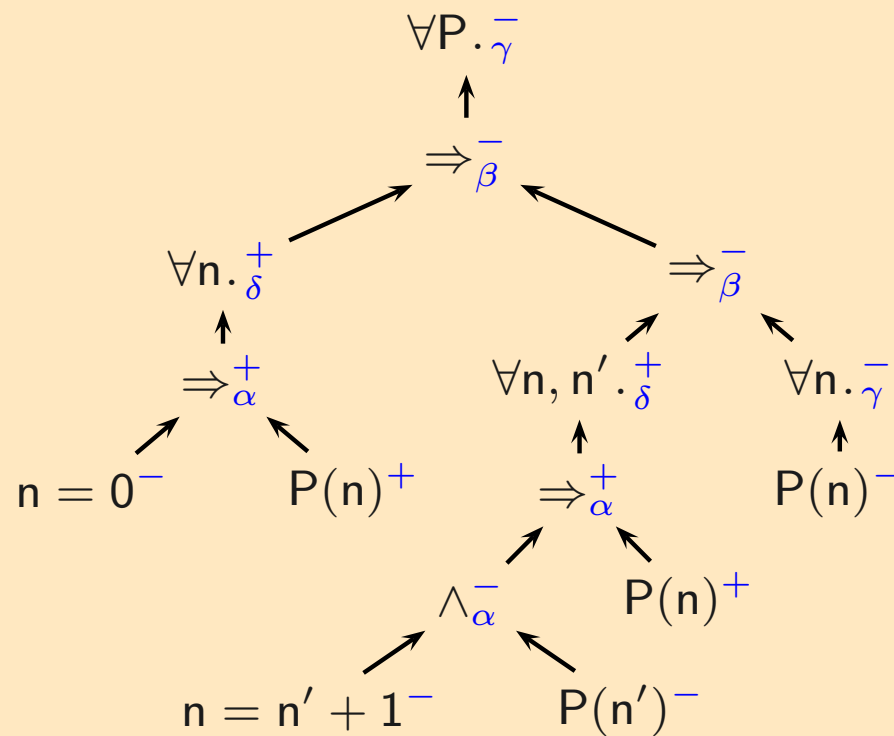


$$\sum_{i=1}^n i^3 = \left(\sum_{i=1}^n i\right)^2 \quad \forall n.$$

Signed Formulas



$$\forall P. (\forall n. n = 0 \Rightarrow P(n)) \Rightarrow ((\forall n, n'. (n = n' + 1 \wedge P(n')) \Rightarrow P(n)) \Rightarrow \forall n. P(n)) \vdash \forall n. \sum_{i=1}^n i^3 = \left(\sum_{i=1}^n i\right)^2$$

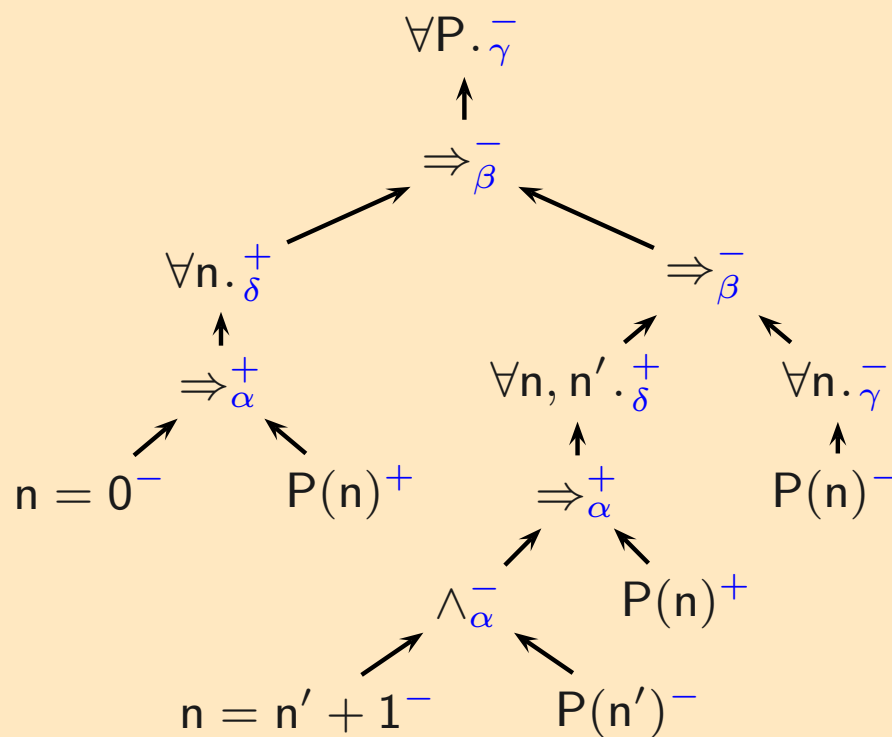


$$\forall n. \delta^+ \uparrow \sum_{i=1}^n i^3 = \left(\sum_{i=1}^n i\right)^{2+}$$

Signed Formulas



$$\forall P. (\forall n. n = 0 \Rightarrow P(n)) \Rightarrow ((\forall n, n'. (n = n' + 1 \wedge P(n')) \Rightarrow P(n)) \Rightarrow P(n)) \Rightarrow \forall n. P(n) \vdash \forall n. \sum_{i=1}^n i^3 = \left(\sum_{i=1}^n i\right)^2$$



$$\forall n. \delta^+ \uparrow \sum_{i=1}^n i^3 = \left(\sum_{i=1}^n i\right)^{2+}$$

Replacement rule for $\forall n. \sum_{i=1}^n i^3 = \left(\sum_{i=1}^n i\right)^{2+}$:

$$\forall n. P(n)^- \rightarrow \left\langle \forall n. n = 0 \Rightarrow P(n)^+, \quad \forall n, n'. (n = n' + 1 \wedge P(n')) \Rightarrow P(n)^+ \right\rangle$$

\implies Fix left-hand side $\forall n. P(n)^-$ and collect **all** β -related formulas

Application...



By substitution $[P \leftarrow \lambda x. \sum_{i=1}^x i^3 = (\sum_{i=1}^x i)^2]$ application of

$$\forall n. P(n)^- \rightarrow \left\langle \begin{array}{l} \forall n. n = 0 \Rightarrow P(n)^+, \\ \forall n, n'. (n = n' + 1 \wedge P(n')) \Rightarrow P(n)^+ \end{array} \right\rangle$$

to $\forall n. \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^{2+}$ yields

$$\forall n. n = 0 \Rightarrow \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2 \wedge$$

$$\forall n, n'. n = n' + 1 \wedge \sum_{i=1}^{n'} i^3 = (\sum_{i=1}^{n'} i)^2 \Rightarrow \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2$$

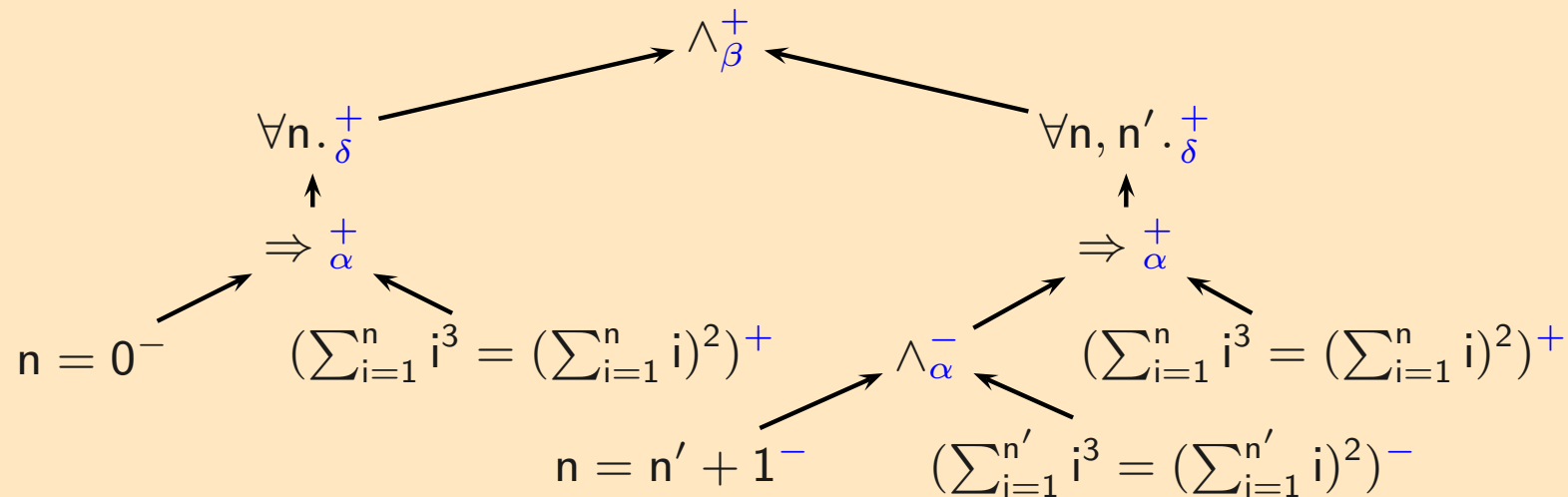
Now: How to enable application of $n = 0$?

Signed Formulas



$$((\forall n. (n = 0^- \Rightarrow (\sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2)^+)_\alpha^+)_\delta^+ \wedge$$

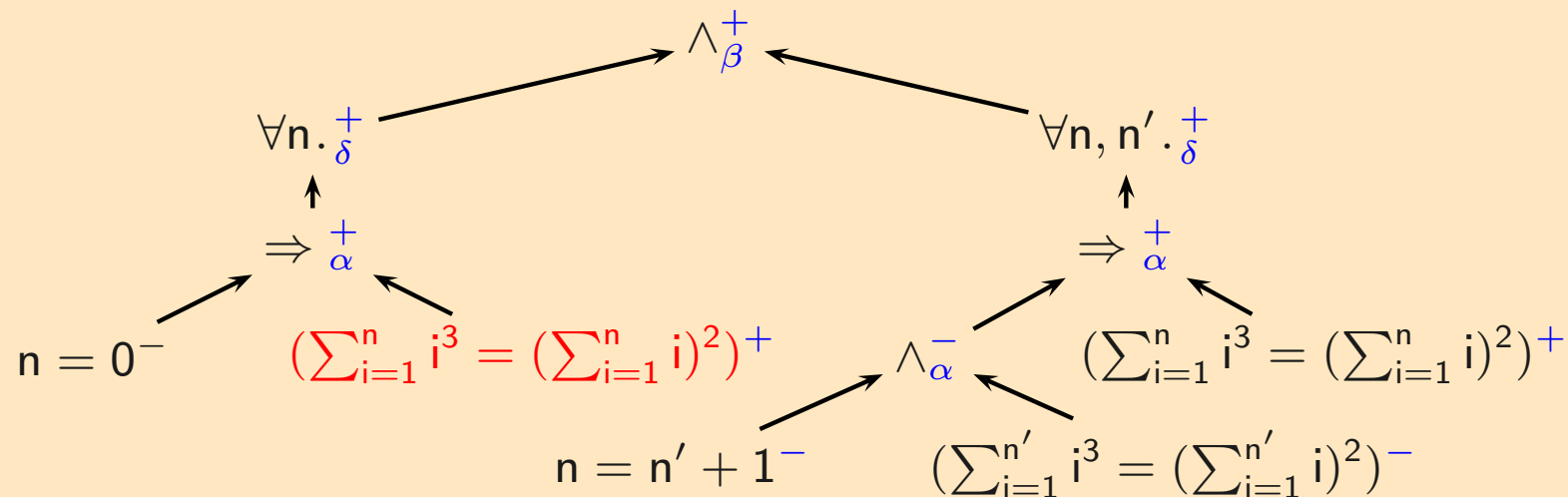
$$(\forall n, n'. ((n = n' + 1 \wedge (\sum_{i=1}^{n'} i^3 = (\sum_{i=1}^{n'} i)^2)^-)_\alpha^- \Rightarrow (\sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2)^+)_\delta^+)_\beta^+$$



Signed Formulas

$$((\forall n. (n = 0^- \Rightarrow (\sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2)^+)_\alpha^+)_\delta^+ \wedge$$

$$(\forall n, n'. ((n = n' + 1 \wedge (\sum_{i=1}^{n'} i^3 = (\sum_{i=1}^{n'} i)^2)^-)_\alpha^- \Rightarrow (\sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2)^+)_\delta^+)_\beta^+$$



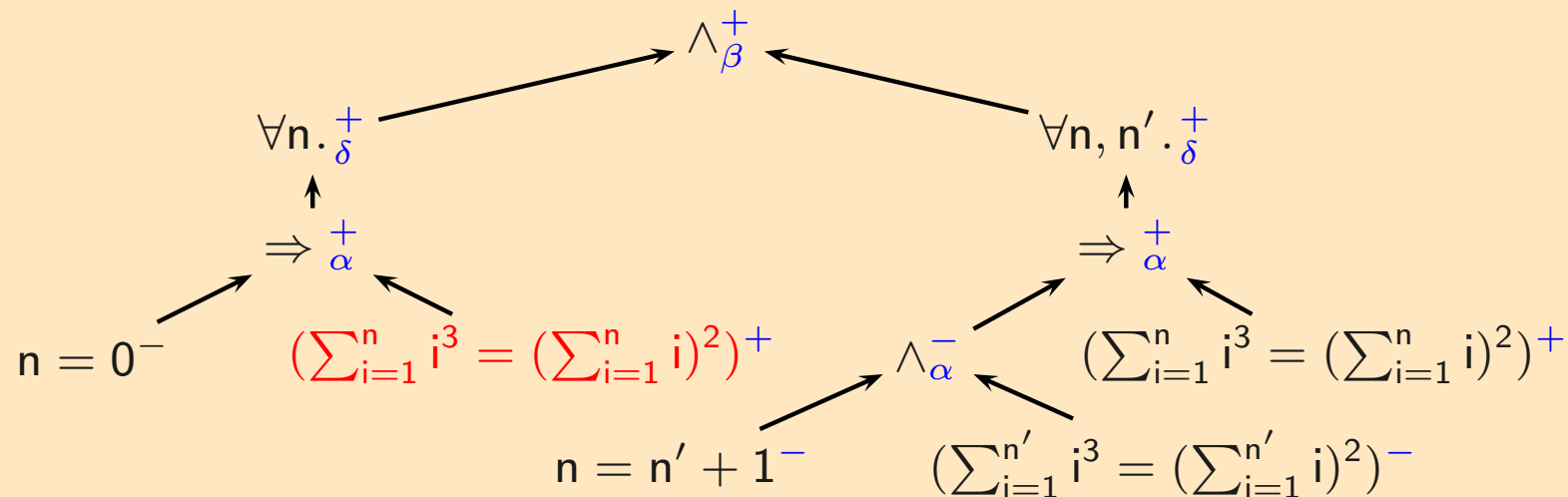
- Formulas connected by α -type node are in the same context

\implies Uniform and static notion of logical context of subformulas!

Signed Formulas

$$((\forall n. (n = 0^- \Rightarrow (\sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2)^+)_\alpha^+)_\delta^+ \wedge$$

$$(\forall n, n'. ((n = n' + 1 \wedge (\sum_{i=1}^{n'} i^3 = (\sum_{i=1}^{n'} i)^2)^-)_\alpha^- \Rightarrow (\sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2)^+)_\delta^+)_\beta^+$$



- Formulas connected by α -type node are in the same context

\implies Uniform and static notion of logical context of subformulas!

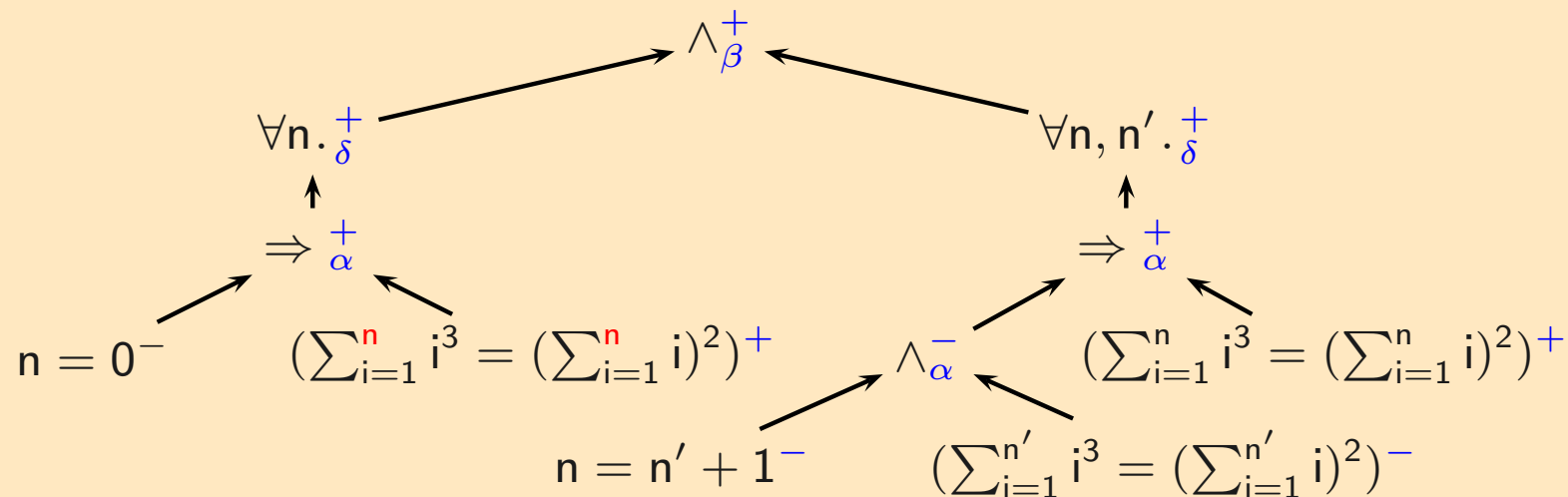
- Derive replacement rules from context

Example: $n \rightarrow 0$ for $(\sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2)^+$

Signed Formulas

$$((\forall n. (n = 0^- \Rightarrow (\sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2)^+)_\alpha^+)_\delta^+ \wedge$$

$$(\forall n, n'. ((n = n' + 1 \wedge (\sum_{i=1}^{n'} i^3 = (\sum_{i=1}^{n'} i)^2)^-)_\alpha^- \Rightarrow (\sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2)^+)_\delta^+)_\beta^+$$



- Formulas connected by **α-type node** are in the same context

⇒ Uniform and static notion of logical context of subformulas!

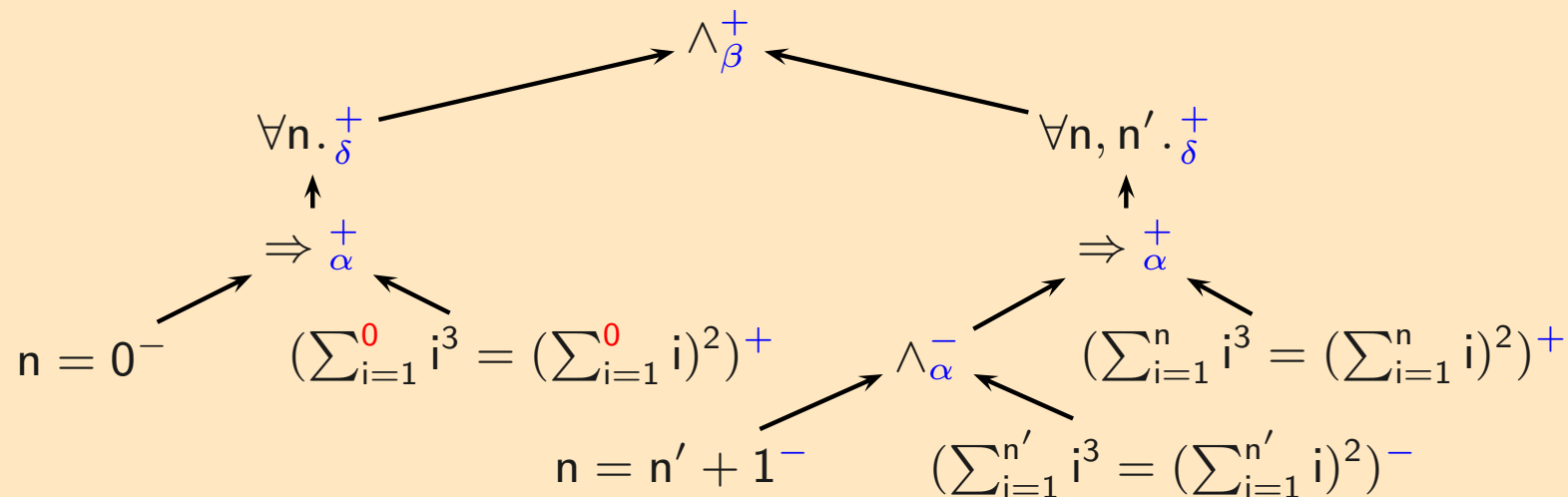
- Derive **replacement rules** from context

Example: $n \rightarrow 0$ for $(\sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2)^+$

Signed Formulas

$$((\forall n. (n = 0^- \Rightarrow (\sum_{i=1}^0 i^3 = (\sum_{i=1}^0 i)^2)^+)_\alpha^+)_\delta^+ \wedge$$

$$(\forall n, n'. ((n = n' + 1 \wedge (\sum_{i=1}^{n'} i^3 = (\sum_{i=1}^{n'} i)^2)^-)_\alpha^- \Rightarrow (\sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2)^+)_\delta^+)_\beta^+$$



- Formulas connected by α -type node are in the same context

\implies Uniform and static notion of logical context of subformulas!

- Derive replacement rules from context

Example: $n \rightarrow 0$ for $(\sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2)^+$

Turning this into a Calculus

- Signed formulas are close to extensional expansion trees (EET) [[Andrews81](#),[Miller83](#), [Pfenning87](#)] and Wallen's indexed formula trees (IFT)[[Wallen90](#)]
- Uniform integration of IFT and EET
 - ▶ as a starting point and
 - ▶ to deal with quantifiers and admissibility of substitution
- Take a free variable representation of our signed formula
- Define a calculus
 - ▶ starting with primitive rules for assertion application
 - ▶ Further rules to obtain a complete calculus

Extensional Expansion Trees



- Defined in [Pfenning87], minimal set of logical connectives, all negations pushed to the literals

Extensional Expansion Trees

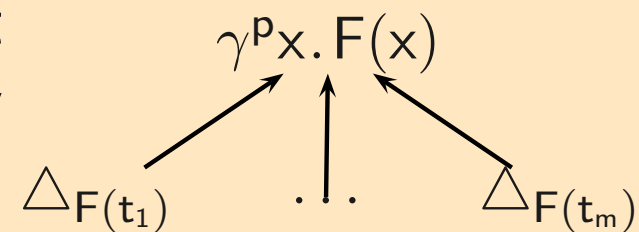


- Defined in [Pfenning87], minimal set of logical connectives, all negations pushed to the literals
- Tree construction follows exactly tree structure of the formula, except. . .

Extensional Expansion Trees

- Defined in [Pfenning87], minimal set of logical connectives, all negations pushed to the literals
- Tree construction follows exactly tree structure of the formula, except...

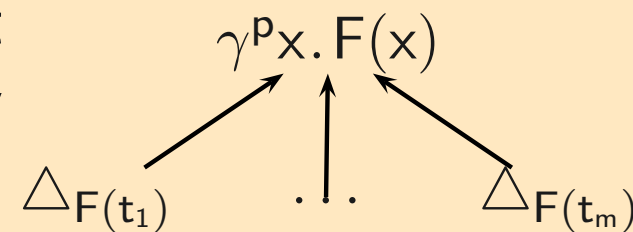
...for existential quantifiers where it allows to have subtrees for arbitrary many instances of the quantifier.



Extensional Expansion Trees

- Defined in [Pfenning87], minimal set of logical connectives, all negations pushed to the literals
- Tree construction follows exactly tree structure of the formula, except...

...for existential quantifiers where it allows to have subtrees for arbitrary many instances of the quantifier.

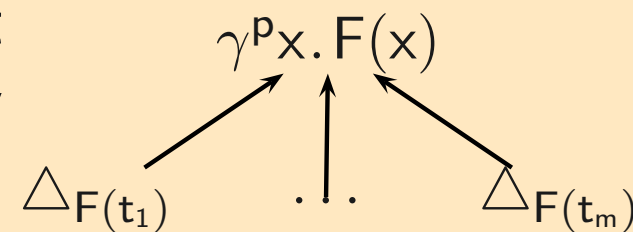


- Rule for functional and Boolean extensionality

Extensional Expansion Trees

- Defined in [Pfenning87], minimal set of logical connectives, all negations pushed to the literals
- Tree construction follows exactly tree structure of the formula, except...

...for existential quantifiers where it allows to have subtrees for arbitrary many instances of the quantifier.

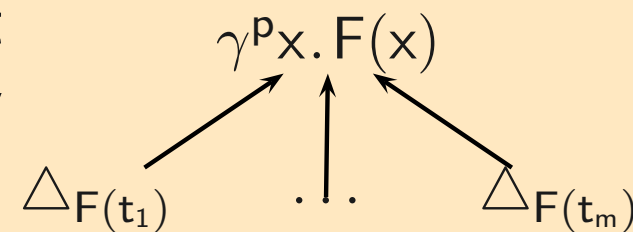


- Rule for functional and Boolean extensionality
- Δ_F^μ denotes EET for F and multiplicity μ

Extensional Expansion Trees

- Defined in [Pfenning87], minimal set of logical connectives, all negations pushed to the literals
- Tree construction follows exactly tree structure of the formula, except...

...for existential quantifiers where it allows to have subtrees for arbitrary many instances of the quantifier.



- Rule for functional and Boolean extensionality
- Δ_F^μ denotes EET for F and multiplicity μ
- Deep formula** $DF(\Delta_F^\mu)$: the formula extracted from Δ_F^μ by omitting all quantifiers.

$$DF \left(\begin{array}{c} \gamma^p x. F(x) \\ \swarrow \quad \uparrow \quad \searrow \\ \Delta_{F(t_1)} \quad \cdot \uparrow \cdot \quad \Delta_{F(t_m)} \end{array} \right) := \alpha(DF(\Delta_{F(t_1)}), \dots, DF(\Delta_{F(t_m)}))$$

Substitutions and Extensions

- Δ_F^μ -*admissibility* of substitution (instances of γ -quantifiers)
- An extensional expansion tree Δ_F^μ is an *extensional expansion proof* for F, if, and only if,
 1. its deep formula is a tautology (i.e., all paths through the deep formula are unsatisfiable) and
 2. the associated substitution σ is Δ_F^μ -admissible.
- *Extend* that calculus by
 1. allowing for rich set of logical connectives ($\Leftrightarrow, =, \top, \perp$)
 2. use polarities like in Wallen's IFT: overcome restrictions on the position of negations
 3. on the fly increase of the multiplicities
avoids "guessing" the initial multiplicities and restart.
- Deep formulas are *quantifier-free (QF) signed formulas*

[Tableau05]

Contexts

- *Context in QF-signed formulas*

Two subformulas related by an α -type connective are in the same logical context

- *Signed formulas obtained by Weakening $\mathcal{W}(F^p)$*

- ▶ $\mathcal{W}(A \vee (B \wedge C)^-) = \{A \vee (B \wedge C)^-, A \vee B^-, A \vee C^-\}$

- *Conditions of L^q in $\varphi(L^q)^p$:*

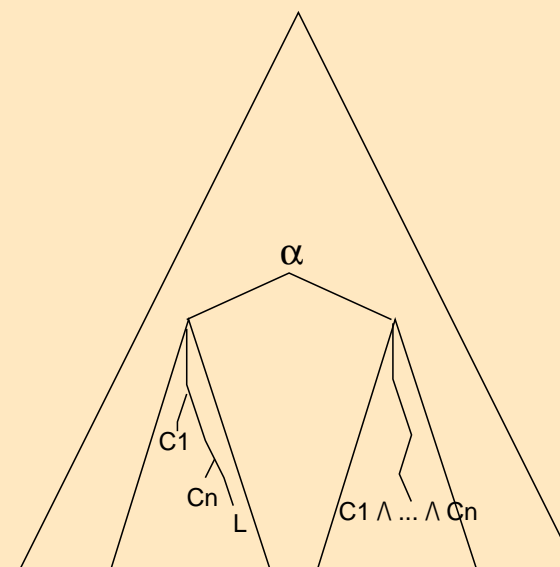
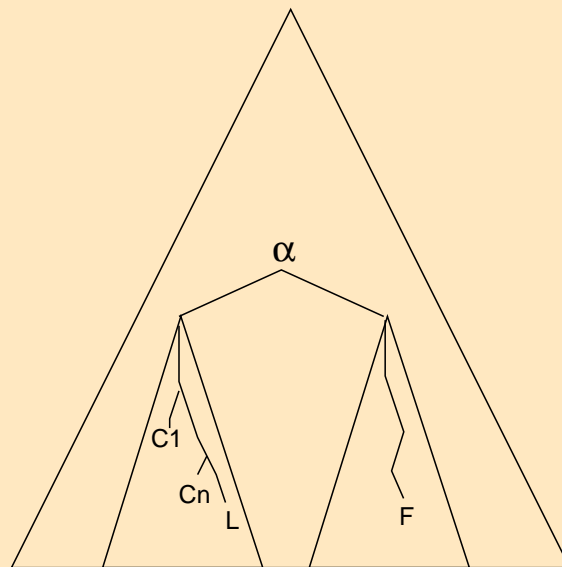
- ▶ If $G_1^{p_1}, \dots, G_n^{p_n}$ are all maximal signed formulas β -related to L^q in $\varphi(L^q)^p$

- ▶ Then the conditions of L^q are $\mathcal{C}_{\varphi(\cdot)^p} := \mathcal{W}(G_1^{p_1}) \times \dots \times \mathcal{W}(G_n^{p_n})$

Replacement rules

■ Resolution Replacement Rules

- ▶ Assume $\psi(\alpha(\varphi(L^{-p}), \varphi'(F^p)))$
- ▶ $(C_1^{p_1}, \dots, C_n^{p_n}) \in \mathcal{C}_{\varphi(\cdot)}$
- ▶ Then $L^{-p} \rightarrow \langle C_1^{p_1}, \dots, C_n^{p_n} \rangle$ is a replacement rule for F^p



Replacement rules

■ *Resolution Replacement Rules*

- ▶ Assume $\psi(\alpha(\varphi(L^{-p}), \varphi'(F^p)))$
- ▶ $(C_1^{p_1}, \dots, C_n^{p_n}) \in \mathcal{C}_{\varphi(\cdot)}$
- ▶ Then $L^{-p} \rightarrow \langle C_1^{p_1}, \dots, C_n^{p_n} \rangle$ is a replacement rule for F^p

■ *Rewriting Replacement Rules*

- ▶ Assume $\psi(\alpha(\varphi(u = v^-), \varphi'(s)))$ or $\psi(\alpha(\varphi(u \Leftrightarrow v^-), \varphi'(s)))$
- ▶ $(C_1^{p_1}, \dots, C_n^{p_n}) \in \mathcal{C}_{\varphi(\cdot)}$
- ▶ Then $u \rightarrow \langle v, C_1^{p_1}, \dots, C_n^{p_n} \rangle$ and $v \rightarrow \langle u, C_1^{p_1}, \dots, C_n^{p_n} \rangle$ are replacement rules for s

Replacement Rules

Examples for rules from $(A \vee B) \Rightarrow (C \wedge D)^-$

- $A^+ \rightarrow \langle C \wedge D^- \rangle$
- $B^+ \rightarrow \langle C \wedge D^- \rangle$
- $A \vee B^+ \rightarrow \langle C \wedge D^- \rangle$
- $C^- \rightarrow \langle A \vee B^+ \rangle$
- $D^- \rightarrow \langle A \vee B^+ \rangle$
- $C \wedge D^- \rightarrow \langle A \vee B^+ \rangle$

Weakening right-hand sides: From $C \wedge D^-$ also have rules with either C or D

- $A^+ \rightarrow \langle C^- \rangle, A^+ \rightarrow \langle D^- \rangle$
- Also: $C \wedge D^- \rightarrow \langle A^+ \rangle, C \wedge D^- \rightarrow \langle B^+ \rangle$

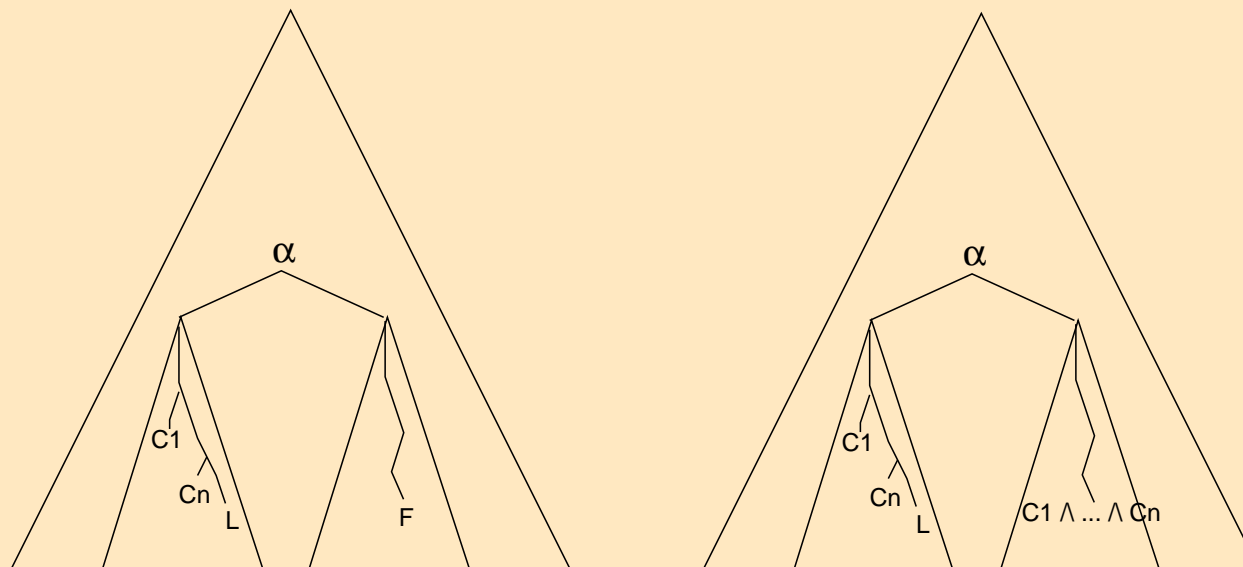
Weakening to enable rule application

- $A \vee B^+ \rightarrow \langle C \wedge D^- \rangle$ is applicable on $(A \vee E)^-$ to rewrite it to D
 - ▶ $A^+ \in \mathcal{W}(A \vee B^+)$ and $A^- \in \mathcal{W}(A \vee E^-)$

CoRE Calculus

Rule application

- Replace by conjunction (β) of subgoals by respecting polarities
- Example: From $((A \Rightarrow (B \Rightarrow (C \wedge D))) \wedge (B \vee E))^-$
by $B^+ \rightarrow \langle A^+, C \wedge D^- \rangle$ on $(B \vee E)$
we obtain $((A \Rightarrow (B \Rightarrow (C \wedge D))) \wedge (A \Rightarrow (C \wedge D)))^-$



CoRE Calculus Rules:

Given a conjecture F

- Construct Δ_F^1 with singular multiplicity and meta-variables
- Extract deep formula $DF(\Delta_F^1) := F_X$
- The initial proof state is $\Delta_F^1; \text{Id} \triangleright F_X$

Rules

$$\frac{\Delta; \sigma \triangleright G}{\Delta'; \sigma' \triangleright G'} R$$

(top-down, deep)

- Replacement rule application NNF Resolution and paramodulation style
- Rules from EET functional and Boolean extensionality,
Substitution of meta-variables, dynamic increase of multiplicities
- Rules to deal with richer set of connectives: Introduction of Leibniz' equality
Elimination of positive equivalences
- Structural rules Weakening, Contraction, propositional simplification
- Cut and Axiom

Axiom, Structural Rules & Positive Equivalences



$$\frac{\Delta; \sigma \triangleright \top}{\text{q.e.d.}} \textit{Axiom}$$

$$\frac{\Delta; \sigma \triangleright \varphi(F^p)}{\Delta; \sigma \triangleright \varphi(\alpha^p(F^p, F^p))} \textit{Contract}$$

$$\frac{\Delta; \sigma \triangleright \varphi(\alpha^p(F^q, G^r))}{\Delta; \sigma \triangleright \varphi(\bar{\alpha}^p(F^q))} \textit{Weak}_L$$

$$\frac{\Delta; \sigma \triangleright \varphi(\alpha^p(F^q, G^r))}{\Delta; \sigma \triangleright \varphi(\bar{\alpha}^p(G^r))} \textit{Weak}_R$$

$$\frac{\Delta_{(F \Leftrightarrow G)^+}; \sigma \triangleright \varphi((F \Leftrightarrow G)^+)}{\Delta_{\alpha((F \Leftrightarrow G)^+, \Delta_{(F \Rightarrow G) \wedge (G \Rightarrow F)^+})}; \sigma \triangleright \varphi(((F \Rightarrow G) \wedge (G \Rightarrow F))^+)} \zeta\textit{-Elim}$$

Rules for Equality/Equivalences



$$\frac{\Delta_{\Delta_{s=t^P}}; \sigma \triangleright \varphi(s = t^P)}{\Delta_{\alpha^P}(\Delta_{s=t^P}, \Delta_{\forall P. P(s) \Rightarrow P(t)}); \sigma \triangleright \varphi(\alpha^P(s = t, P(s) \Rightarrow P(t)))} \textit{Leibniz}$$

$$\frac{\Delta_{s=t^P}; \sigma \triangleright \varphi(s = t^P)}{\Delta_{\alpha^P}(s=t^P, \lambda x. s = \lambda x. t); \sigma \triangleright \varphi(\alpha^P(s = t, \lambda x. s = \lambda x. t))} \textit{f-Ext}$$

if x local to $s = t$ in $\varphi(s = t^P)$.

$$\frac{\Delta_{A \Leftrightarrow B^P}; \sigma \triangleright \varphi(A \Leftrightarrow B^P)}{\Delta_{\alpha^P}(A \Leftrightarrow B^P, \lambda x. A = \lambda x. B); \sigma \triangleright \varphi(\alpha^P(A \Leftrightarrow B, \lambda x. A = \lambda x. B))} \textit{b-Ext}$$

if x local to $A \Leftrightarrow B$ in $\varphi(A \Leftrightarrow B^P)$.

Substitution & Assertion Application



$$\frac{\Delta; \sigma \triangleright F}{\sigma'(\Delta); \sigma' \circ \sigma \triangleright \sigma'(F)} \textit{Subst}$$

if $\sigma' \circ \sigma$ admissible wrt. Δ .

$$\frac{\Delta; \sigma \triangleright \varphi(F^P)}{\Delta; \sigma \triangleright \varphi(\overline{\beta}^P(V_1^{P_1}, \dots, V_n^{P_n}))} \textit{Res}$$

If $U^{-P} \rightarrow \langle V_1^{P_1}, \dots, V_n^{P_n} \rangle$ admissible for F^P and U^{-P} and F^P are connectable.

$$\frac{\Delta_{\epsilon(s,t)}; \sigma \triangleright \varphi(L(s)^P)}{\Delta_{\alpha^-(\epsilon(s,t), \forall P. \beta^-(L(s)^{-P}, L(t)^P)); [\lambda x. L(x)/P] \circ \sigma \triangleright \varphi(\overline{\beta}^P(L(t)^P, V_1^{P_1}, \dots, V_n^{P_n}))} \textit{Rew}$$

If $L(s)^P$ is a literal, $s \rightarrow \langle t, V_1^{P_1}, \dots, V_n^{P_n} \rangle$ is admissible for $L(s)^P$, and $\epsilon(s, t)$ is the equation or equivalence this rule results from.

Multiplicity Adjustment & Cut



$$\frac{\Delta_{\Delta_1 \dots \Delta_n}; \sigma \triangleright \varphi(\psi_1(\vec{V}_1), \dots, \psi_n(\vec{V}_k))}{\Delta_{\Delta_1 \Delta'_1 \dots \Delta_n, \Delta'_n}; \sigma' \circ \sigma \triangleright \varphi(\alpha(\psi_1(\vec{V}_1), \psi_1(\rho(\vec{V}_1))), \dots, \alpha(\psi_n(\vec{V}_k), \psi_n(\rho(\vec{V}_k))))} \quad \mu\text{-Inc}$$

where ρ is the renaming of γ - and δ -variables declared in the Δ_i , $(\vec{V}_i)_{i=1\dots k}$ is a disjoint partition of $dom(\rho)$, such that for all i , all the variables in \vec{V}_i occur only in $\psi_i(\vec{V}_i)$, and $\sigma' := [\rho(\sigma(x)) / \rho(x) \mid \text{for all } \gamma\text{-variables } x \in dom(\rho)]$.

$$\frac{\Delta_{\Delta_{FP}}; \sigma \triangleright \varphi(F^P)}{\Delta_{\Delta_C}; \rho \circ \sigma \triangleright \varphi(\beta^P(\alpha^P(A^-, F^P), \alpha^P(A^+, F^P)))} \quad \text{Cut A}$$

where Δ_{FP} is the minimal subtree containing all literals in F^P , \vec{x}' are the free variables of A not bound above Δ_{FP} , $\rho := [\vec{x}' / \vec{x}]$, and Δ_C is the subtree $\gamma^P \vec{x}' . \beta^P(\alpha^P(A^-, \Delta_{FP}), \alpha^P(A^+, \Delta_{FP}))$.

Simplification



$$\begin{array}{ccccccc}
 \frac{\Delta; \sigma \triangleright \varphi(T \vee F)}{\Delta; \sigma \triangleright \varphi(T)} & \vee_L^T \frac{\Delta; \sigma \triangleright \varphi(F \vee T)}{\Delta; \sigma \triangleright \varphi(T)} & \vee_R^T \frac{\Delta; \sigma \triangleright \varphi(\perp \vee F)}{\Delta; \sigma \triangleright \varphi(F)} & \vee_L^\perp \frac{\Delta; \sigma \triangleright \varphi(F \vee \perp)}{\Delta; \sigma \triangleright \varphi(F)} & \vee_R^\perp & & \\
 \frac{\Delta; \sigma \triangleright \varphi(T \Rightarrow F)}{\Delta; \sigma \triangleright \varphi(F)} & \Rightarrow_L^T \frac{\Delta; \sigma \triangleright \varphi(F \Rightarrow T)}{\Delta; \sigma \triangleright \varphi(T)} & \Rightarrow_R^T \frac{\Delta; \sigma \triangleright \varphi(\perp \Rightarrow F)}{\Delta; \sigma \triangleright \varphi(T)} & \Rightarrow_L^\perp \frac{\Delta; \sigma \triangleright \varphi(F \Rightarrow \perp)}{\Delta; \sigma \triangleright \varphi(\neg(F))} & \Rightarrow_R^\perp & & \\
 \frac{\Delta; \sigma \triangleright \varphi(T \wedge F)}{\Delta; \sigma \triangleright \varphi(F)} & \wedge_L^T \frac{\Delta; \sigma \triangleright \varphi(F \wedge T)}{\Delta; \sigma \triangleright \varphi(F)} & \wedge_R^T \frac{\Delta; \sigma \triangleright \varphi(\perp \wedge F)}{\Delta; \sigma \triangleright \varphi(\perp)} & \wedge_L^\perp \frac{\Delta; \sigma \triangleright \varphi(F \wedge \perp)}{\Delta; \sigma \triangleright \varphi(\perp)} & \wedge_R^\perp & &
 \end{array}$$

Soundness and Completeness



Soundness:

- Define paths through the QF signed formula
- Initial QF signed formula is deep formula of initial EET
- Show for each rule that it preserves the existence of satisfiable paths

Completeness:

- From completeness of EEP: Guess the right initial multiplicity and substitution, + applications of *Leibniz*, *f-Ext*, *b-Ext*, *ζ-Elim*, and *Cut*.
- The resulting QF signed formula F_P is propositionally unsatisfiable.
- Show that we can reduce $\Delta_P; \sigma \triangleright F_P$ to $\Delta; \sigma \triangleright \top$ (Axiom rule)
 - ▶ *path resolution* [MurrayRosenthal87] is admissible using *Res*, *Contract*, *Weakening* and the simplification rules.
 - ▶ path resolution complete for propositional logic

Example

Assume Axioms and Lemmas

$$\forall p. \forall x. \exists y. (p(0) \wedge (p(y) \Rightarrow p(s(y)))) \Rightarrow p(x) \quad (1)$$

$$\forall n. \sum_{i=1}^0 i^n = 0 \quad (3)$$

$$\forall n, m. \sum_{i=1}^{s(m)} i^n = s(m)^n + \sum_{i=1}^m i^n \quad (4)$$

$$\forall a, b. (a + b)^2 = (a^2 + (2 \times b) \times a) + b^2 \quad (5)$$

$$\forall a, b, c. a = b \Rightarrow a + c = b + c \quad (6)$$

$$\forall n. \sum_{i=1}^n i^1 = \frac{n \times s(n)}{2} \quad (7)$$

$$\forall m. 2 \times \frac{m}{2} = m \quad (8)$$

$$\forall q. s(q)^3 = s(q)^2 + (q \times s(q)) \times s(q) \quad (9)$$

$$0 = (0)^2 \quad (10)$$

Conjecture: $\forall n. \sum_{i=1}^n i^3 = (\sum_{i=1}^n i^1)^2$

The initial proof state

$$\Delta_0; \text{id} \triangleright \left(\begin{array}{ll} (P(0) \wedge (P(y) \Rightarrow P(s(y)))) \Rightarrow P(X) & (1) \\ \sum_{i=1}^0 i^N = 0 & (3) \\ \sum_{i=1}^{s(M)} i^N = s(M)^N + \sum_{i=1}^M i^N & (4) \\ (A + B)^2 = (A^2 + (2 \times B) \times A) + B^2 & (5) \\ \end{array} \right. \left. \begin{array}{ll} A' = B' \Rightarrow A' + C' = B' + C' & (6) \\ \sum_{i=1}^{N'} i^1 = \frac{N' \times s(N')}{2} & (7) \\ 2 \times \frac{M'}{2} = M' & (8) \\ s(Q)^3 = s(Q)^2 + (Q \times s(Q)) \times s(Q) & (9) \\ 0 = (0)^2 & (10) \\ \end{array} \right)$$

$$\Rightarrow \sum_{i=1}^n i^3 = (\sum_{i=1}^n i^1)^2$$

Example Proof

1. By *RuleApplication from* (1) $(P(0) \wedge (P(y) \Rightarrow P(s(y)))) \Rightarrow P(X)$

$\Delta_1; \sigma_1 \triangleright$

$$\left((1) \wedge (3) \wedge (4) \wedge (5) \wedge (6) \wedge (7) \wedge (8) \wedge (9) \wedge (10) \right)$$

$$\Rightarrow (y' = 0 \Rightarrow \sum_{i=1}^{y'} i^3 = (\sum_{i=1}^{y'} i^1)^2) \wedge (\sum_{i=1}^{y'} i^3 = (\sum_{i=1}^{y'} i^1)^2) \Rightarrow (\sum_{i=1}^{s(y')} i^3 = (\sum_{i=1}^{s(y')} i^1)^2)$$

2. By *RuleApplication from* $y' = 0$

$\Delta_1; \sigma_1 \triangleright$

$$\left((1) \wedge \sum_{i=1}^0 i^N = 0 \wedge (4) \wedge (5) \wedge (6) \wedge (7) \wedge (8) \wedge (9) \wedge (10) \right)$$

$$\Rightarrow (y' = 0 \Rightarrow \sum_{i=1}^0 i^3 = (\sum_{i=1}^0 i^1)^2) \wedge (\sum_{i=1}^{y'} i^3 = (\sum_{i=1}^{y'} i^1)^2) \Rightarrow (\sum_{i=1}^{s(y')} i^3 = (\sum_{i=1}^{s(y')} i^1)^2)$$

3. By *RuleApplication from* (3) $\sum_{i=1}^0 i^N = 0(2 \times)$

$\Delta_2; \sigma_2 \triangleright$

$$\left((1) \wedge (3) \wedge (4) \wedge (5) \wedge (6) \wedge (7) \wedge (8) \wedge (9) \wedge 0 = (0)^2 \right)$$

$$\Rightarrow (y' = 0 \Rightarrow 0 = (0)^2) \wedge (\sum_{i=1}^{y'} i^3 = (\sum_{i=1}^{y'} i^1)^2) \Rightarrow (\sum_{i=1}^{s(y')} i^3 = (\sum_{i=1}^{s(y')} i^1)^2)$$

Example Proof (cont'd)

4. By *RuleApplication from* (10) $0 = (0)^2$

$$\Delta_3; \sigma_3 \triangleright \left((1) \wedge (3) \wedge (4) \wedge (5) \wedge (6) \wedge (7) \wedge (8) \wedge (9) \wedge (10) \right)$$

$$\Rightarrow (y' = 0 \Rightarrow \top) \wedge (\sum_{i=1}^{y'} i^3 = (\sum_{i=1}^{y'} i^1)^2) \Rightarrow (\sum_{i=1}^{s(y')} i^3 = (\sum_{i=1}^{s(y')} i^1)^2)$$

5. By *Simplify with* \Rightarrow_R^T and \wedge_L^T

$\Delta_4; \sigma_4 \triangleright$

$$\left((1) \wedge (3) \wedge \sum_{i=1}^{s(M)} i^N = s(M)^N + \sum_{i=1}^M i^N \wedge (5) \wedge (6) \wedge (7) \wedge (8) \wedge (9) \wedge (10) \right)$$

$$\Rightarrow (\sum_{i=1}^{y'} i^3 = (\sum_{i=1}^{y'} i^1)^2) \Rightarrow (\sum_{i=1}^{s(y')} i^3 = (\sum_{i=1}^{s(y')} i^1)^2)$$

⋮

Compare first two Steps to...



To prove $\forall n. \sum_{i=1}^n i^3 = \left(\sum_{i=1}^n i\right)^2$

we apply $\forall P. (\forall n. n = 0 \Rightarrow P(n)) \Rightarrow ((\forall n, n'. (n = n' + 1 \wedge P(n')) \Rightarrow P(n)) \Rightarrow \forall n. P(n))$

to obtain $\forall n. n = 0 \Rightarrow \sum_{i=1}^n i^3 = \left(\sum_{i=1}^n i\right)^2 \wedge$
 $\forall n, n'. n = n' + 1 \wedge \sum_{i=1}^{n'} i^3 = \left(\sum_{i=1}^{n'} i\right)^2 \Rightarrow \sum_{i=1}^n i^3 = \left(\sum_{i=1}^n i\right)^2$

then apply $n = 0$

to obtain $\forall n. n = 0 \Rightarrow \sum_{i=1}^0 i^3 = \left(\sum_{i=1}^0 i\right)^2 \wedge$
 $\forall n, n'. n = n' + 1 \wedge \sum_{i=1}^{n'} i^3 = \left(\sum_{i=1}^{n'} i\right)^2 \Rightarrow \sum_{i=1}^n i^3 = \left(\sum_{i=1}^n i\right)^2$

Sequent Calculus Proof



$$\begin{array}{c}
 \vdots \\
 \hline
 n = n' + 1 \wedge \sum_{i=1}^{n'} i^3 = (\sum_{i=1}^{n'} i)^2 \vdash \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2 \\
 \hline
 \vdash (n = n' + 1 \wedge \sum_{i=1}^{n'} i^3 = (\sum_{i=1}^{n'} i)^2) \\
 \Rightarrow \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2 \\
 \hline
 \vdash \forall n, n'. (n = n' + 1 \wedge \sum_{i=1}^{n'} i^3 = (\sum_{i=1}^{n'} i)^2) \\
 \Rightarrow \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2 \\
 \hline
 \vdash \forall n, n'. (n = n' + 1 \wedge \sum_{i=1}^{n'} i^3 = (\sum_{i=1}^{n'} i)^2) \\
 \Rightarrow \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2, \forall n. \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2 \\
 \hline
 (\forall n, n'. (n = n' + 1 \wedge \sum_{i=1}^{n'} i^3 = (\sum_{i=1}^{n'} i)^2)) \\
 \Rightarrow \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2 \Rightarrow \forall n. \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2 \\
 \vdash \forall n. \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2 \\
 \hline
 (\forall n. n = 0 \Rightarrow \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2) \\
 \Rightarrow ((\forall n, n'. (n = n' + 1 \wedge \sum_{i=1}^{n'} i^3 = (\sum_{i=1}^{n'} i)^2)) \\
 \Rightarrow \forall n. \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2) \vdash \forall n. \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2 \\
 \hline
 \forall \mathbf{P}. (\forall n. n = 0 \Rightarrow \mathbf{P}(n)) \\
 \Rightarrow ((\forall n, n'. (n = n' + 1 \wedge \mathbf{P}(n'))) \Rightarrow \forall n. \mathbf{P}(n)) \vdash \forall n. \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2
 \end{array}$$

$$\begin{array}{c}
 \vdots \\
 \hline
 n = 0 \vdash \sum_{i=1}^0 i^3 = (\sum_{i=1}^0 i)^2, \\
 \forall n. \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2 \\
 \hline
 n = 0 \vdash \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2, \\
 \forall n. \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2 \\
 \hline
 \vdash n = 0 \Rightarrow \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2, \\
 \forall n. \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2 \\
 \hline
 \vdash \forall n. n = 0 \Rightarrow \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2, \\
 \forall n. \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2 \\
 \hline
 \vdash \forall n. n = 0 \Rightarrow \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2, \\
 \forall n. \sum_{i=1}^n i^3 = (\sum_{i=1}^n i)^2
 \end{array}$$

Related Work



- Deep Inference paradigm for calculi:
 - ▶ Calculus of Structures (CoS) [Guglielmi,Brünnler04, ...]
 - ▶ Used to study proof theoretic properties, no assertion level reasoning
 - ▶ CORE calculus supports reasoning with assertions, no proof theoretic studies
 - ▶ Maybe combination/integration with CoS could provide a basis for proof checking

- Focusing Proof Construction [Andreoli00]
 - ▶ Derives sequent calculus macro steps from available assertions.
 - ▶ Derived rules can only be applied on top-level formulas

- IMPS “macetes” [FarmerGuttmanThayer1993]
 - ▶ Builtin procedures to extract very specific application procedures from theorems.

Implementation



- It is implemented in Common Lisp and provides all major features
 - ▶ determine available assertions for arbitrary subformulas
 - ▶ determine possibilities to apply an assertion on arbitrary subformulas
 - ▶ apply assertions on arbitrary subformulas
 - ▶ get additional renamed copies of formulas via dynamic increase of multiplicities

to the Task-Level of Ω MEGA

- ... and thus to the automatic procedures like the proof planner, to the Ω ANTS.

Conclusion

CoRE Calculus where

- Assertion application is primitive rule (closer to CML, more “mathural”)
- Assertion application is actively supported
- Deep structural
- Technically assertion application corresponds to NNF resolution and paramodulation
 - ▶ Interactive proof construction: Technicalities of assertion application are hidden from the user
 - ▶ Automated proof construction: Directly presentable as a proof at the level of assertions

Here we focused on higher-order logic (Cut not admissible–yet!).

The formulations (Cut-free) for many first-order modal logics considered in [\[Wallen90\]](#) can be found in [\[AutexierPhD03\]](#)

Future Work

- Define CoRE calculi for further logics, for instance intuitionistic logic
- Define cut-free version of CoRE for HOL (mostly done)
- Define CoRE with primitive equality (so far via Leibniz) [\[Brown,Cade05\]](#)
- Transformations of CoRE proofs into SK or CoS proofs (proof checking)
- Add Schütte-style proof splits (structuring proofs into subproofs)
- Deduction modulo [\[DowekHardinKirchner98\]](#) to enable rule application:
 - ▶ Wish: Take local context into account for Deduction modulo in place
 - ▶ Difficulty: Rules from different contexts for left-hand side of rule and application position
- Investigate automated reasoning procedures for CoRE
 - Ordering-based techniques like [\[GanzingerStuber03\]](#) for NNF

More Future...

- *Integrate/combine CORE and Calculus of Structures (CoS)*
 - ▶ Get independent from expansion trees
 - ▶ “Inherit” nice properties of CoS (shorter proofs)
- *Questions:* Get rid of expansion trees or not?
 - ▶ *With expansion trees* in the background:
 - Expansion trees deal efficiently with quantifier dependencies
 - Allows to have free variable formulas to work with
 - Quantifiers are not visible (except those below equivalences)
 - Unclear explanation of mult. increase just from free variable formula
 - ▶ *Without expansion trees* (only formulas):
 - “Cleaner” definition of the calculus, closer to CoS
 - Requires to move quantifiers around
 - Quantifiers are moved by rule application which may introduce unnecessary dependencies

Demo



- test2 for resolution rule
- test3a or test3b for multiplicity
- test6 for rewriting below quantifier