Classical Higher-Order Logic
– Theory, Applications and Problems –

Christoph E. Benzmüller

University of Cambridge, UK & Universität des Saarlandes, Germany

Saarbrücken, 20 December 2006

Research Interest in AI


Can machines think?

Research Interest in AI

 Can machines think?

At the end of the century, [. . .] one will be able to speak of "machines thinking" without expecting to be contradicted.

Alan Turing, 1950

Research Interest in AI

Can machines think?

At the end of the century, [. . .] one will be able to speak of "machines thinking" without expecting to be contradicted.

Alan Turing, 1950

The last match
man vs machine?

Can machines play chess?

Research Interest in AI

Can machines think?

At the end of the century, [...] one will be able to speak of "machines thinking" without expecting to be contradicted.

Alan Turing, 1950

The last match
man vs machine?

Can machines play chess?

And how about mathematics?
Can we built intelligent
Mathematics Assistant Systems?

Overall Scientific Goal

Mathematics Assistance Systems

Mathematics Assistance Systems

- Computing

Mathematics Assistance Systems

- Computing
- Proving

Overall Scientific Goal

Mathematics Assistance Systems

- Computing
- Proving
- Exploring/Inventing

Overall Scientific Goal

Mathematics Assistance Systems

- Computing
- Proving
- Exploring/Inventing
- Illustrating/Publishing

Overall Scientific Goal

Mathematics Assistance Systems

- Computing
- Proving
- Exploring/Inventing
- Illustrating/Publishing
- Structuring/Organizing

Overall Scientific Goal

Mathematics Assistance Systems

- Computing
- Proving
- Exploring/Inventing
- Illustrating/Publishing
- Structuring/Organizing
- Explaining/Teaching

Overall Scientific Goal

Mathematics Assistance Systems

- Computing
- Proving
- Exploring/Inventing
- Illustrating/Publishing
- Structuring/Organizing
- Explaining/Teaching
- . . .

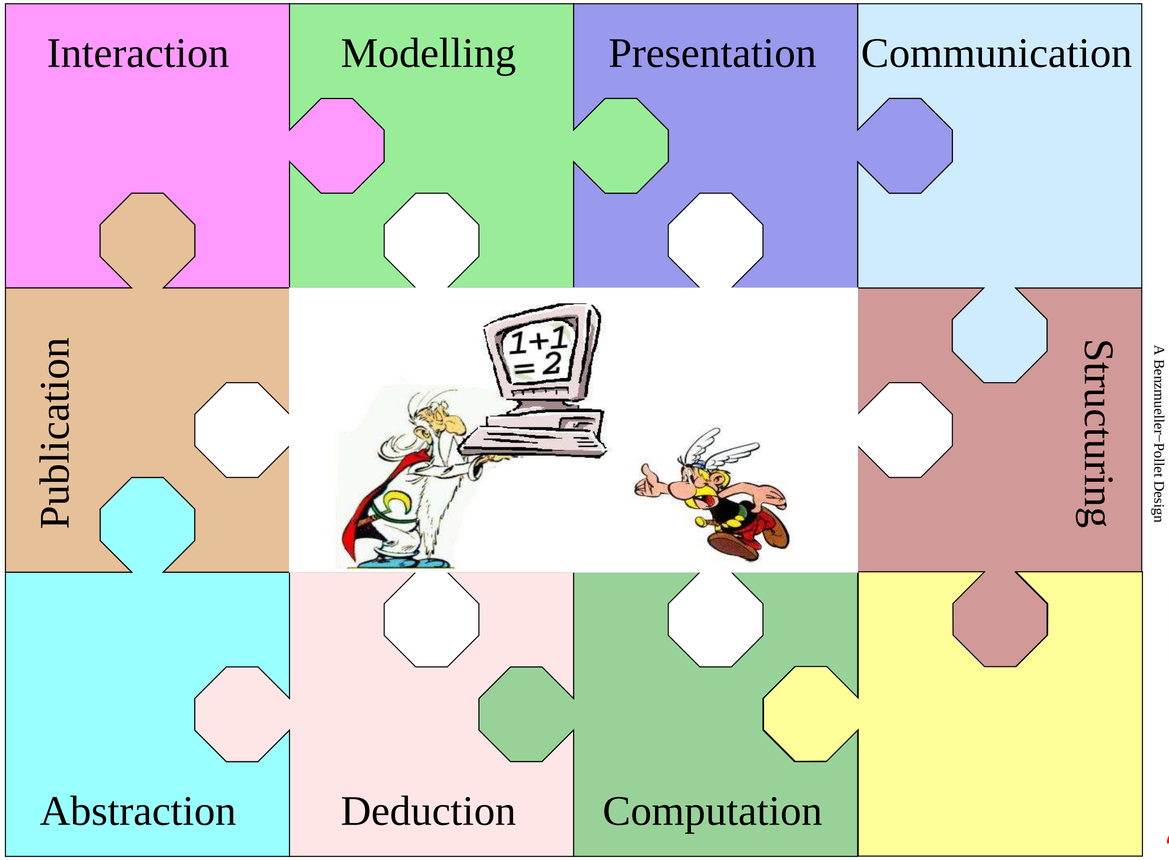
Overall Scientific Goal

Mathematics Assistance Systems

- Computing
- Proving
- Exploring/Inventing
- Illustrating/Publishing
- Structuring/Organizing
- Explaining/Teaching
- ...
- Architecture & HCI & ...

Overall Scientific Goal

Mathematics Assistance Systems



. . .

Architecture & HCI & . . .

Overall Scientific Goal

Mathematics Assistance Systems

see publication list

- Computing
- Proving
- Exploring/Inventing
- Illustrating/Publishing
- Structuring/Organizing
- Explaining/Teaching
- . . .
- (Architecture & HCI & . . .)

Overall Scientific Goal

Applications/Specialisations of Mathematics Assistance Systems

Formal Methods	Formal Methods	. . .	E-Learning in all
in Computer Science	in Mathematics	. . .	of these areas

Overall Scientific Goal

Applications/Specialisations of Mathematics Assistance Systems

Formal Methods in Computer Science	Formal Methods in Mathematics	... ...	E-Learning in all of these areas

Why Classical Higher-Order Logic (HOL)?

	textbooks	higher-order logic	first-order logic
$\mathcal{P}(A)$	$\{x \mid x \subseteq A\}$	$\lambda x.x \subseteq A$	$x \in \mathcal{P}(A) \Leftrightarrow x \subseteq A$

Overall Scientific Goal

Applications/Specialisations of Mathematics Assistance Systems

Formal Methods in Computer Science	Formal Methods in Mathematics	...	E-Learning in all of these areas

Why Classical Higher-Order Logic (HOL)?

	textbooks	higher-order logic	first-order logic
$\mathcal{P}(A)$	$\{x \mid x \subseteq A\}$	$\lambda x. x \subseteq A$	$x \in \mathcal{P}(A) \Leftrightarrow x \subseteq A$

A Big Challenge

Automation of HOL (research is decades behind)

Own Research in HOL

Automated Theorem Proving

Own Research in HOL

Automated Theorem Proving

- Extensional Resolution, Paramodulation

Own Research in HOL

Semantics

Automated Theorem Proving

- Extensional Resolution, Paramodulation

Own Research in HOL

Semantics

- Model Classes (Extensionality)

Automated Theorem Proving

- Extensional Resolution, Paramodulation

Own Research in HOL

Semantics

- Model Classes (Extensionality)
- Abstract Consistency Proof Method

Automated Theorem Proving

- Extensional Resolution, Paramodulation

Own Research in HOL

Semantics

- Model Classes (Extensionality)
- Abstract Consistency Proof Method
- Test Problems

Automated Theorem Proving

- Extensional Resolution, Paramodulation

Own Research in HOL

Semantics

- Model Classes (Extensionality)
- Abstract Consistency Proof Method
- Test Problems

Automated Theorem Proving

- Extensional Resolution, Paramodulation
- Combination with FO-ATP

Own Research in HOL

Semantics

- Model Classes (Extensionality)
- Abstract Consistency Proof Method
- Test Problems

Proof Theory

Automated Theorem Proving

- Extensional Resolution, Paramodulation
- Combination with FO-ATP

Own Research in HOL

Semantics

- Model Classes (Extensionality)
- Abstract Consistency Proof Method
- Test Problems

Proof Theory

- Cut-simulation

Automated Theorem Proving

- Extensional Resolution, Paramodulation
- Combination with FO-ATP

Own Research in HOL

Semantics — ESSLLI-06, WS-05/06

- Model Classes (Extensionality) [JSL'04]
- Abstract Consistency Proof Method [JSL'04]
- Test Problems [TPHOLs'05]

Proof Theory

- Cut-simulation [IJCAR'06]

Automated Theorem Proving — SS-06 (DA), WS-04/05

- Extensional Resolution, Paramod. [CADE'98/99,Synthese'02]
- Combination with FO-ATP [LPAR'04]

Syntax

Syntax

HOL-Syntax: Simple Typed λ-Calculus

Simple Types $\mathcal{T}$:	o	(truth values)
	ι	(individuals)
	$(\alpha \longrightarrow \beta)$	(functions from α to β)

HOL-Syntax: Simple Typed λ-Calculus

Simple Types $\mathcal{T}$:

o	(truth values)
ι	(individuals)
$(\alpha \longrightarrow \beta)$	(functions from α to β)

Typed Terms:

X_α	Variables ($\mathcal{V}$)
c_α	Constants & Parameters (Σ & $\mathcal{P}$)
$(\mathbf{F}_{\alpha\longrightarrow\beta}\,\mathbf{B}_\alpha)_\beta$	Application
$(\lambda Y_\alpha\,\mathbf{A}_\beta)_{\alpha\longrightarrow\beta}$	λ-abstraction

HOL-Syntax: Simple Typed λ-Calculus

Simple Types $\mathcal{T}$:

o	(truth values)
ι	(individuals)
$(\alpha \rightarrow \beta)$	(functions from α to β)

Typed Terms:

X_α	Variables ($\mathcal{V}$)
c_α	Constants & Parameters (Σ & $\mathcal{P}$)
$(\mathbf{F}_{\alpha\rightarrow\beta}\,\mathbf{B}_\alpha)_\beta$	Application
$(\lambda Y_\alpha\,\mathbf{A}_\beta)_{\alpha\rightarrow\beta}$	λ-abstraction

Equality of Terms: α, β, η

HOL: Adding Logical Connectives

$\top_o$ – true

$\bot_o$ – false

$\neg_{o\rightarrow o}$ – negation

$\vee_{o\rightarrow o\rightarrow o}$ – disjunction

$\wedge_{o\rightarrow o\rightarrow o}$ – conjunction

$\Rightarrow_{o\rightarrow o\rightarrow o}$ – implication

$\Leftrightarrow_{o\rightarrow o\rightarrow o}$ – equivalence

$\forall X_\alpha . \ldots$ – universal quantification over type α ($\forall$ types α)

$\exists X_\alpha . \ldots$ – existential quantification over type α ($\forall$ types α)

$=_{\alpha\rightarrow\alpha\rightarrow o}$ – equality at type α ($\forall$ types α)

HOL: Adding Logical Connectives

$\neg_{o \to o}$ – negation

$\vee_{o \to o \to o}$ – disjunction

$\forall X_\alpha \blacksquare \ldots$ – universal quantification over type α $\qquad$ ($\forall$ types α)

HOL: Leibniz Equality

Impredicative definition of equality

$\mathbf{A}_\alpha \doteq \mathbf{B}_\alpha$ means $\forall \mathsf{P}_{\alpha \longrightarrow o}(\mathsf{P}\,\mathbf{A} \Rightarrow \mathsf{P}\,\mathbf{B})$

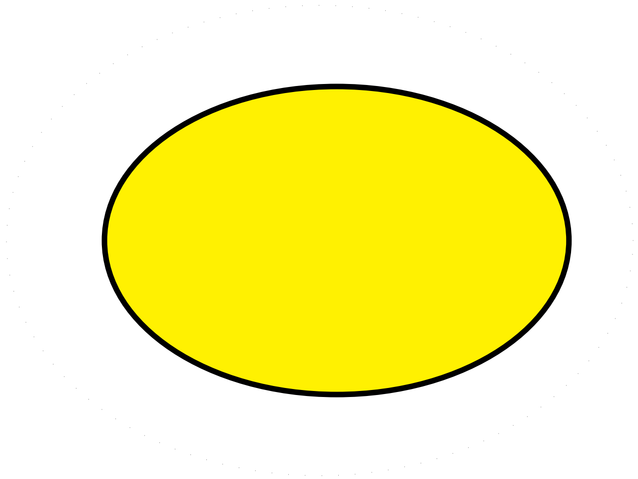
$\forall \mathsf{P}_{\alpha \longrightarrow o}(\neg \mathsf{P}\,\mathbf{A} \vee \mathsf{P}\,\mathbf{B})$

Semantics

Model Classes

(Extensionality)

Model Classes (Extensionality)



Standard Models $\mathfrak{ST}(\Sigma)$

- Idea of Standard Semantics:

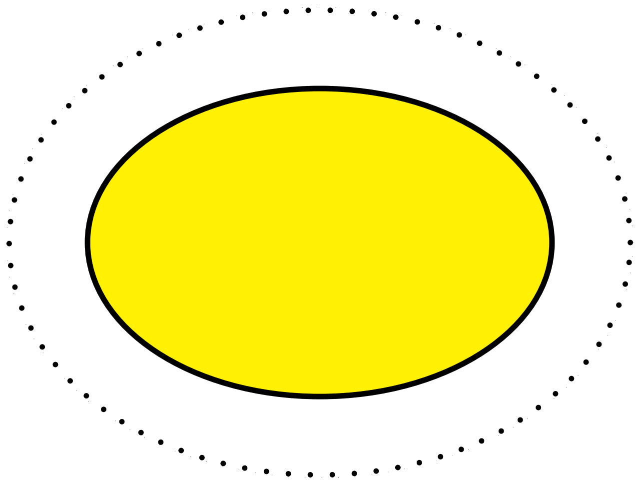
$$\iota \longrightarrow \mathcal{D}_\iota \qquad \text{(choose)}$$

$$o \longrightarrow \mathcal{D}_o = \{\mathtt{T}, \mathtt{F}\} \qquad \text{(fixed)}$$

$$(\alpha \to \beta) \longrightarrow$$

$$\mathcal{D}_{\alpha\to\beta} = \mathcal{F}(\mathcal{D}_\alpha, \mathcal{D}_\beta) \qquad \text{(fixed)}$$

Model Classes (Extensionality)



Standard Models $\mathfrak{ST}(\Sigma)$

- Idea of Standard Semantics:

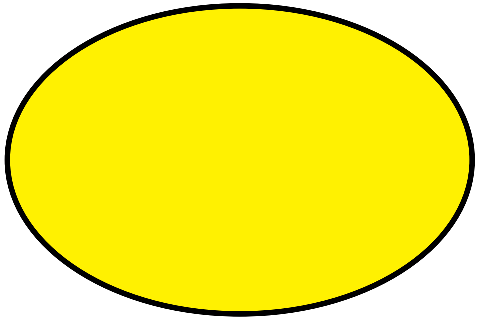
$$\iota \longrightarrow \mathcal{D}_\iota \qquad \text{(choose)}$$
$$o \longrightarrow \mathcal{D}_o = \{\mathtt{T}, \mathtt{F}\} \qquad \text{(fixed)}$$
$$(\alpha \to \beta) \longrightarrow$$
$$\mathcal{D}_{\alpha\to\beta} = \mathcal{F}(\mathcal{D}_\alpha, \mathcal{D}_\beta) \qquad \text{(fixed)}$$

- Henkin's Generalization:

$$\mathcal{D}_{\alpha\to\beta} \subseteq \mathcal{F}(\mathcal{D}_\alpha, \mathcal{D}_\beta) \qquad \text{(choose)}$$
but elements are still functions!

[Henkin-50]

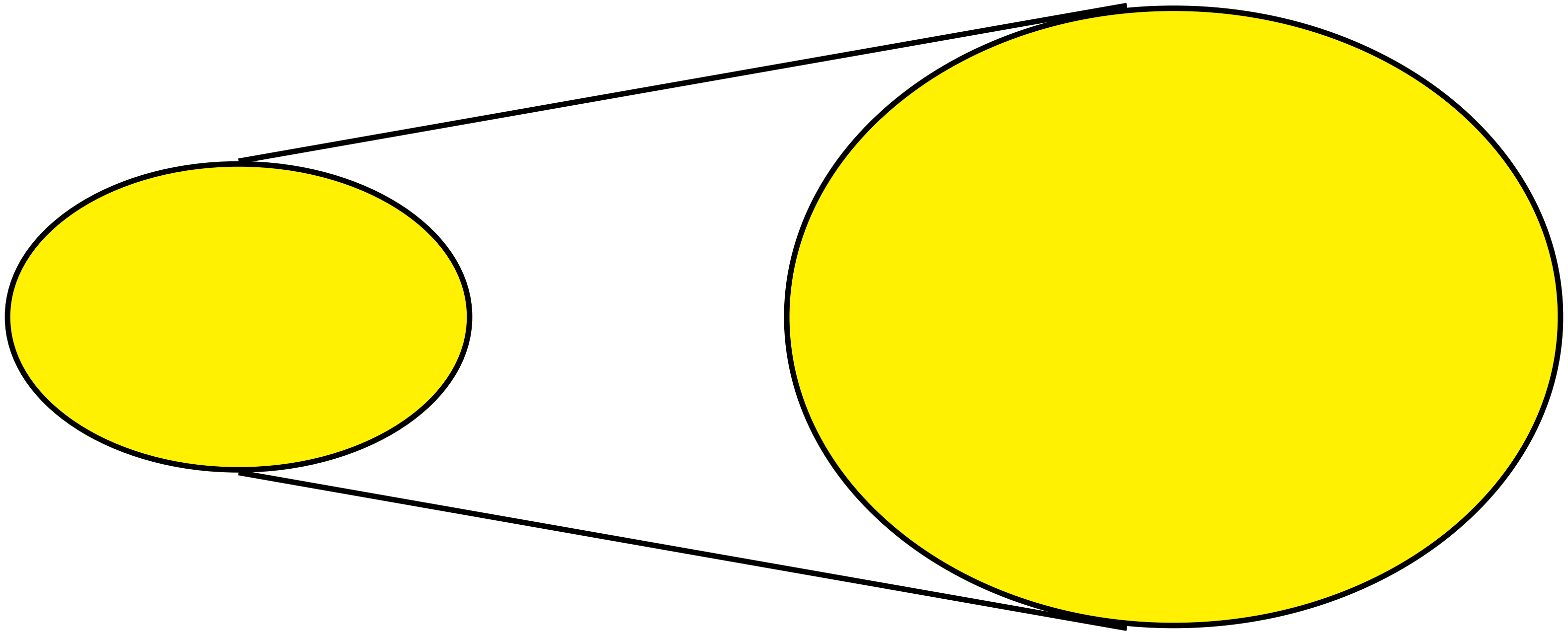
Model Classes (Extensionality)



Standard Models $\mathfrak{ST}(\Sigma)$

choose: $\mathcal{D}_\iota$
fixed: $\mathcal{D}_o, \mathcal{D}_{\alpha\rightarrow\beta}$, functions

Model Classes (Extensionality)

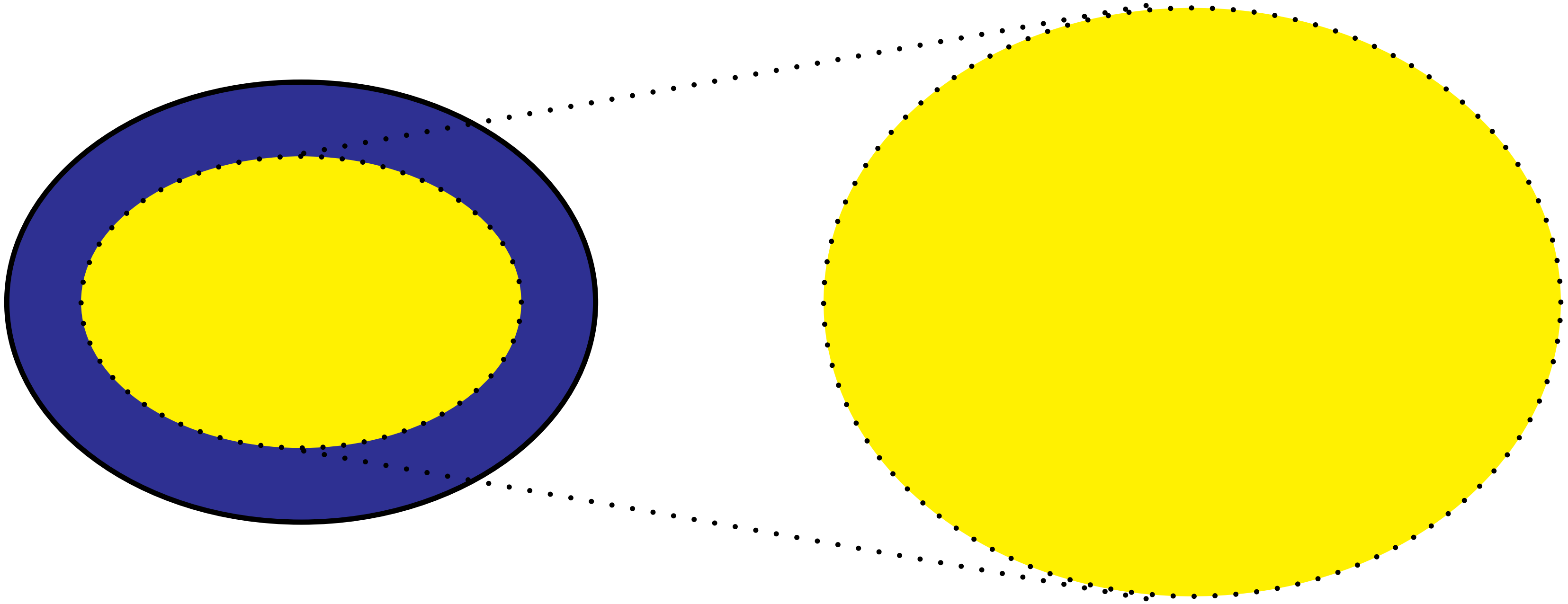


Standard Models $\mathfrak{ST}(\Sigma)$

Formulas valid in $\mathfrak{ST}(\Sigma)$

choose: $\mathcal{D}_\iota$
fixed: $\mathcal{D}_o, \mathcal{D}_{\alpha\to\beta}$, functions

Model Classes (Extensionality)

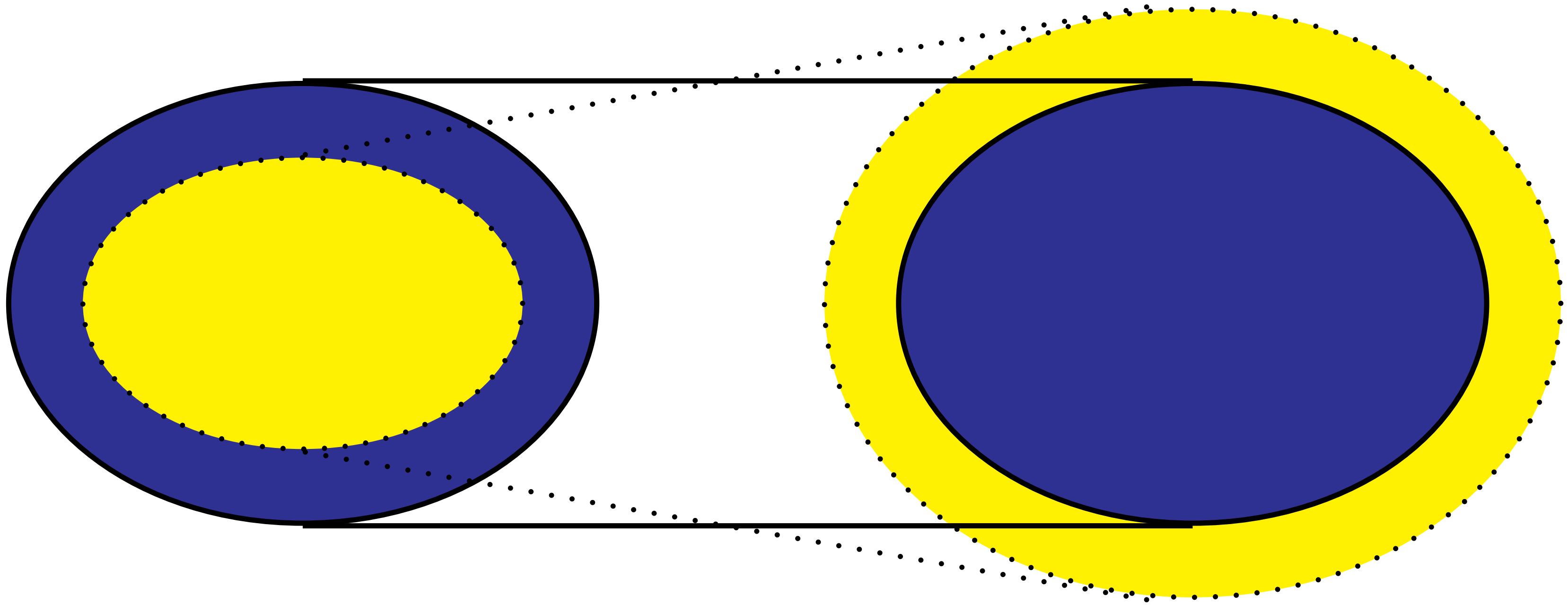


Henkin Models $\mathfrak{H}(\Sigma) = \mathfrak{M}_{\beta\mathfrak{f}\mathfrak{b}}(\Sigma)$

Formulas valid in $\mathfrak{M}_{\beta\mathfrak{f}\mathfrak{b}}(\Sigma)$?

choose: $\mathcal{D}_\iota, \mathcal{D}_{\alpha\to\beta}$
fixed: $\mathcal{D}_o$, functions

Model Classes (Extensionality)

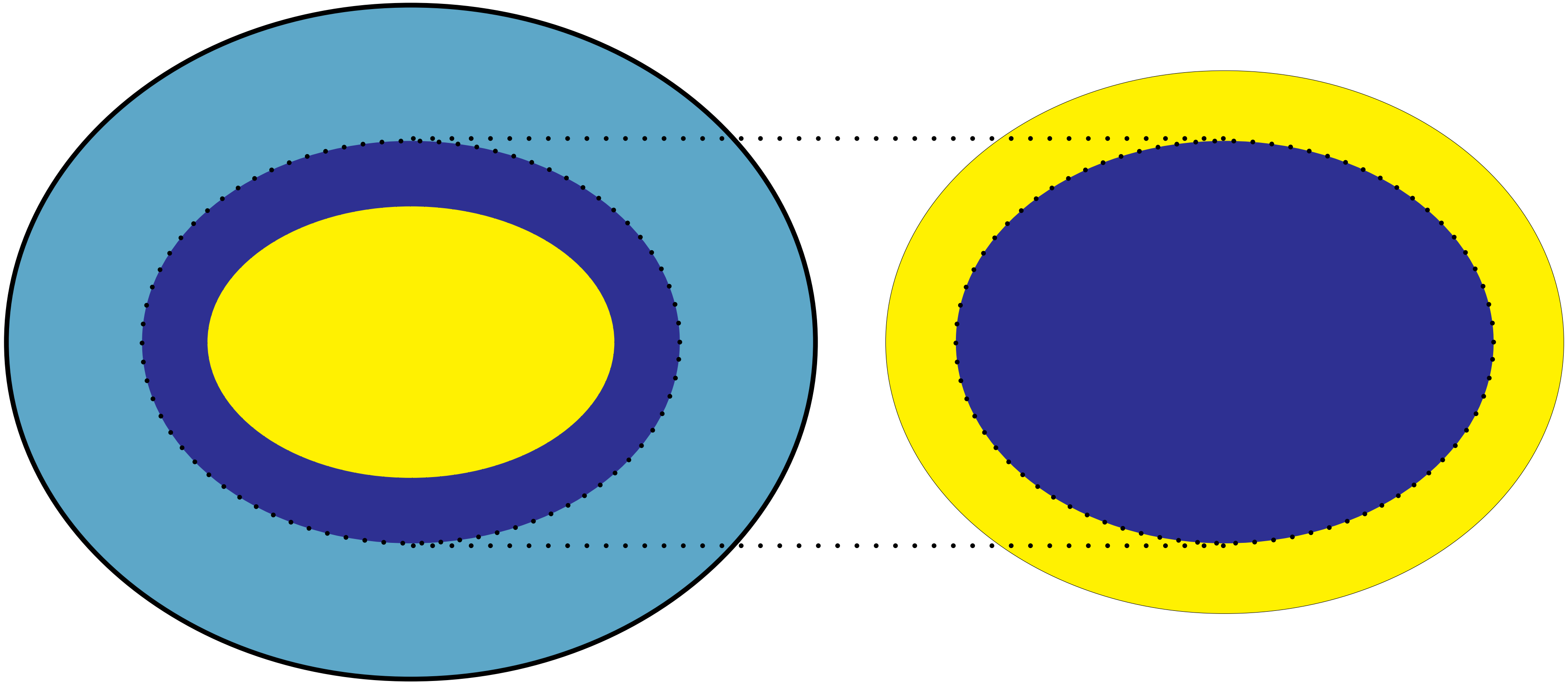


Henkin Models $\mathfrak{H}(\Sigma) = \mathfrak{M}_{\beta\mathfrak{f}\mathfrak{b}}(\Sigma)$

Formulas valid in $\mathfrak{M}_{\beta\mathfrak{f}\mathfrak{b}}(\Sigma)$

choose: $\mathcal{D}_\iota, \mathcal{D}_{\alpha\rightarrow\beta}$
fixed: $\mathcal{D}_o$, functions

Model Classes (Extensionality)



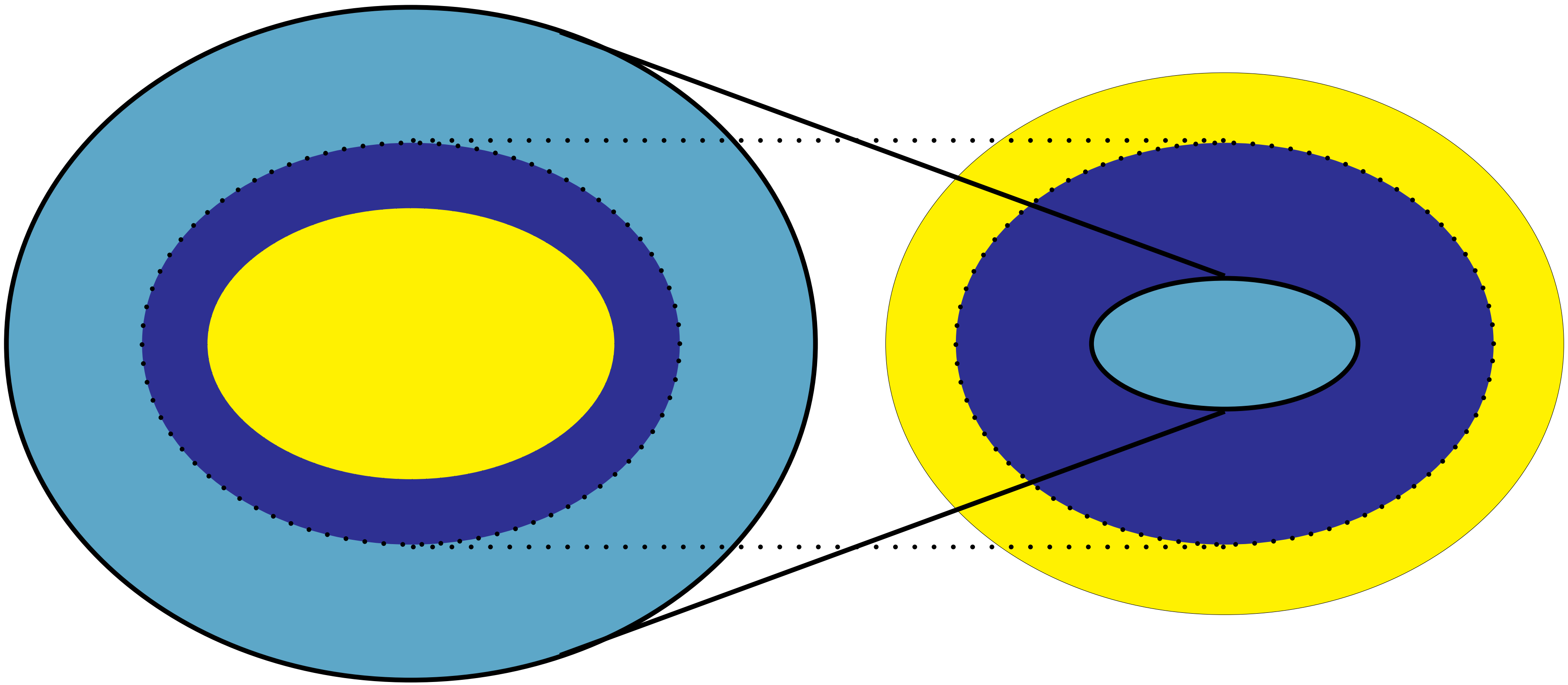
Non-Extensional Models $\mathfrak{M}_\beta(\Sigma)$

Formulas valid in $\mathfrak{M}_\beta(\Sigma)$?

choose: $\mathcal{D}_\iota, \mathcal{D}_{\alpha\rightarrow\beta}$, also non$-$functions, $\mathcal{D}_o$

fixed:

Model Classes (Extensionality)



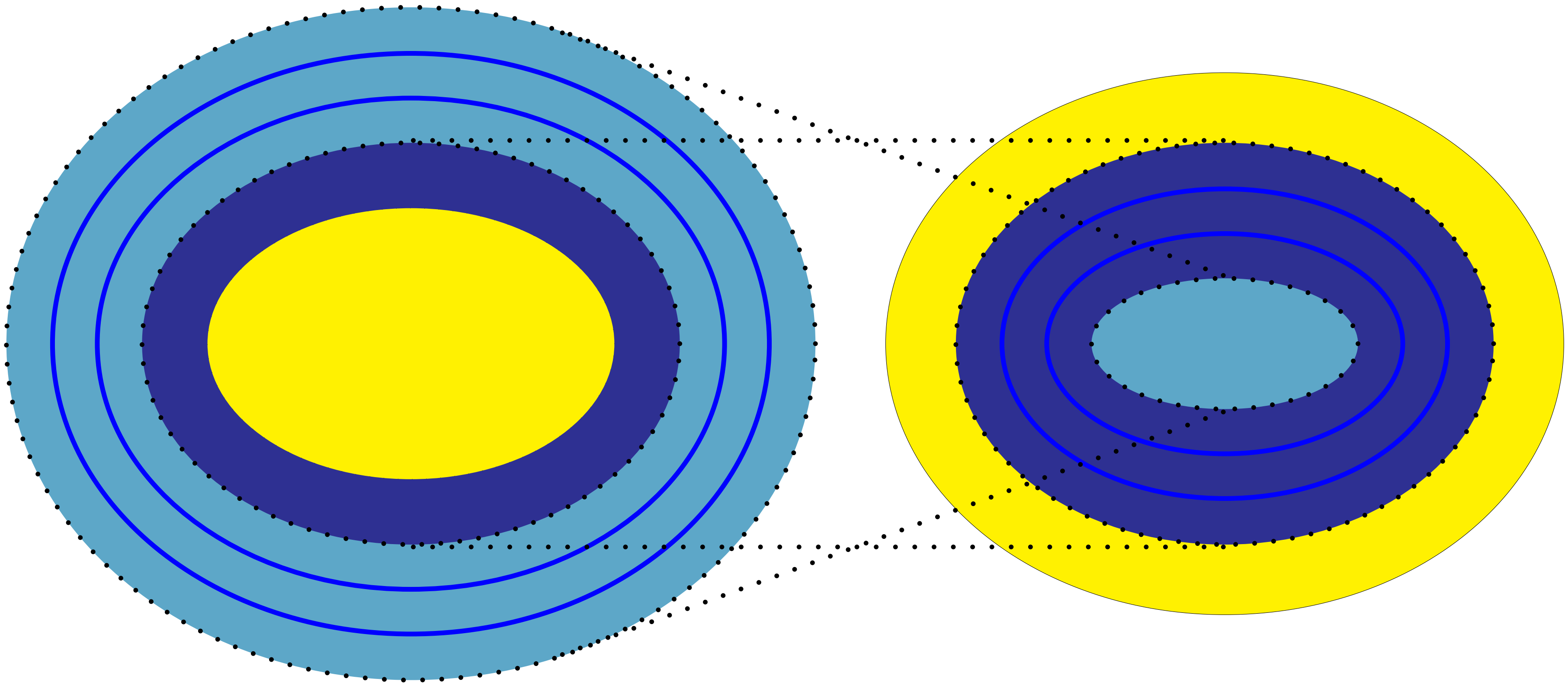
Non-Extensional Models $\mathfrak{M}_\beta(\Sigma)$

Formulas valid in $\mathfrak{M}_\beta(\Sigma)$?

choose: $\mathcal{D}_\iota, \mathcal{D}_{\alpha\to\beta}$, also non$-$functions, $\mathcal{D}_o$

fixed:

Model Classes (Extensionality)



We additionally studied different model classes with 'varying degrees of extensionality'

Model Classes (Extensionality)

$\mathfrak{M}_\beta(\Sigma)$ — non-extensional models

$\mathfrak{b}$: Boolean extensionality, $\mathcal{D}_o = \{\mathtt{T}, \mathtt{F}\}$

$\mathfrak{f}(= \eta + \xi)$: functional extensionality

η: η-functional

ξ: ξ-functionality

$\mathfrak{M}_{\beta\mathfrak{f}\mathfrak{b}}(\Sigma) \simeq \mathfrak{H}(\Sigma)$ — Henkin models

full

$\mathfrak{ST}(\Sigma)$ — standard models

Model Classes (Extensionality)

$$\begin{tikzcd}
\mathfrak{M}_{\beta}(\Sigma) \arrow[dr, "\mathfrak{b}"] \arrow[dd, dotted, "{\mathfrak{b}, \mathfrak{f}(= \eta + \xi)}"'] & \\
& \mathfrak{M}_{\beta\mathfrak{b}}(\Sigma) \arrow[dl, "\mathfrak{f}"] \\
\mathfrak{M}_{\beta\mathfrak{f}\mathfrak{b}}(\Sigma) \simeq \mathfrak{H}(\Sigma) \arrow[d, dotted, "\text{full}"] & \\
\mathfrak{S}\mathfrak{T}(\Sigma) &
\end{tikzcd}$$

non-extensional models

Henkin models

standard models

Model Classes (Extensionality)

$$
\begin{tikzcd}
\mathfrak{M}_{\beta}(\Sigma) \arrow[dr, "\mathfrak{b}"] \arrow[dd, dotted, "{\mathfrak{b}, \mathfrak{f}(= \eta + \xi)}"'] & \\
& \mathfrak{M}_{\beta\mathfrak{b}}(\Sigma) \arrow[dl, "\mathfrak{f}"] \\
\mathfrak{M}_{\beta\mathfrak{f}\mathfrak{b}}(\Sigma) \simeq \mathfrak{H}(\Sigma) \arrow[d, dotted, "\text{full}"] & \\
\mathfrak{S}\mathfrak{T}(\Sigma) &
\end{tikzcd}
$$

non-extensional models

Henkin models

standard models

Model Classes (Extensionality)

$$
\begin{tikzcd}
 & \mathfrak{M}_{\beta}(\Sigma) \arrow[dl, "\mathfrak{f}"'] \arrow[dr, "\mathfrak{b}"] \arrow[dd, dotted, "{\mathfrak{b}, \mathfrak{f}(= \eta + \xi)}" description] & & \text{non-extensional models} \\
\mathfrak{M}_{\beta\mathfrak{f}}(\Sigma) \arrow[dr, "\mathfrak{b}"'] & & \mathfrak{M}_{\beta\mathfrak{b}}(\Sigma) \arrow[dl, "\mathfrak{f}"] & \\
 & \mathfrak{M}_{\beta\mathfrak{f}\mathfrak{b}}(\Sigma) \simeq \mathfrak{H}(\Sigma) \arrow[d, dotted, "\text{full}"] & & \text{Henkin models} \\
 & \mathfrak{ST}(\Sigma) & & \text{standard models}
\end{tikzcd}
$$

Model Classes (Extensionality)

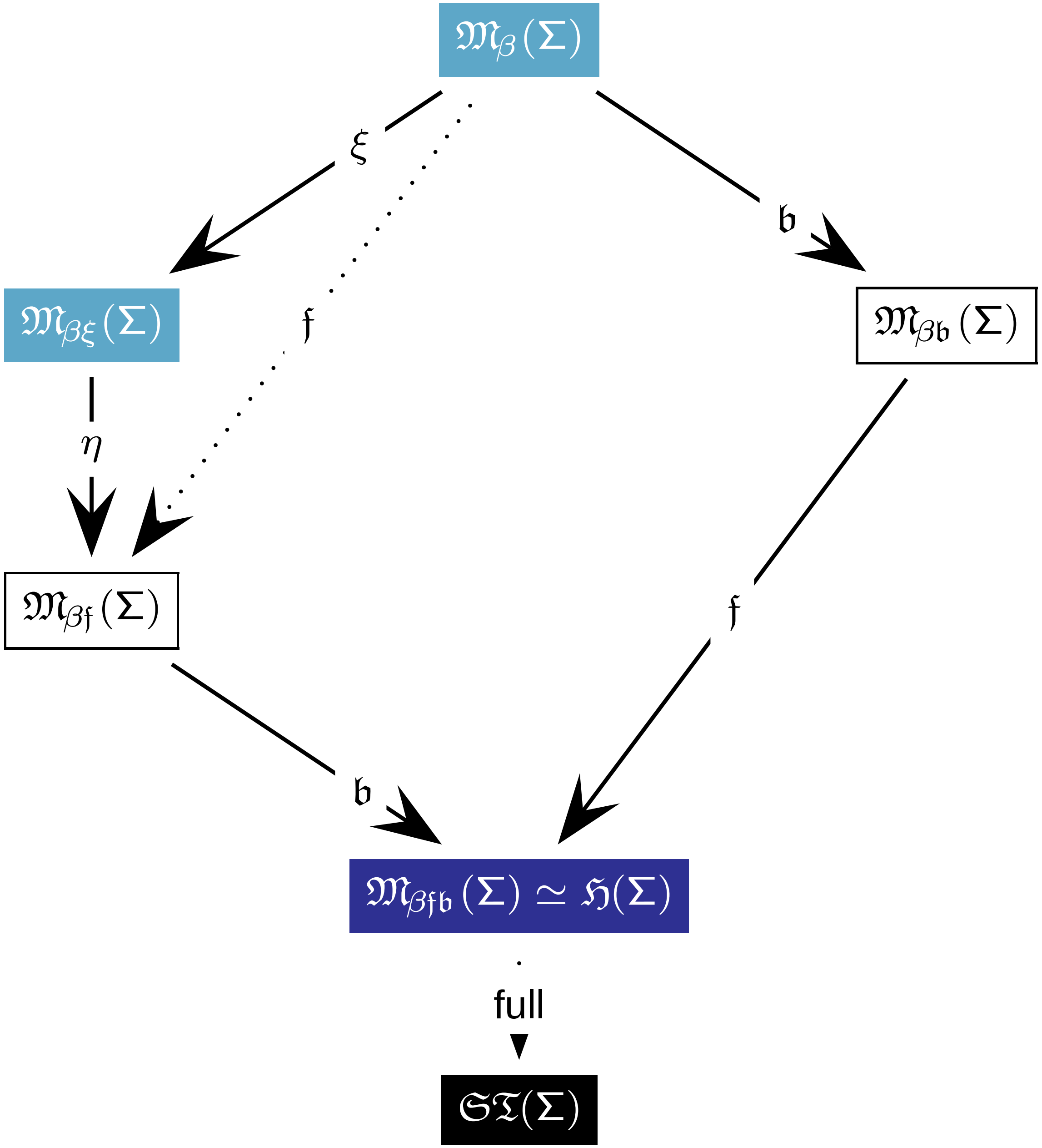
$$
\begin{tikzcd}
 & \mathfrak{M}_{\beta}(\Sigma) \arrow[dl, "\mathfrak{f}"'] \arrow[dr, "\mathfrak{b}"] \arrow[dd, dotted, "{\mathfrak{b},\mathfrak{f}(=\eta+\xi)}"] & \\
\mathfrak{M}_{\beta\mathfrak{f}}(\Sigma) \arrow[dr, "\mathfrak{b}"'] & & \mathfrak{M}_{\beta\mathfrak{b}}(\Sigma) \arrow[dl, "\mathfrak{f}"] \\
 & \mathfrak{M}_{\beta\mathfrak{f}\mathfrak{b}}(\Sigma) \simeq \mathfrak{H}(\Sigma) \arrow[d, dotted, "\text{full}"] & \\
 & \mathfrak{S}\mathfrak{T}(\Sigma) &
\end{tikzcd}
$$

non-extensional models

Henkin models

standard models

Model Classes (Extensionality)



non-extensional models

Henkin models

standard models

Model Classes (Extensionality)

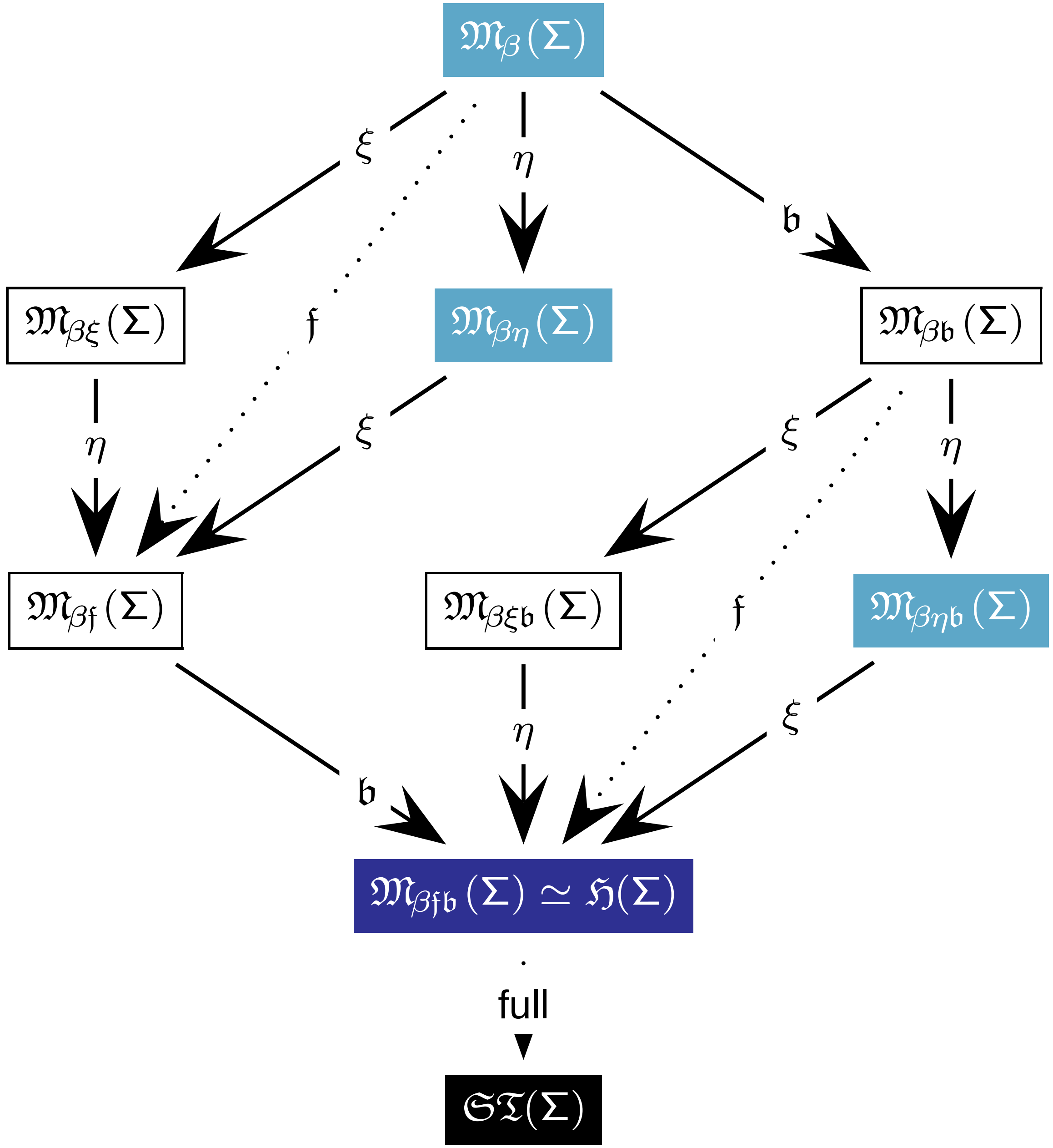
$$
\begin{tikzcd}
 & \mathfrak{M}_{\beta}(\Sigma) \arrow[dl, "\xi"'] \arrow[ddl, dotted, "\mathfrak{f}"] \arrow[dr, "\mathfrak{b}"] & \\
\mathfrak{M}_{\beta\xi}(\Sigma) \arrow[d, "\eta"] & & \mathfrak{M}_{\beta\mathfrak{b}}(\Sigma) \arrow[dl, "\xi"'] \arrow[ddl, dotted, "\mathfrak{f}"] \\
\mathfrak{M}_{\beta\mathfrak{f}}(\Sigma) \arrow[dr, "\mathfrak{b}"] & \mathfrak{M}_{\beta\xi\mathfrak{b}}(\Sigma) \arrow[d, "\eta"] & \\
 & \mathfrak{M}_{\beta\mathfrak{f}\mathfrak{b}}(\Sigma) \simeq \mathfrak{H}(\Sigma) \arrow[d, dotted, "\text{full}"] & \\
 & \mathfrak{ST}(\Sigma) &
\end{tikzcd}
$$

non-extensional models

Henkin models

standard models

Model Classes (Extensionality)

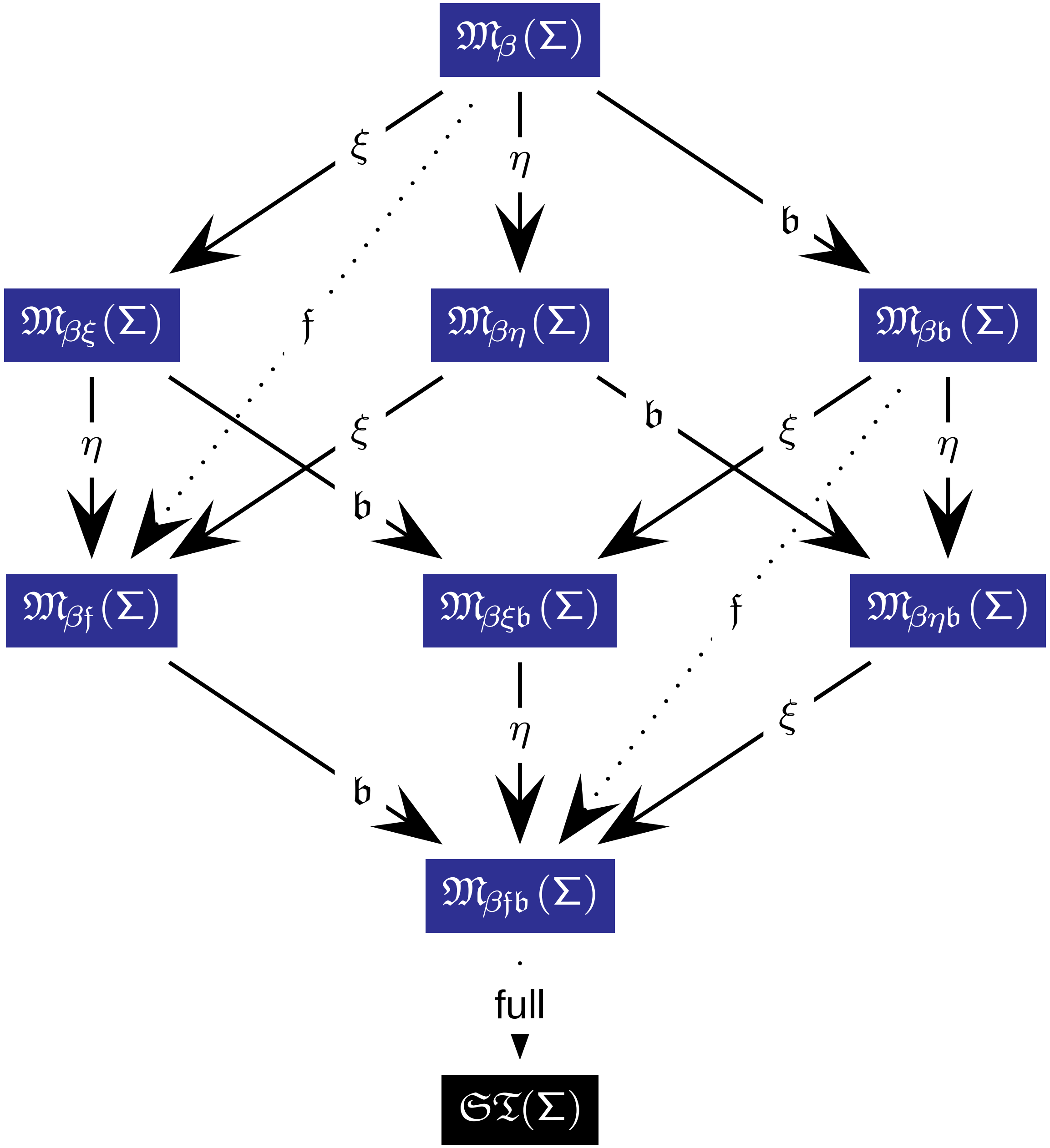


non-extensional models

Henkin models

standard models

Model Classes (Extensionality)



non-extensional models

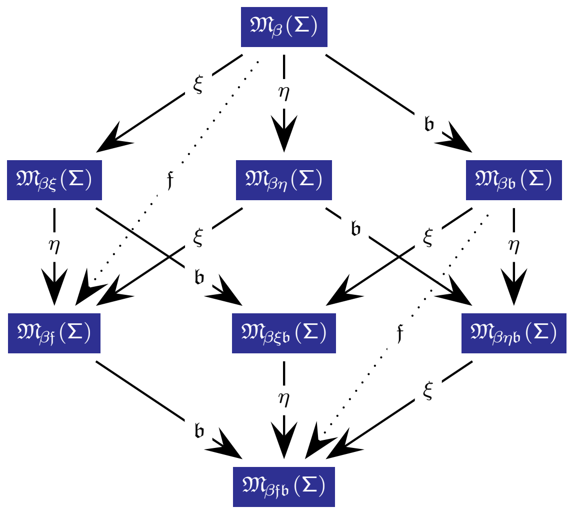
Henkin models

standard models

Semantics - Calculi - Abstract Consistency

Semantics:

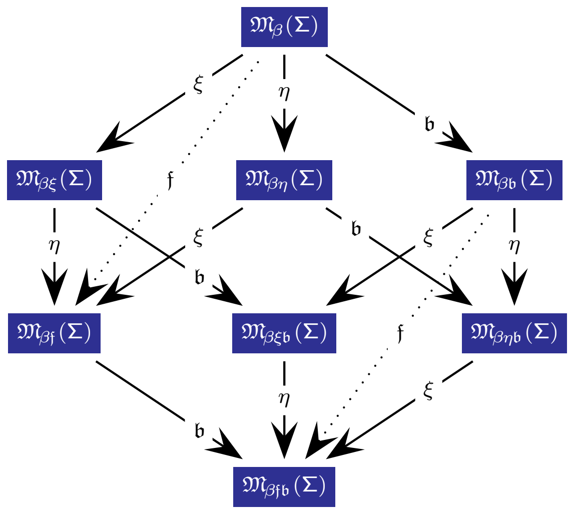
Model Classes (Extensionality)



Semantics - Calculi - Abstract Consistency

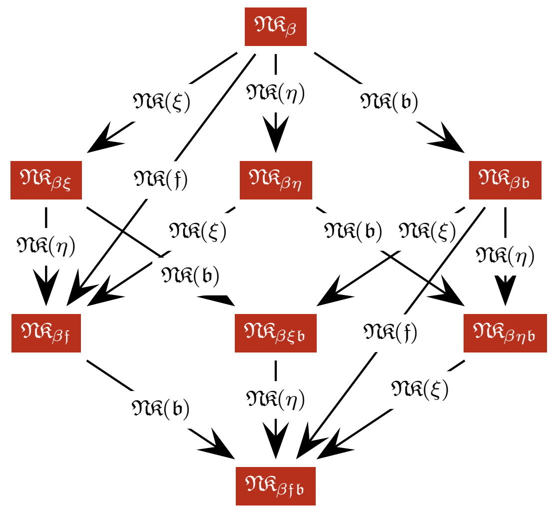
Semantics:

Model Classes (Extensionality)



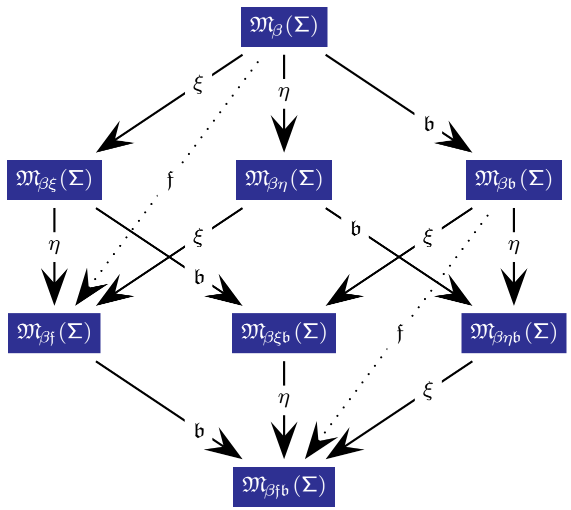
Reference Calculi:

ND (and others)

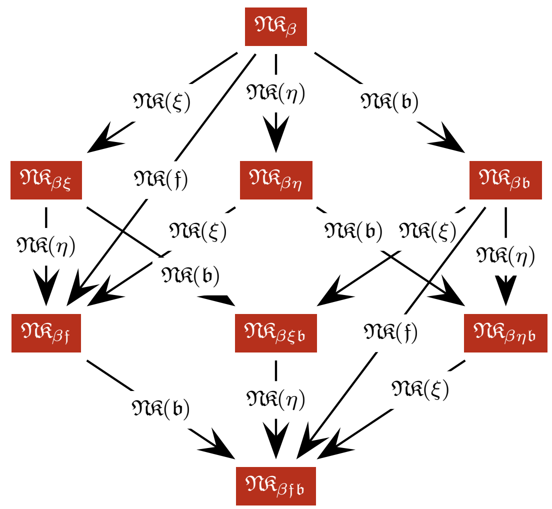


Semantics - Calculi - Abstract Consistency

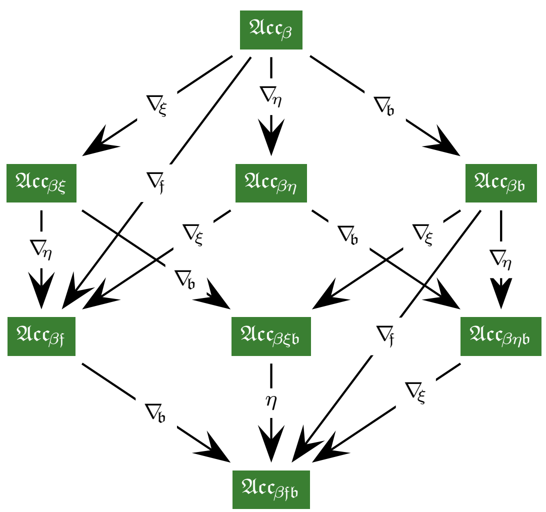
Semantics:
Model Classes (Extensionality)



Reference Calculi:
ND (and others)



Abstract Consistency / Unifying Principle:
Extensions of Smullyan-63 and Andrews-71



Extensional Resolution

Extensional HO Resolution $\mathcal{ER}$

- [Andrews-71] Higher-order resolution (without unification)

ext. axioms	proof search & blind variable instantiation

- [Huet-73/75] Higher-order constrained resolution

ext. axioms	proof search & eager unification	delayed pre-unification

- [Benzmüller-99] Extensional higher-order resolution

interleaved proof search & unification

Cut-simulation

Sequent Calculi for HOL

We work with a one-sided sequent calculus:

Sequent Calculi for HOL

We work with a one-sided sequent calculus:

- examples for two-sided rules:

Sequent Calculi for HOL

We work with a one-sided sequent calculus:

- examples for two-sided rules:

$$\frac{}{\Gamma \Longrightarrow \Delta, \mathbf{A} \vee \mathbf{B}}\ \mathcal{G}(\vee_{Intro})$$

$\Delta,\mathbf{C}$ stands for $\Delta \cup \{\mathbf{C}\}$

Sequent Calculi for HOL

We work with a one-sided sequent calculus:

- examples for two-sided rules:

$$\frac{\Gamma \Longrightarrow \mathbf{\Delta},\mathbf{A},\mathbf{B}}{\Gamma \Longrightarrow \mathbf{\Delta},\mathbf{A} \lor \mathbf{B}}\, \mathcal{G}(\lor_{Intro})$$

$\mathbf{\Delta},\mathbf{C}$ stands for $\mathbf{\Delta} \cup \{\mathbf{C}\}$

Sequent Calculi for HOL

We work with a one-sided sequent calculus:

- examples for two-sided rules:

$$\frac{\Gamma \Longrightarrow \mathbf{\Delta},\mathbf{A},\mathbf{B}}{\Gamma \Longrightarrow \mathbf{\Delta},\mathbf{A} \vee \mathbf{B}}\,\mathcal{G}(\vee_{Intro})$$

- corresponding one-sided rules:

$$\frac{}{\neg(\Gamma) \cup \mathbf{\Delta},\mathbf{A} \vee \mathbf{B}}\,\mathcal{G}(\vee_{+})$$

$\mathbf{\Delta},\mathbf{C}$ stands for $\mathbf{\Delta} \cup \{\mathbf{C}\}$

Sequent Calculi for HOL

We work with a one-sided sequent calculus:

- examples for two-sided rules:

$$\frac{\Gamma \Longrightarrow \mathbf{\Delta},\mathbf{A},\mathbf{B}}{\Gamma \Longrightarrow \mathbf{\Delta},\mathbf{A} \vee \mathbf{B}}\,\mathcal{G}(\vee_{Intro})$$

- corresponding one-sided rules:

$$\frac{\neg(\Gamma) \cup \mathbf{\Delta},\mathbf{A},\mathbf{B}}{\neg(\Gamma) \cup \mathbf{\Delta},\mathbf{A} \vee \mathbf{B}}\,\mathcal{G}(\vee_{+})$$

$\mathbf{\Delta},\mathbf{C}$ stands for $\mathbf{\Delta} \cup \{\mathbf{C}\}$

Sequent Calculi for HOL

We work with a one-sided sequent calculus:

- examples for two-sided rules:

$$\frac{\Gamma \Longrightarrow \mathbf{\Delta},\mathbf{A},\mathbf{B}}{\Gamma \Longrightarrow \mathbf{\Delta},\mathbf{A} \vee \mathbf{B}}\,\mathcal{G}(\vee_{Intro}) \qquad \frac{}{\Gamma,\mathbf{A} \vee \mathbf{B} \Longrightarrow \mathbf{\Delta}}\,\mathcal{G}(\vee_{Elim})$$

- corresponding one-sided rules:

$$\frac{\neg(\Gamma) \cup \mathbf{\Delta},\mathbf{A},\mathbf{B}}{\neg(\Gamma) \cup \mathbf{\Delta},\mathbf{A} \vee \mathbf{B}}\,\mathcal{G}(\vee_{+})$$

$\mathbf{\Delta},\mathbf{C}$ stands for $\mathbf{\Delta} \cup \{\mathbf{C}\}$

Sequent Calculi for HOL

We work with a one-sided sequent calculus:

- examples for two-sided rules:

$$\frac{\Gamma \Longrightarrow \Delta, \mathbf{A}, \mathbf{B}}{\Gamma \Longrightarrow \Delta, \mathbf{A} \vee \mathbf{B}} \mathcal{G}(\vee_{Intro}) \qquad \frac{\Gamma, \mathbf{A} \Longrightarrow \Delta \quad \Gamma, \mathbf{B} \Longrightarrow \Delta}{\Gamma, \mathbf{A} \vee \mathbf{B} \Longrightarrow \Delta} \mathcal{G}(\vee_{Elim})$$

- corresponding one-sided rules:

$$\frac{\neg(\Gamma) \cup \Delta, \mathbf{A}, \mathbf{B}}{\neg(\Gamma) \cup \Delta, \mathbf{A} \vee \mathbf{B}} \mathcal{G}(\vee_{+})$$

$\Delta, \mathbf{C}$ stands for $\Delta \cup \{\mathbf{C}\}$

Sequent Calculi for HOL

We work with a one-sided sequent calculus:

- examples for two-sided rules:

$$\frac{\Gamma \Longrightarrow \Delta,\mathbf{A},\mathbf{B}}{\Gamma \Longrightarrow \Delta,\mathbf{A} \lor \mathbf{B}}\,\mathcal{G}(\lor_{Intro}) \qquad \frac{\Gamma,\mathbf{A} \Longrightarrow \Delta \quad \Gamma,\mathbf{B} \Longrightarrow \Delta}{\Gamma,\mathbf{A} \lor \mathbf{B} \Longrightarrow \Delta}\,\mathcal{G}(\lor_{Elim})$$

- corresponding one-sided rules:

$$\frac{\neg(\Gamma) \cup \Delta,\mathbf{A},\mathbf{B}}{\neg(\Gamma) \cup \Delta,\mathbf{A} \lor \mathbf{B}}\,\mathcal{G}(\lor_{+}) \qquad \frac{}{\neg(\Gamma) \cup \Delta,\neg(\mathbf{A} \lor \mathbf{B})}\,\mathcal{G}(\lor_{-})$$

$\Delta,\mathbf{C}$ stands for $\Delta \cup \{\mathbf{C}\}$

Sequent Calculi for HOL

We work with a one-sided sequent calculus:

- examples for two-sided rules:

$$\frac{\Gamma \Longrightarrow \Delta,\mathbf{A},\mathbf{B}}{\Gamma \Longrightarrow \Delta,\mathbf{A} \vee \mathbf{B}}\,\mathcal{G}(\vee_{Intro}) \qquad \frac{\Gamma,\mathbf{A} \Longrightarrow \Delta \quad \Gamma,\mathbf{B} \Longrightarrow \Delta}{\Gamma,\mathbf{A} \vee \mathbf{B} \Longrightarrow \Delta}\,\mathcal{G}(\vee_{Elim})$$

- corresponding one-sided rules:

$$\frac{\neg(\Gamma) \cup \Delta,\mathbf{A},\mathbf{B}}{\neg(\Gamma) \cup \Delta,\mathbf{A} \vee \mathbf{B}}\,\mathcal{G}(\vee_{+}) \qquad \frac{\neg(\Gamma) \cup \Delta,\neg\mathbf{A} \quad \neg(\Gamma) \cup \Delta,\neg\mathbf{B}}{\neg(\Gamma) \cup \Delta,\neg(\mathbf{A} \vee \mathbf{B})}\,\mathcal{G}(\vee_{-})$$

$\Delta,\mathbf{C}$ stands for $\Delta \cup \{\mathbf{C}\}$

Sequent Calculi for HOL

$$\frac{\mathbf{A} \text{ atomic (and } \beta\text{-normal)}}{\Delta, \neg\mathbf{A}, \mathbf{A}}\ \mathcal{G}(init) \qquad \frac{\Delta, \mathbf{A}}{\Delta, \neg\neg\mathbf{A}}\ \mathcal{G}(\neg)$$

$$\frac{\Delta, \neg\mathbf{A} \quad \Delta, \neg\mathbf{B}}{\Delta, \neg(\mathbf{A} \vee \mathbf{B})}\ \mathcal{G}(\vee_-) \qquad \frac{\Delta, \mathbf{A}, \mathbf{B}}{\Delta, (\mathbf{A} \vee \mathbf{B})}\ \mathcal{G}(\vee_+)$$

$$\frac{\Delta, \neg(\mathbf{A}\mathbf{C}){\downarrow_\beta} \quad \mathbf{C} \in cwff_\alpha(\Sigma)}{\Delta, \neg\forall X_\alpha \mathbf{A}}\ \mathcal{G}(\Pi_-^{\mathbf{C}}) \qquad \frac{\Delta, (\mathbf{A}c){\downarrow_\beta} \quad c_\alpha \in \Sigma \text{ new}}{\Delta, \forall X_\alpha \mathbf{A}}\ \mathcal{G}(\Pi_+^{c})$$

$\Delta, \mathbf{A}$ stands for $\Delta \cup \{\mathbf{A}\}$

Sequent Calculi for HOL: $\mathcal{G}_\beta$

The sequent calculus $\mathcal{G}_\beta$ is defined by the rules

$$\mathcal{G}(init), \mathcal{G}(\neg), \mathcal{G}(\vee_-), \mathcal{G}(\vee_+), \mathcal{G}(\Pi_-^{\mathbf{C}}), \mathcal{G}(\Pi_+^c)$$

- is sound for the eight model classes $\mathfrak{M}_*$
- is complete for the model class $\mathfrak{M}_\beta(\Sigma)$

- suitable for automation? $\longrightarrow$ Analysis of admissibility of cut:

$$\frac{\Delta, \mathbf{C} \quad \Delta, \neg\mathbf{C}}{\Delta}\,\mathcal{G}(cut)$$

- $\mathcal{G}_\beta$ is indeed cut-free

Cut-simulation with Leibnizequations

Leibniz-equations $\mathbf{M} \doteq^{\alpha} \mathbf{N}$ $(:= \forall \mathrm{P}_{o\alpha} \blacksquare \neg \mathrm{P}\mathbf{M} \lor \mathrm{P}\mathbf{N})$ support cut-simulation in $\mathcal{G}_{\beta}$ in only 3 steps.

Proof:

$$\cfrac{\cfrac{\cfrac{\Delta, \mathbf{C}}{\Delta, \neg\neg\mathbf{C}}\ \mathcal{G}(\neg) \qquad \Delta, \neg\mathbf{C}}{\Delta, \neg(\neg\mathbf{C} \lor \mathbf{C})}\ \mathcal{G}(\lor_{-})}{\Delta' := \Delta, \neg\forall \mathrm{P}_{o\alpha} \blacksquare \neg \mathrm{P}\mathbf{M} \lor \mathrm{P}\mathbf{N}}\ \mathcal{G}(\Pi_{-}^{\lambda X_{\alpha} \blacksquare \mathbf{C}})$$

Cut-simulation with Extensionality Axioms

The Boolean extensionality axiom $\mathcal{B}_o$ is:

$$\forall A_o . \forall B_o . (A \Leftrightarrow B) \Rightarrow A \doteq^o B$$

The infinitely many functional extensionality axioms $\mathcal{F}_{\alpha\beta}$ are:

$$\forall F_{\alpha\rightarrow\beta} . \forall G_{\alpha\rightarrow\beta} . (\forall X_\alpha . FX \doteq^\beta GX) \Rightarrow F \doteq^{\alpha\rightarrow\beta} G$$

Cut-simulation with Extensionality Axioms

The functional extensionality axioms support effective cut-simulation in $\mathcal{G}_\beta$ in 11-steps.

Proof:

$$
\cfrac{
\cfrac{
\cfrac{
\begin{array}{c}\text{3 steps; easy}\\ \vdots \\ \Delta, \mathsf{fa} \doteq^{\beta} \mathsf{fa}\end{array}
}{
\Delta, (\forall X_\alpha . \mathsf{f}X \doteq^{\beta} \mathsf{f}X)
}\ \mathcal{G}(\Pi_{+}^{a_\alpha})
}{
\Delta, \neg\neg\forall X_\alpha . \mathsf{f}X \doteq^{\beta} \mathsf{f}X
}\ \mathcal{G}(\neg)
\qquad
\begin{array}{c}\Delta, \mathbf{C} \quad \Delta, \neg\mathbf{C}\\ \vdots\ \text{3 steps; see before}\\ \Delta, \neg(\mathsf{f} \doteq^{\alpha\to\beta} \mathsf{f})\end{array}
}{
\cfrac{
\Delta, \neg(\neg(\forall X_\alpha . \mathsf{f}X \doteq^{\beta} \mathsf{f}X) \vee \mathsf{f} \doteq^{\alpha\to\beta} \mathsf{f})
}{
\Delta, \neg\mathcal{F}_{\alpha\beta}
}\ 2 \times \mathcal{G}(\Pi_{-}^{f})
}\ \mathcal{G}(\vee_{-})
$$

Cut-simulation with Extensionality Axioms

It also works with Boolean extensionality axiom – in 14 steps.

Proof:

$$
\dfrac{\dfrac{\dfrac{\begin{array}{c}\text{7 steps; easy}\\ \vdots \\ \Delta, a \Leftrightarrow a\end{array}}{\Delta,\neg\neg(a\Leftrightarrow a)}\;\mathcal{G}(\neg) \qquad \begin{array}{c}\Delta,\mathbf{C}\quad\Delta,\neg\mathbf{C}\\ \vdots\;\text{3 steps; see before}\\ \Delta,\neg(a \doteq^{o} a)\end{array}}{\Delta,\neg(\neg(a\Leftrightarrow a)\vee a\doteq^{o} a)}\;\mathcal{G}(\vee_{-})}{\Delta,\neg\mathcal{B}_{o}}\;2\times\mathcal{G}(\Pi^{a}_{-})
$$

Cut-simulation with other Axioms

- Reflexivity definition of equality (Andrews) 4 steps

$$\lambda X_\alpha . \lambda Y_\alpha . \forall Q_{\alpha\to\alpha\to o} . (\forall Z_\alpha . (Q\ Z\ Z)) \Rightarrow (Q\ X\ Y)$$

Cut-simulation with other Axioms

- Reflexivity definition of equality (Andrews) 4 steps
- Instances of Comprehension axioms 16 steps

$$\exists P_{\iota \rightarrow o} \centerdot \forall X_{\iota} \centerdot P X \Leftrightarrow X \doteq^{\iota} X$$

Cut-simulation with other Axioms

- Reflexivity definition of equality (Andrews) 4 steps
- Instances of Comprehension axioms 16 steps
- Axiom of Induction 18 steps

$$\forall P_{\iota\to o}.P0 \wedge (\forall X_\iota.PX \Rightarrow P(sX)) \Rightarrow \forall X_\iota.PX$$

Cut-simulation with other Axioms

- Reflexivity definition of equality (Andrews) 4 steps
- Instances of Comprehension axioms 16 steps
- Axiom of Induction 18 steps
- Axiom of Choice 7 steps

$$\exists I_{(\alpha\to o)\to o}.\forall Q_{\alpha\to o}.\exists X_{\alpha}.QX \Rightarrow Q(IQ)$$

Cut-simulation with other Axioms

- Reflexivity definition of equality (Andrews) 4 steps
- Instances of Comprehension axioms 16 steps
- Axiom of Induction 18 steps
- Axiom of Choice 7 steps
- Axiom of Description 25 steps

$$\exists I_{(\alpha\to o)\to o}.\forall Q_{\alpha\to o}.(\exists_1 Y_\alpha.QY) \Rightarrow Q(IQ)$$

Cut-simulation with other Axioms

- Reflexivity definition of equality (Andrews) 4 steps
- Instances of Comprehension axioms 16 steps
- Axiom of Induction 18 steps
- Axiom of Choice 7 steps
- Axiom of Description 25 steps
- Axiom of Excluded Middle 3 steps

$$\forall Q_o . Q \vee \neg Q$$

Cut-simulation with other Axioms

Reflexivity definition of equality (Andrews)	4 steps
Instances of Comprehension axioms	16 steps
Axiom of Induction	18 steps
Axiom of Choice	7 steps
Axiom of Description	25 steps
Axiom of Excluded Middle	3 steps
...	

Cut-simulation with other Axioms

- Reflexivity definition of equality (Andrews) — 4 steps
- Instances of Comprehension axioms — 16 steps
- Axiom of Induction — 18 steps
- Axiom of Choice — 7 steps
- Axiom of Description — 25 steps
- Axiom of Excluded Middle — 3 steps
- ...

This motivates lots of further research on HOL automation:

How to avoid / treat cut-strong axioms and formulas!

Conclusion

($\geq$) Two hearts are beating in my chest:

- Integrated and intelligent mathematics assistance systems
  - ΩMEGA at forefront of systems under this vision
  - several of our ideas meanwhile picked up by others
  - several cooperations
- Foundations and automation (not only!) of HOL
  - contributions to: semantics $\leftrightarrow$ proof theory $\leftrightarrow$ automation
  - automation still decades behind

Currently I am

- working towards a HOL prover competition
- implementing LEO-II, a new version of the prover LEO
- planning to integrate LEO-II with Isabelle/HOL & OMEGA

HOL Challenge: Impredicativity

Notion: (Impredicativity)

- quantification over sets and predicates
- support impredicative definitions and reflection

HOL Challenge: Impredicativity

Notion: (Impredicativity)

- quantification over sets and predicates
- support impredicative definitions and reflection

Ex.: Automation already problematic for very simple quantifications over sets: $\exists P\,(P\ 1)$

HOL Challenge: Impredicativity

Notion: (Impredicativity)

- quantification over sets and predicates
- support impredicative definitions and reflection

Ex.: Automation already problematic for very simple quantifications over sets: $\exists P\,(P\ 1)$

- $P \longleftarrow \{x|\mathsf{true}\}$ $\qquad (\lambda X\ \mathbf{T}_o)$
- $P \longleftarrow \{x|x = 1\}$ $\qquad (\lambda X\ X = 1)$
- $P \longleftarrow \{x|x = 1 \vee x = 2\}$ $\qquad (\lambda X\ X = 1 \vee X = 2)$
- $P \longleftarrow \{x|x > 0\}$ $\qquad (\lambda X\ X > 0)$
- etc.

HOL Challenge: Impredicativity

Notion: (Impredicativity)

- quantification over sets and predicates
- support impredicative definitions and reflection

Ex.: Automation already problematic for very simple quantifications over sets: $\exists P\,(P\ 1)$

- $P \longleftarrow \{x|\mathsf{true}\}$ $(\lambda X\ \mathbf{T}_o)$
- $P \longleftarrow \{x|x = 1\}$ $(\lambda X\ X = 1)$
- $P \longleftarrow \{x|x = 1 \vee x = 2\}$ $(\lambda X\ X = 1 \vee X = 2)$
- $P \longleftarrow \{x|x > 0\}$ $(\lambda X\ X > 0)$
- etc.

- unification not powerful enough $\Longrightarrow$ guessing is state of the art
- problem not limited to HOL

Extensional Resolution

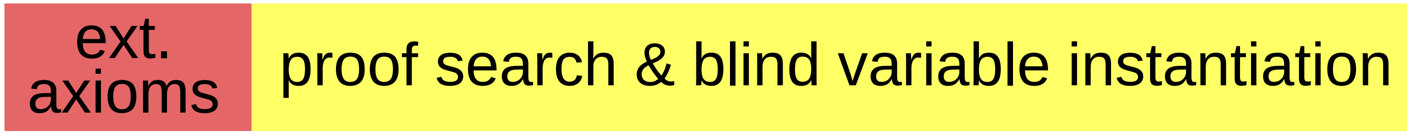
Extensional HO Resolution $\mathcal{ER}$

- [Andrews-71]

ext. axioms	proof search & blind variable instantiation

Extensional HO Resolution $\mathcal{ER}$

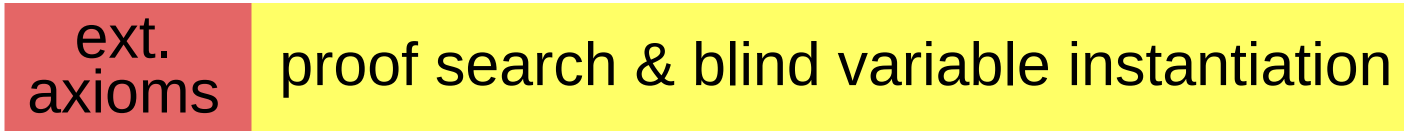
- [Andrews-71]



- [Huet-73/75]

Extensional HO Resolution $\mathcal{ER}$

- [Andrews-71]



- [Huet-73/75]

- [Benzmüller-99]

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

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

Negation and definition expansion with

$\cup = \lambda A_{\alpha\to o}, B_{\alpha\to o}, X_{\alpha} \centerdot (A\ X) \vee (B\ X) \qquad \cap = \lambda A_{\alpha\to o}, B_{\alpha\to o}, X_{\alpha} \centerdot (A\ X) \wedge (B\ X)$

leads to:

$$C_1 : [\lambda X_{\alpha} \centerdot (b\ X) \vee ((c\ X) \wedge (d\ X)) \neq^{?} \lambda X_{\alpha} \centerdot ((b\ X) \vee (c\ X)) \wedge ((b\ X) \vee (d\ X))]$$

Goal directed functional and Boolean extensionality treatment:

$$C_2 : [(b\ x) \vee ((c\ x) \wedge (d\ x)) \Leftrightarrow ((b\ x) \vee (c\ x)) \wedge ((b\ x) \vee (d\ x))]^{F}$$

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:

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

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

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

Negation and clause normalization

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

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

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

Decomposition

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

Goal directed extensionality treatment and clause normalisation:

- from $\mathcal{C}_6$ $\qquad \mathcal{C}_8 : [a]^F \vee [b]^F \qquad \mathcal{C}_9 : [a]^T \vee [b]^T \qquad \mathcal{C}_{10} : [a]^T$
- from $\mathcal{C}_7$ $\qquad \mathcal{C}_{11} : [a]^F \vee [b]^F \qquad \mathcal{C}_{12} : [a]^T \vee [b]^T \qquad \mathcal{C}_{13} : [b]^T$

The Prover LEO

My theorem prover LEO

- implements extensional resolution as seen before

The Prover LEO

My theorem prover LEO

- implements extensional resolution as seen before
- employs the agent-based OANTS architecture to cooperate with state of the art first-order ATPs

The Prover LEO

My theorem prover LEO

- implements extensional resolution as seen before
- employs the agent-based OANTS architecture to cooperate with state of the art first-order ATPs
- outperforms all first-order ATPs on set examples

The Prover LEO

My theorem prover LEO

- implements extensional resolution as seen before
- employs the agent-based OANTS architecture to cooperate with state of the art first-order ATPs
- outperforms all first-order ATPs on set examples

- has gained interest (not only) from the Isabelle/HOL community

The Prover LEO

My theorem prover LEO

- implements extensional resolution as seen before
- employs the agent-based OANTS architecture to cooperate with state of the art first-order ATPs
- outperforms all first-order ATPs on set examples

- has gained interest (not only) from the Isabelle/HOL community

Biggest Challenge: Impredicativity

Notion: (Impredicativity)

- quantification over sets and predicates
- support impredicative definitions and reflection

Biggest Challenge: Impredicativity

Notion: (Impredicativity)

- quantification over sets and predicates
- support impredicative definitions and reflection

Ex.: Automation already problematic for very simple quantifications over sets: $\exists P\,(P\;1)$

Biggest Challenge: Impredicativity

Notion: (Impredicativity)

- quantification over sets and predicates
- support impredicative definitions and reflection

Ex.: Automation already problematic for very simple quantifications over sets: $\exists P\ (P\ 1)$

- $P \longleftarrow \{x|\mathrm{true}\}$ $\quad (\lambda X\ \mathbf{T}_o)$
- $P \longleftarrow \{x|x = 1\}$ $\quad (\lambda X\ X = 1)$
- $P \longleftarrow \{x|x = 1 \vee x = 2\}$ $\quad (\lambda X\ X = 1 \vee X = 2)$
- $P \longleftarrow \{x|x > 0\}$ $\quad (\lambda X\ X > 0)$
- etc.

Biggest Challenge: Impredicativity

Notion: (Impredicativity)

- quantification over sets and predicates
- support impredicative definitions and reflection

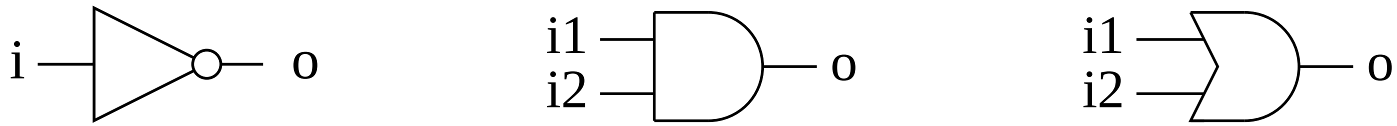
Ex.: Automation already problematic for very simple quantifications over sets: $\exists P\,(P\ 1)$

- $P \longleftarrow \{x|\mathsf{true}\}$ $\quad (\lambda X\ \mathbf{T}_o)$
- $P \longleftarrow \{x|x = 1\}$ $\quad (\lambda X\ X = 1)$
- $P \longleftarrow \{x|x = 1 \lor x = 2\}$ $\quad (\lambda X\ X = 1 \lor X = 2)$
- $P \longleftarrow \{x|x > 0\}$ $\quad (\lambda X\ X > 0)$
- etc.

- unification not powerful enough $\Longrightarrow$ guessing is state of the art
- problem not limited to HOL

HOL Application: Hardware Verification

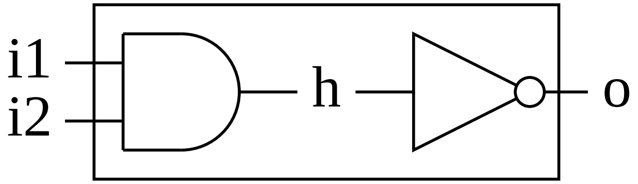
- Some Basic Devices



$\mathsf{NOT}(\mathsf{i},\mathsf{o}) =$ $(\mathsf{o} = \neg \mathsf{i})$	$\mathsf{AND}(\mathsf{i}_1,\mathsf{i}_2,\mathsf{o}) =$ $(\mathsf{o} = (\mathsf{i}_1 \wedge \mathsf{i}_2))$	$\mathsf{OR}(\mathsf{i}_1,\mathsf{i}_2,\mathsf{o}) =$ $(\mathsf{o} = (\mathsf{i}_1 \vee \mathsf{i}_2))$
$\mathsf{NOT}'(\mathsf{i},\mathsf{o}) =$ $(\forall \mathsf{t}. \mathsf{o}(\mathsf{t}) = \neg \mathsf{i}(\mathsf{t}))$	$\mathsf{AND}'(\mathsf{i}_1,\mathsf{i}_2,\mathsf{o}) =$ $(\forall \mathsf{t}. \mathsf{o}(\mathsf{t}) = (\mathsf{i}_1(\mathsf{t}) \wedge \mathsf{i}_2(\mathsf{t})))$	$\mathsf{OR}'(\mathsf{i}_1,\mathsf{i}_2,\mathsf{o}) =$ $(\forall \mathsf{t}. \mathsf{o}(\mathsf{t}) = (\mathsf{i}_1(\mathsf{t}) \vee \mathsf{i}_2(\mathsf{t})))$

HOL Application: Hardware Verification

- Specification of NAND Device

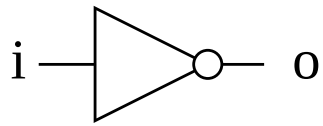


$$\mathsf{NAND{-}SPEC}(\mathsf{i}_1, \mathsf{i}_2, \mathsf{o}) = (\mathsf{o} = \neg(\mathsf{i}_1 \wedge \mathsf{i}_2))$$

$$\mathsf{NAND{-}SPEC}'(\mathsf{i}_1, \mathsf{i}_2, \mathsf{o}) = (\forall \mathsf{t}.\mathsf{o}(\mathsf{t}) = \neg(\mathsf{i}_1(\mathsf{t}) \wedge \mathsf{i}_2(\mathsf{t})))$$

HOL Application: Hardware Verification

- Implementation of NAND Device



$$\mathsf{NAND{-}IMP}(\mathsf{i}_1, \mathsf{i}_2, \mathsf{o}) =$$
$$\exists \mathsf{h}_{\mathsf{o}} \centerdot \mathsf{AND}(\mathsf{i}_1, \mathsf{i}_2, \mathsf{h}) \wedge \mathsf{NOT}(\mathsf{h}, \mathsf{o})$$

$$\mathsf{NAND{-}IMP}'(\mathsf{i}_1, \mathsf{i}_2, \mathsf{o}) =$$
$$\exists \mathsf{h}_{\iota \rightarrow \mathsf{o}} \centerdot \mathsf{AND}(\mathsf{i}_1, \mathsf{i}_2, \mathsf{h}) \wedge \mathsf{NOT}(\mathsf{h}, \mathsf{o})$$

HOL Application: Hardware Verification

- Implementation is correct

$$\mathsf{NAND{-}IMP}(\mathsf{i}_1, \mathsf{i}_2, \mathsf{o}) \Rightarrow \mathsf{NAND{-}SPEC}(\mathsf{i}_1, \mathsf{i}_2, \mathsf{o})$$
$$\mathsf{NAND{-}IMP}'(\mathsf{i}_1, \mathsf{i}_2, \mathsf{o}) \Rightarrow \mathsf{NAND{-}SPEC}'(\mathsf{i}_1, \mathsf{i}_2, \mathsf{o})$$

HOL Application: Hardware Verification

- Implementation is correct

  $$\mathsf{NAND{-}IMP}(i_1, i_2, o) \Rightarrow \mathsf{NAND{-}SPEC}(i_1, i_2, o)$$
  $$\mathsf{NAND{-}IMP}'(i_1, i_2, o) \Rightarrow \mathsf{NAND{-}SPEC}'(i_1, i_2, o)$$

- Definition expansion

HOL Application: Hardware Verification

- Implementation is correct

$$\mathsf{NAND{-}IMP}(i_1, i_2, o) \Rightarrow \mathsf{NAND{-}SPEC}(i_1, i_2, o)$$
$$\mathsf{NAND{-}IMP}'(i_1, i_2, o) \Rightarrow \mathsf{NAND{-}SPEC}'(i_1, i_2, o)$$

- Definition expansion

$$(o = \neg(i_1 \wedge i_2)) \Rightarrow (\exists h_o . \mathsf{AND}(i_1, i_2, h) \wedge \mathsf{NOT}(h, o))$$

HOL Application: Hardware Verification

- Implementation is correct

$$\mathsf{NAND{-}IMP}(i_1, i_2, o) \Rightarrow \mathsf{NAND{-}SPEC}(i_1, i_2, o)$$
$$\mathsf{NAND{-}IMP}'(i_1, i_2, o) \Rightarrow \mathsf{NAND{-}SPEC}'(i_1, i_2, o)$$

- Definition expansion

$$(o = \neg(i_1 \wedge i_2)) \Rightarrow (\exists h_o \centerdot \mathsf{AND}(i_1, i_2, h) \wedge \mathsf{NOT}(h, o))$$
$$(o = \neg(i_1 \wedge i_2)) \Rightarrow (\exists h_o \centerdot (h = (i_1 \wedge i_2)) \wedge (o = \neg h))$$

HOL Application: Hardware Verification

- Implementation is correct

$$\mathsf{NAND{-}IMP}(i_1, i_2, o) \Rightarrow \mathsf{NAND{-}SPEC}(i_1, i_2, o)$$
$$\mathsf{NAND{-}IMP}'(i_1, i_2, o) \Rightarrow \mathsf{NAND{-}SPEC}'(i_1, i_2, o)$$

- Definition expansion

$$(o = \neg(i_1 \wedge i_2)) \Rightarrow (\exists h_o . \mathsf{AND}(i_1, i_2, h) \wedge \mathsf{NOT}(h, o))$$
$$(o = \neg(i_1 \wedge i_2)) \Rightarrow (\exists h_o . (h = (i_1 \wedge i_2)) \wedge (o = \neg h))$$

$$(out = \neg(i_1 \wedge i_2)) \Rightarrow (\exists h_{\iota \to o} . \mathsf{AND}(i_1, i_2, h) \wedge \mathsf{NOT}(h, o))$$

HOL Application: Hardware Verification

- Implementation is correct

$$\mathsf{NAND{-}IMP}(i_1, i_2, o) \Rightarrow \mathsf{NAND{-}SPEC}(i_1, i_2, o)$$
$$\mathsf{NAND{-}IMP}'(i_1, i_2, o) \Rightarrow \mathsf{NAND{-}SPEC}'(i_1, i_2, o)$$

- Definition expansion

$$(o = \neg(i_1 \wedge i_2)) \Rightarrow (\exists h_o . \mathsf{AND}(i_1, i_2, h) \wedge \mathsf{NOT}(h, o))$$
$$(o = \neg(i_1 \wedge i_2)) \Rightarrow (\exists h_o . (h = (i_1 \wedge i_2)) \wedge (o = \neg h))$$

$$(out = \neg(i_1 \wedge i_2)) \Rightarrow (\exists h_{\iota \to o} . \mathsf{AND}(i_1, i_2, h) \wedge \mathsf{NOT}(h, o))$$
$$(out = \neg(i_1 \wedge i_2)) \Rightarrow$$
$$(\exists h_{\iota \to o} . (\forall t_i . (h(t) = (i_1(t) \wedge i_2(t)))) \wedge (\forall t_i . (o(t) = \neg h(t))))$$

HOL Application: Hardware Verification

- Implementation is correct

$$\mathsf{NAND{-}IMP}(i_1, i_2, o) \Rightarrow \mathsf{NAND{-}SPEC}(i_1, i_2, o)$$
$$\mathsf{NAND{-}IMP}'(i_1, i_2, o) \Rightarrow \mathsf{NAND{-}SPEC}'(i_1, i_2, o)$$

- Definition expansion

$$(o = \neg(i_1 \wedge i_2)) \Rightarrow (\exists h_{o}.\mathsf{AND}(i_1, i_2, h) \wedge \mathsf{NOT}(h, o))$$
$$(o = \neg(i_1 \wedge i_2)) \Rightarrow (\exists h_{o}.(h = (i_1 \wedge i_2)) \wedge (o = \neg h))$$

$$(out = \neg(i_1 \wedge i_2)) \Rightarrow (\exists h_{\iota \to o}.\mathsf{AND}(i_1, i_2, h) \wedge \mathsf{NOT}(h, o))$$
$$(out = \neg(i_1 \wedge i_2)) \Rightarrow$$
$$(\exists h_{\iota \to o}.(\forall t_{i}.(h(t) = (i_1(t) \wedge i_2(t)))) \wedge (\forall t_{i}.(o(t) = \neg h(t))))$$

- LEO's proof:
  time: 620ms, cl. gen.: 309, cl. fo-like: 68, proof length: 55 cl.

Extensionality Axioms as Clauses

- EXT-Func$^{\dot=}$: $\forall F_{\alpha\rightarrow\beta}.\forall G_{\alpha\rightarrow\beta}(\forall X_{\beta}.F\ X \doteq G\ X) \Rightarrow F \doteq G$

  Clauses:

  $\mathcal{C}_1 : [p_{\beta\rightarrow o}\ (F\ s_{\beta})]^T \vee [Q\ F]^F \vee [Q\ G]^T$

  $\mathcal{C}_2 : [p_{\beta\rightarrow o}\ (G\ s_{\beta})]^T \vee [Q\ F]^F \vee [Q\ G]^T$

- EXT-Bool$^{\dot=}$: $\forall A_o.\forall B_o.(A \Leftrightarrow B) \Leftrightarrow A \doteq^o B$

  Clauses:

  $\mathcal{C}_1 : [A]^F \vee [B]^F \vee [P\ A]^F \vee [P\ B]^T$

  $\mathcal{C}_2 : [A]^T \vee [B]^T \vee [P\ A]^F \vee [P\ B]^T,$

  $\mathcal{C}_3 : [A]^F \vee [B]^T \vee [p\ A]^T,$

  $\mathcal{C}_4 : [A]^F \vee [B]^T \vee [p\ B]^F,$

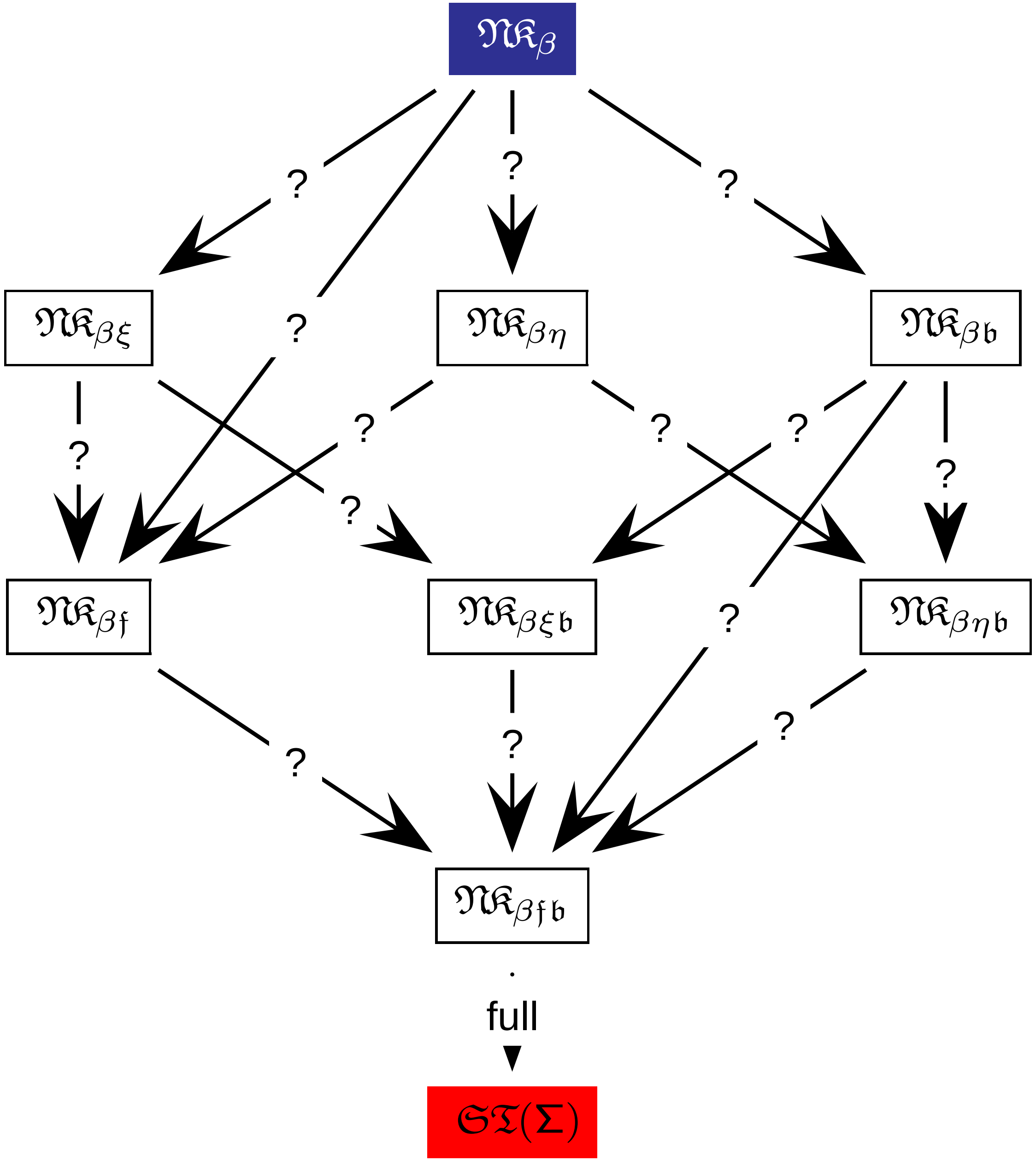
  $\mathcal{C}_5 : [A]^T \vee [B]^F \vee [p\ A]^T,$

  $\mathcal{C}_6 : [A]^T \vee [B]^F \vee [p\ B]^F$

Proof Theory

Calculi for HOL

ND Calculi for HOL



Base Calculus $\mathfrak{NK}_\beta$

— $\mathfrak{NK}(Hyp)$ — $\mathfrak{NK}(\beta)$

— $\mathfrak{NK}(\neg I)$ — $\mathfrak{NK}(\neg E)$

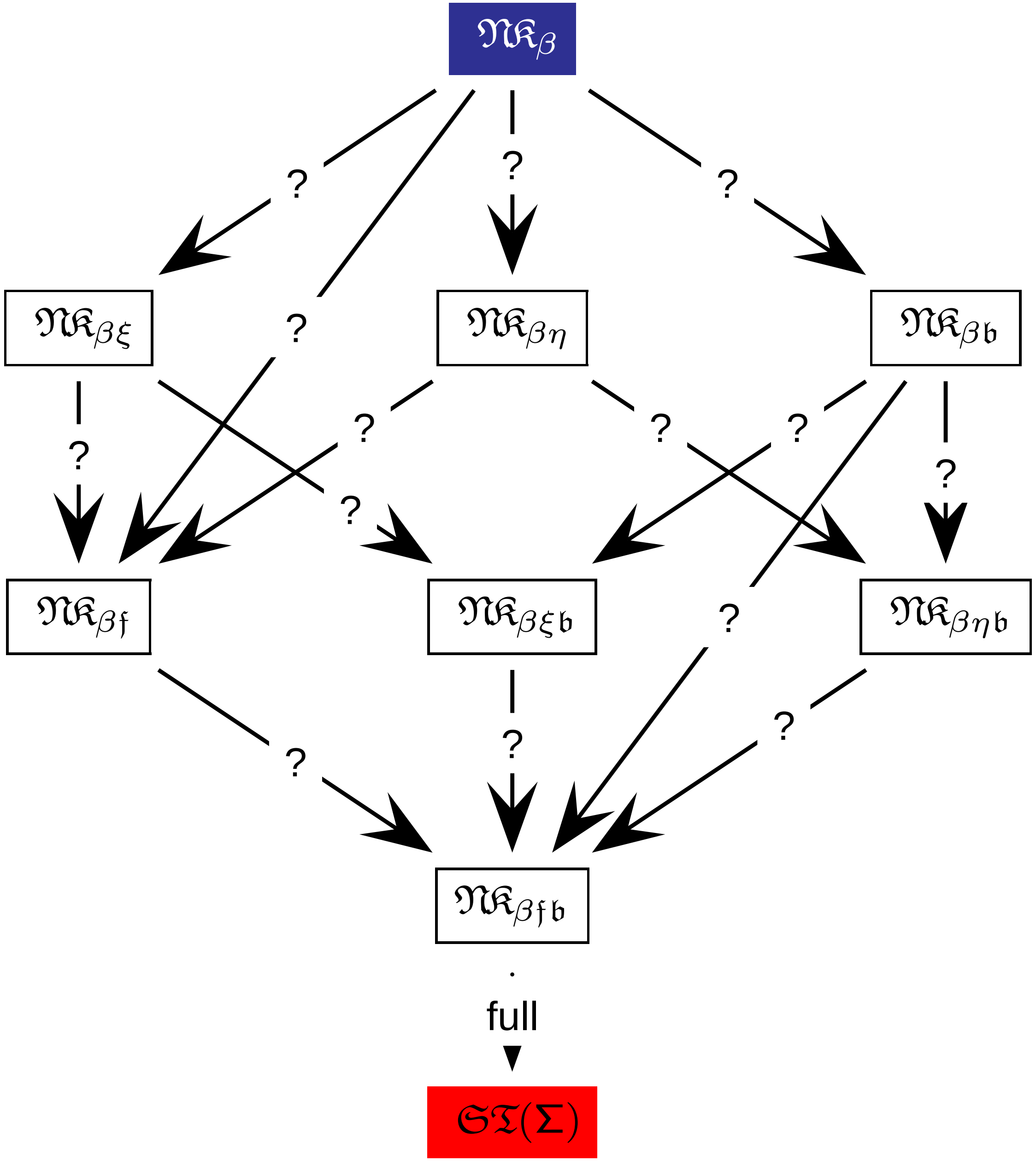
— $\mathfrak{NK}(\vee I_L)$ — $\mathfrak{NK}(\vee I_R)$

— $\mathfrak{NK}(\vee E)$

— $\mathfrak{NK}(\Pi I)^{\mathrm{w}}$

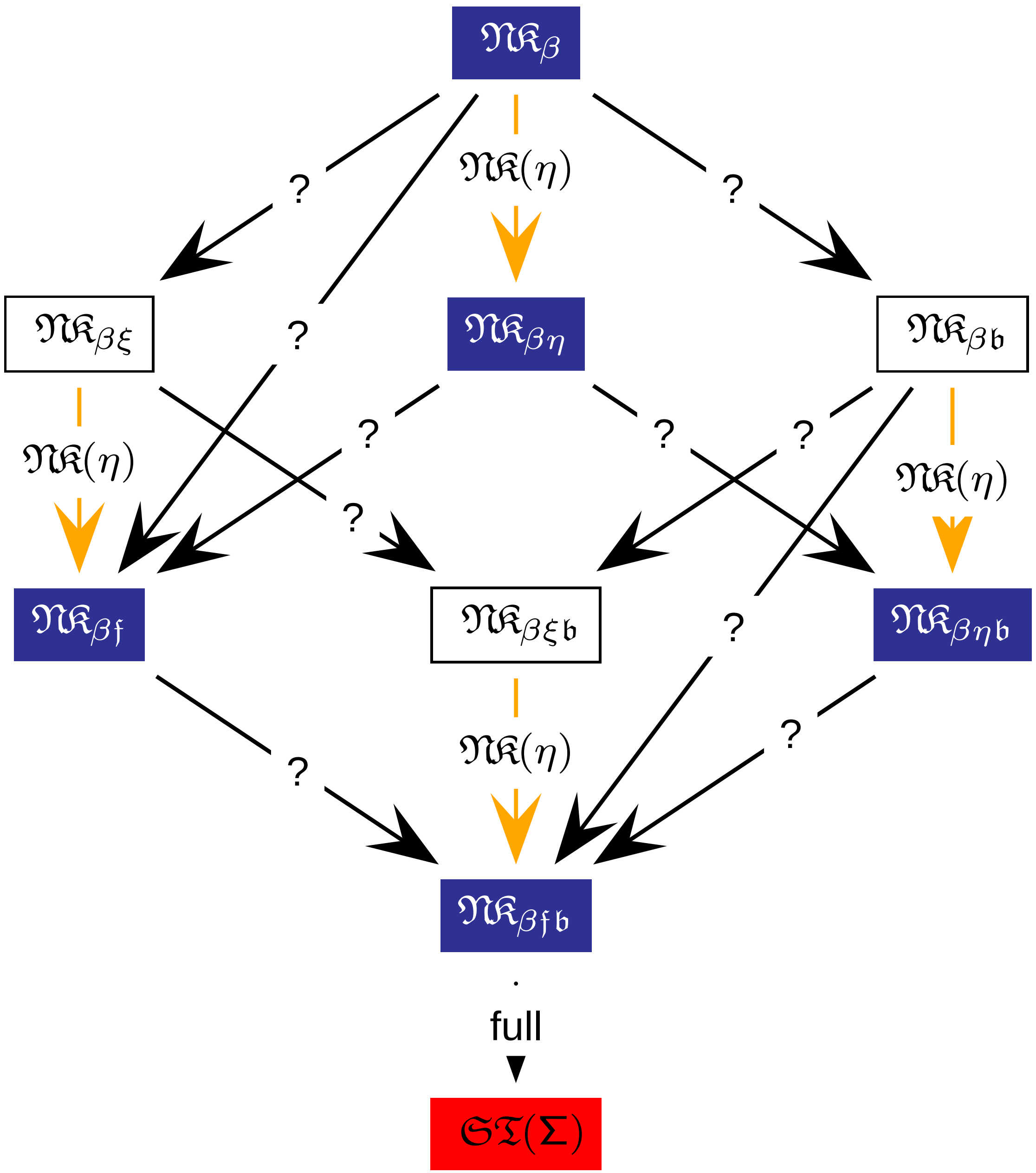
— $\mathfrak{NK}(\Pi E)$ — $\mathfrak{NK}(Contr)$

ND Calculi for HOL



Extensionality Rules

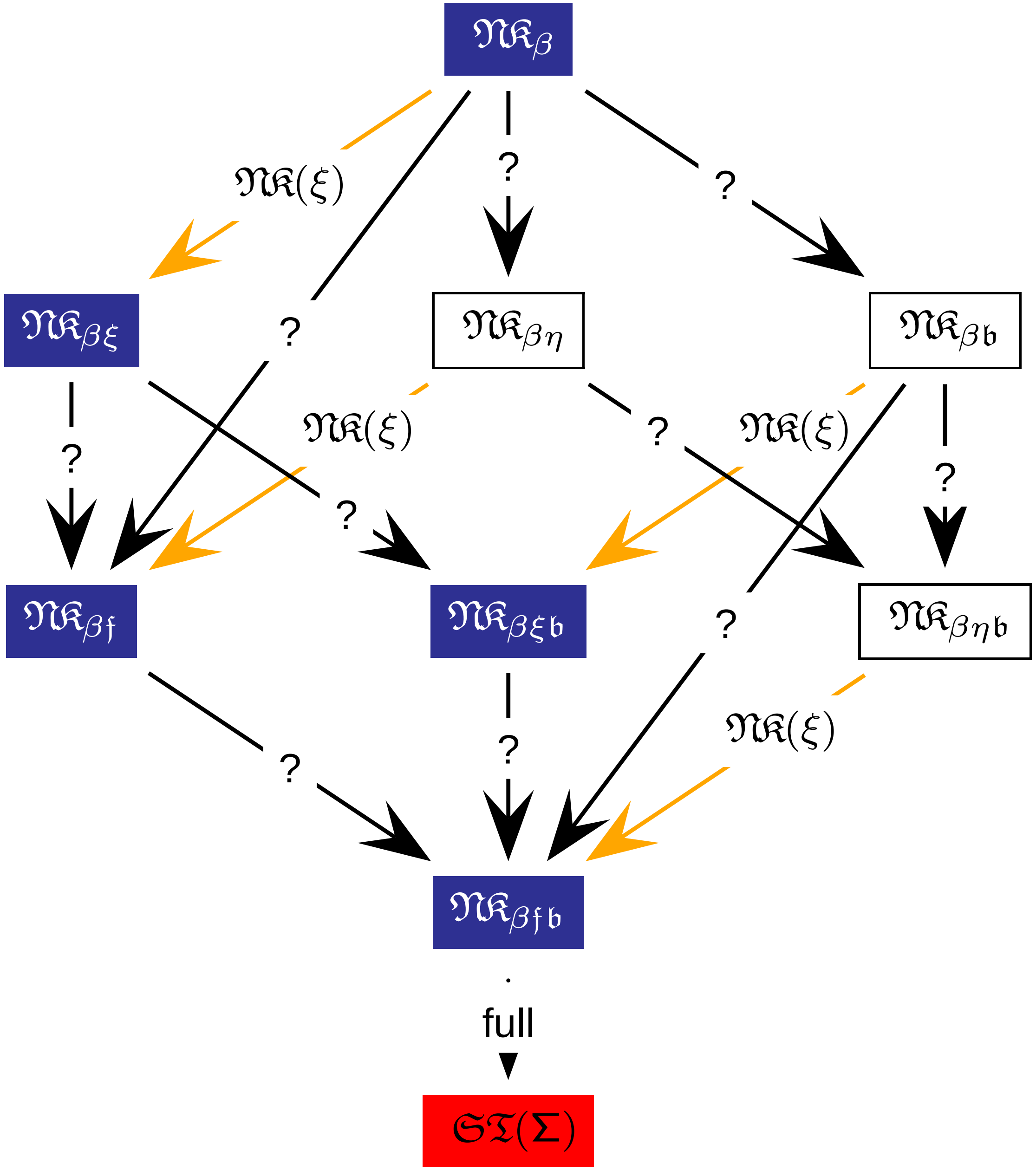
ND Calculi for HOL



Extensionality Rules

$$\frac{\mathbf{A} \stackrel{\beta\eta}{=} \mathbf{B} \qquad \Phi \Vdash \mathbf{A}}{\Phi \Vdash \mathbf{B}}\ \mathfrak{NK}(\eta)$$

ND Calculi for HOL

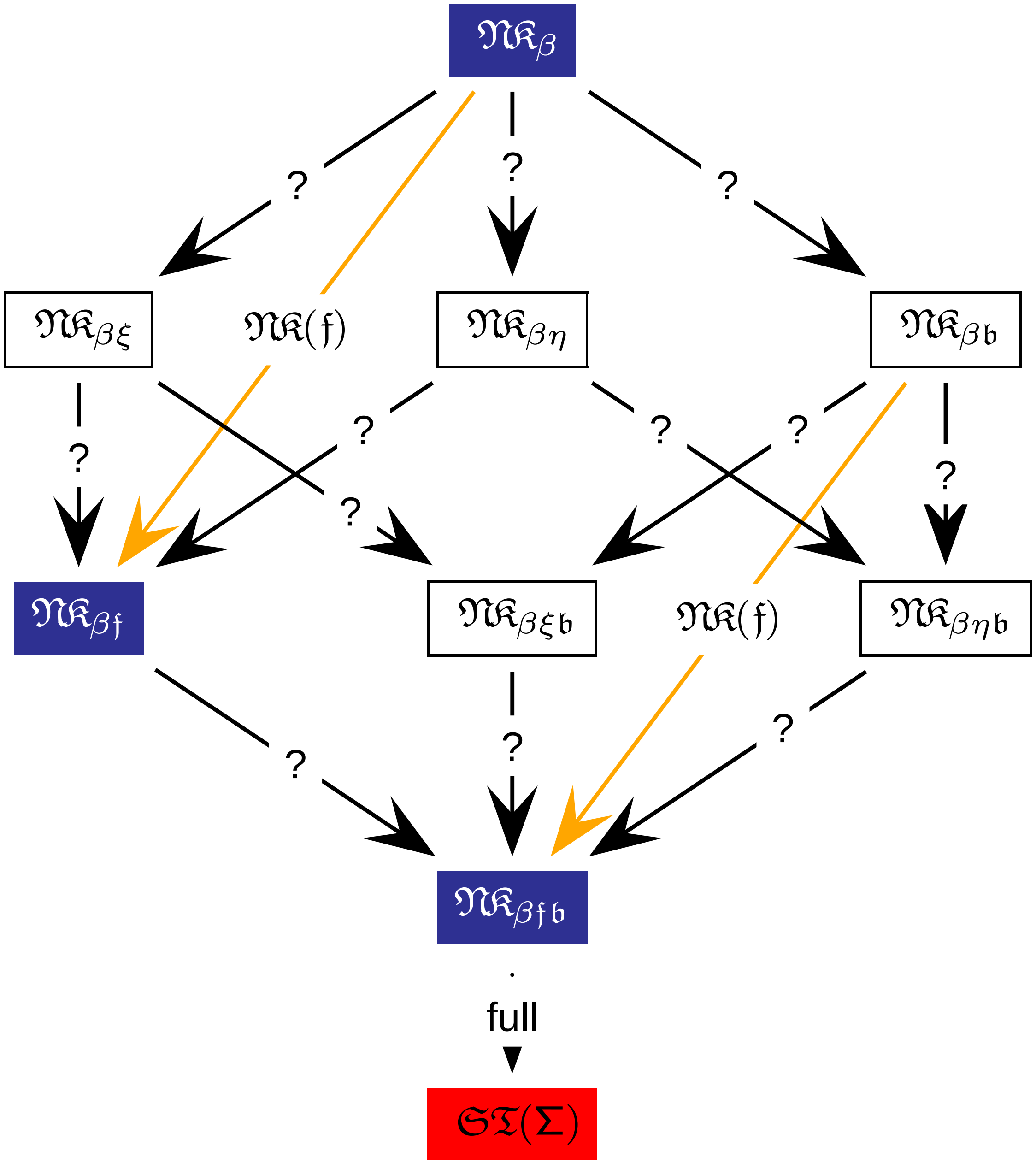


Extensionality Rules

$$\frac{\mathbf{A} \stackrel{\beta\eta}{=} \mathbf{B} \quad \Phi \Vdash \mathbf{A}}{\Phi \Vdash \mathbf{B}}\ \mathfrak{NK}(\eta)$$

$$\frac{\Phi \Vdash \forall x_\alpha.\mathbf{M} \doteq^{\beta} \mathbf{N}}{\Phi \Vdash (\lambda x_\alpha.\mathbf{M}) \doteq^{\beta\alpha} (\lambda x_\alpha.\mathbf{N})}\ \mathfrak{NK}(\xi)$$

ND Calculi for HOL



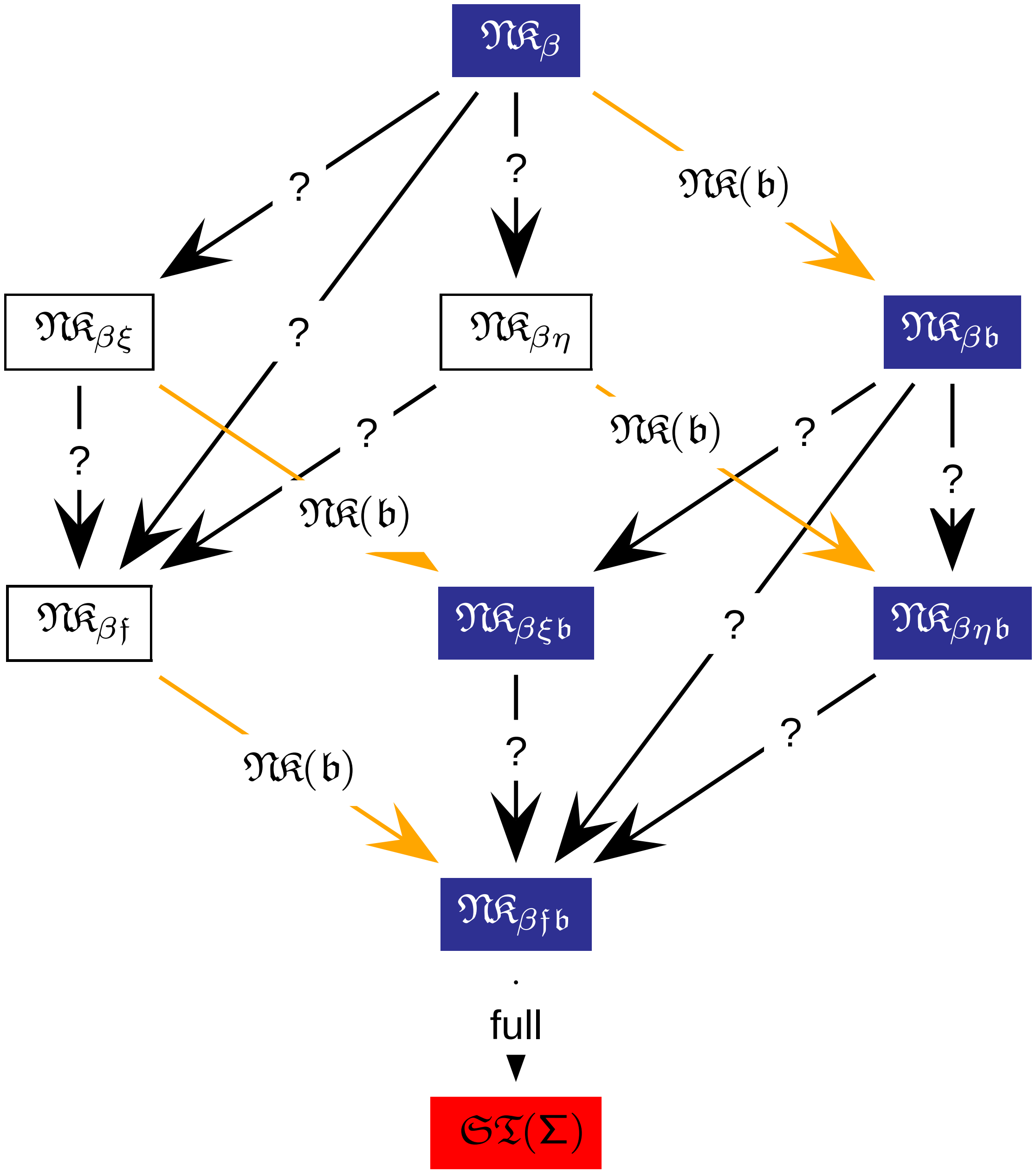
Extensionality Rules

$$\frac{\mathbf{A} \stackrel{\beta\eta}{=} \mathbf{B} \quad \Phi \Vdash \mathbf{A}}{\Phi \Vdash \mathbf{B}}\ \mathfrak{NK}(\eta)$$

$$\frac{\Phi \Vdash \forall X_\alpha.\mathbf{M} \doteq^{\beta} \mathbf{N}}{\Phi \Vdash (\lambda X_\alpha.\mathbf{M}) \doteq^{\beta\alpha} (\lambda X_\alpha.\mathbf{N})}\ \mathfrak{NK}(\xi)$$

$$\frac{\Phi \Vdash \forall X_\alpha.\mathbf{G}X \doteq^{\beta} \mathbf{H}X}{\Phi \Vdash \mathbf{G} \doteq^{\beta\alpha} \mathbf{H}}\ \mathfrak{NK}(\mathfrak{f})$$

ND Calculi for HOL



Extensionality Rules

$$\frac{\mathbf{A} \stackrel{\beta\eta}{=} \mathbf{B} \quad \Phi \Vdash \mathbf{A}}{\Phi \Vdash \mathbf{B}}\ \mathfrak{NK}(\eta)$$

$$\frac{\Phi \Vdash \forall \mathsf{x}_\alpha . \mathbf{M} \doteq^{\beta} \mathbf{N}}{\Phi \Vdash (\lambda \mathsf{x}_\alpha . \mathbf{M}) \doteq^{\beta\alpha} (\lambda \mathsf{x}_\alpha . \mathbf{N})}\ \mathfrak{NK}(\xi)$$

$$\frac{\Phi \Vdash \forall \mathsf{x}_\alpha . \mathbf{G}\mathsf{x} \doteq^{\beta} \mathbf{H}\mathsf{x}}{\Phi \Vdash \mathbf{G} \doteq^{\beta\alpha} \mathbf{H}}\ \mathfrak{NK}(\mathfrak{f})$$

$$\frac{\Phi * \mathbf{A} \Vdash \mathbf{B} \quad \Phi * \mathbf{B} \Vdash \mathbf{A}}{\Phi \Vdash \mathbf{A} \doteq^{o} \mathbf{B}}\ \mathfrak{NK}(\mathfrak{b})$$

Soundness and Completeness of $\mathfrak{NK}_*$

Thm.: Each calculus is sound wrt. the corresponding model class

Thm.: Each calculus complete wrt. the corresponding model class

For this we extended the

- abstract consistency proof method (unifying principle) of

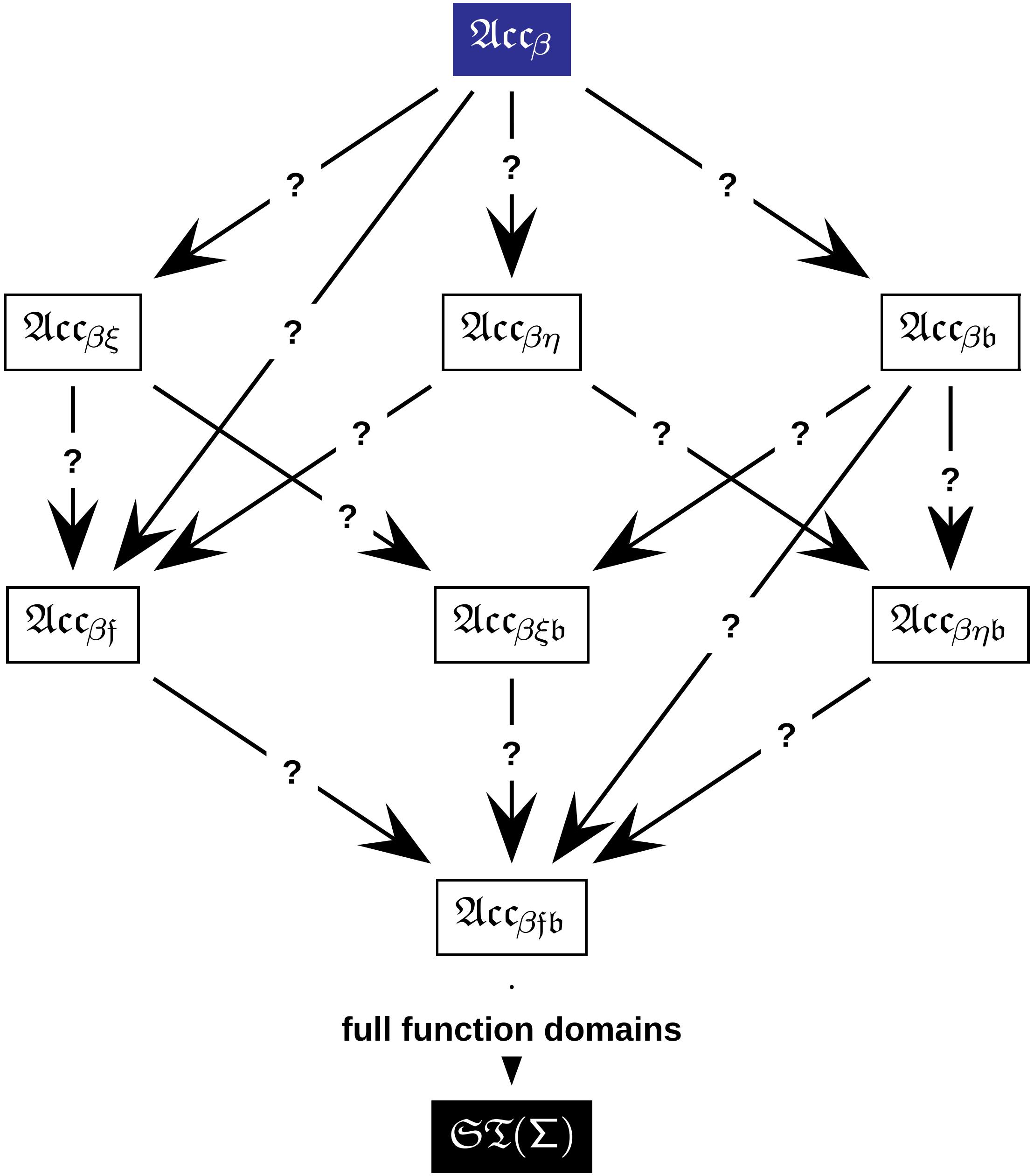
[Smullyan-63]

[Andrews-71]

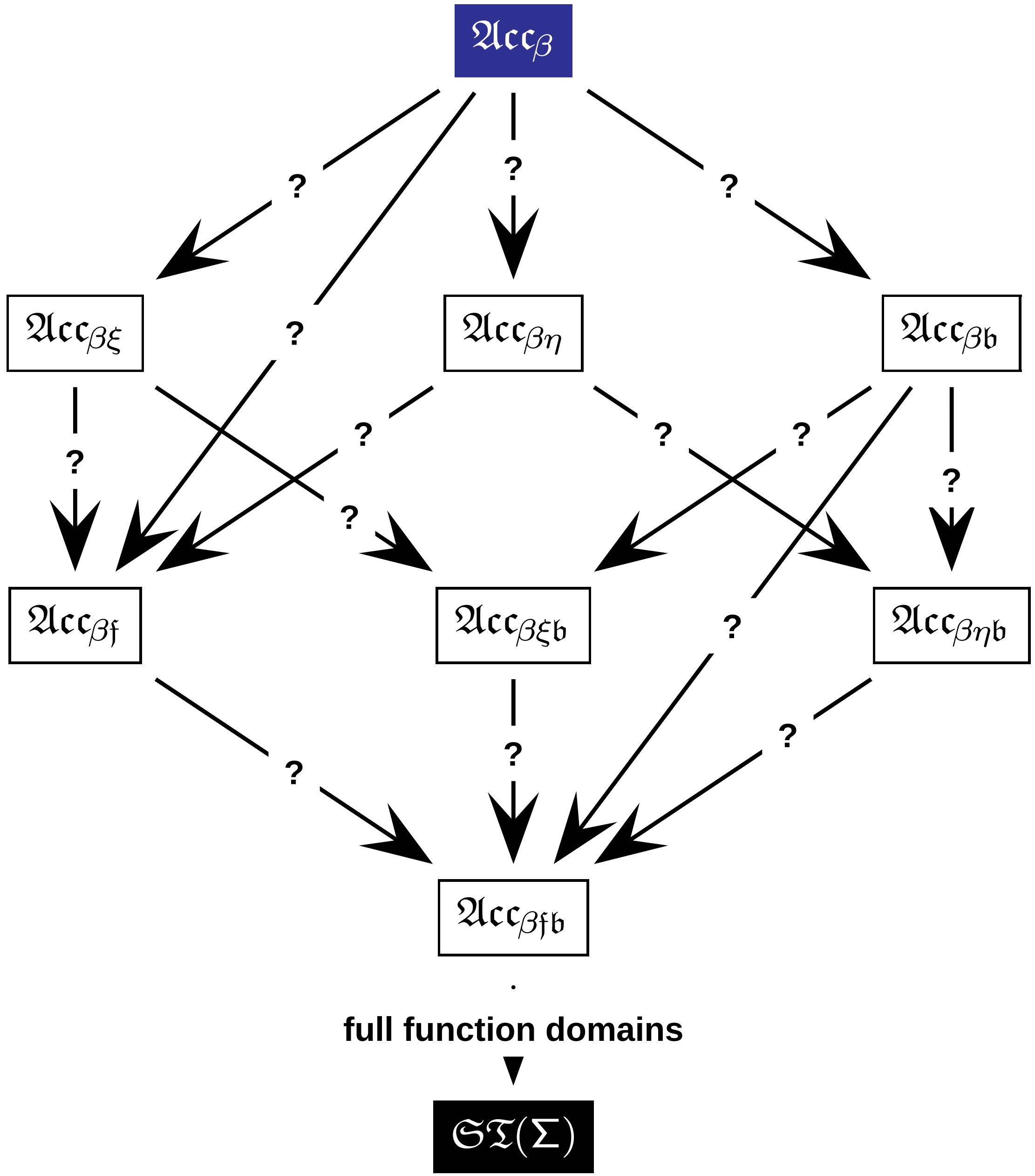
Semantics

Abstract Consistency Proof
Method

Abstract Consistency



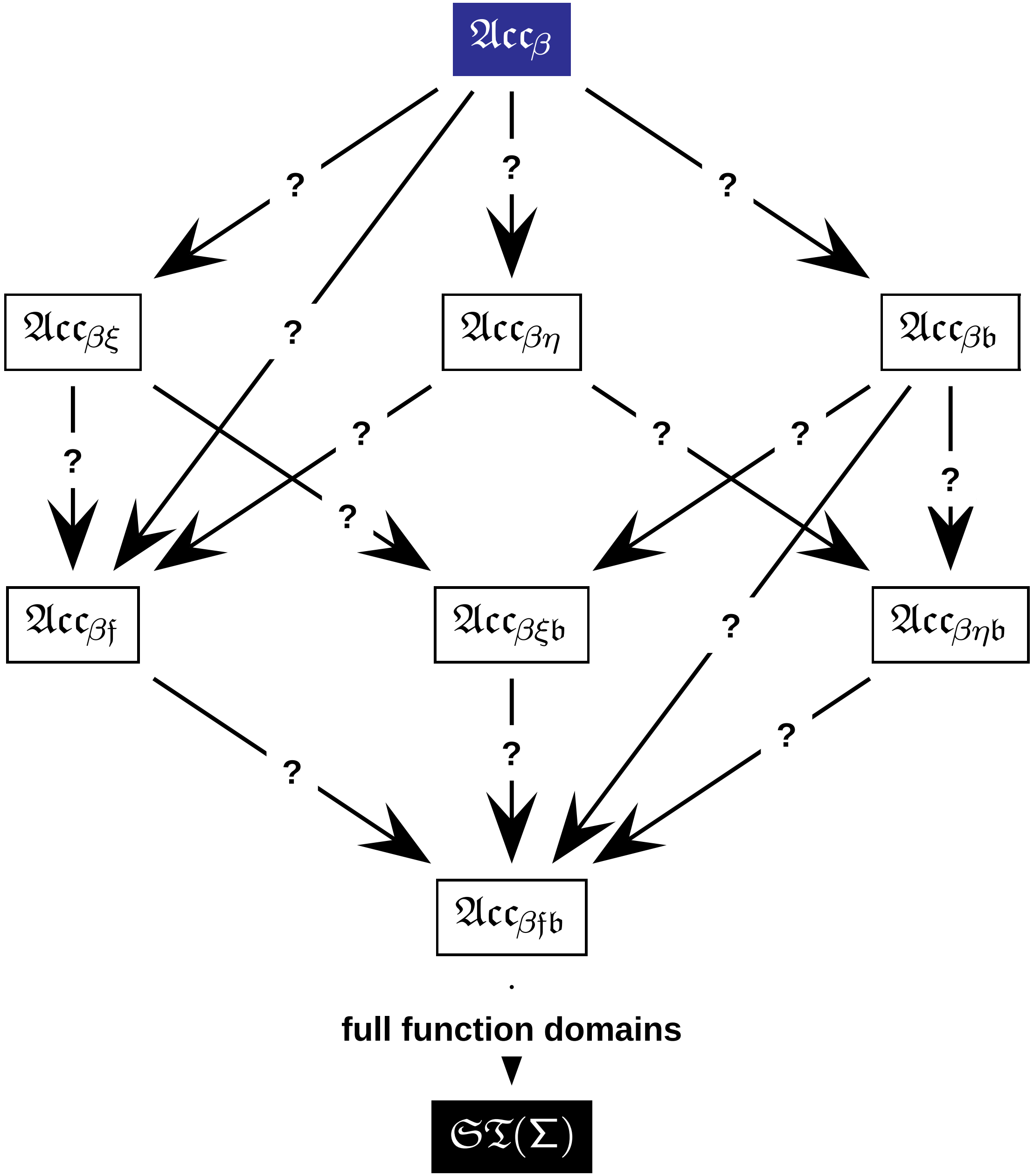
Abstract Consistency



Properties for $\mathfrak{Acc}_\beta$: (Γ_Σ is class of sets of formulas; $\Phi \in \Gamma_\Sigma$)

- ∇_c If $\mathbf{A}$ is atomic, then $\mathbf{A} \notin \Phi$ or $\neg\mathbf{A} \notin \Phi$.
- $\nabla_\neg$ If $\neg\neg\mathbf{A} \in \Phi$, then $\Phi{,}\mathbf{A} \in \Gamma_\Sigma$.
- $\nabla_\vee$ If $\mathbf{A} \vee \mathbf{B} \in \Phi$, then $\Phi{,}\mathbf{A} \in \Gamma_\Sigma$ or $\Phi{,}\mathbf{B} \in \Gamma_\Sigma$.
- $\nabla_\wedge$ If $\neg(\mathbf{A} \vee \mathbf{B}) \in \Phi$, then $\Phi{,}\neg\mathbf{A}{,}\neg\mathbf{B} \in \Gamma_\Sigma$.
- $\nabla_\forall$ If $\Pi^\alpha\mathbf{F} \in \Phi$, then $\Phi{,}\mathbf{F}\mathbf{W} \in \Gamma_\Sigma$ for each $\mathbf{W} \in cwff_\alpha(\Sigma)$.
- $\nabla_\exists$ If $\neg\Pi^\alpha\mathbf{F} \in \Phi$, then $\Phi{,}\neg(\mathbf{F}w) \in \Gamma_\Sigma$ for any parameter $w_\alpha \in \Sigma_\alpha$ which does not occur in any sentence of Φ.

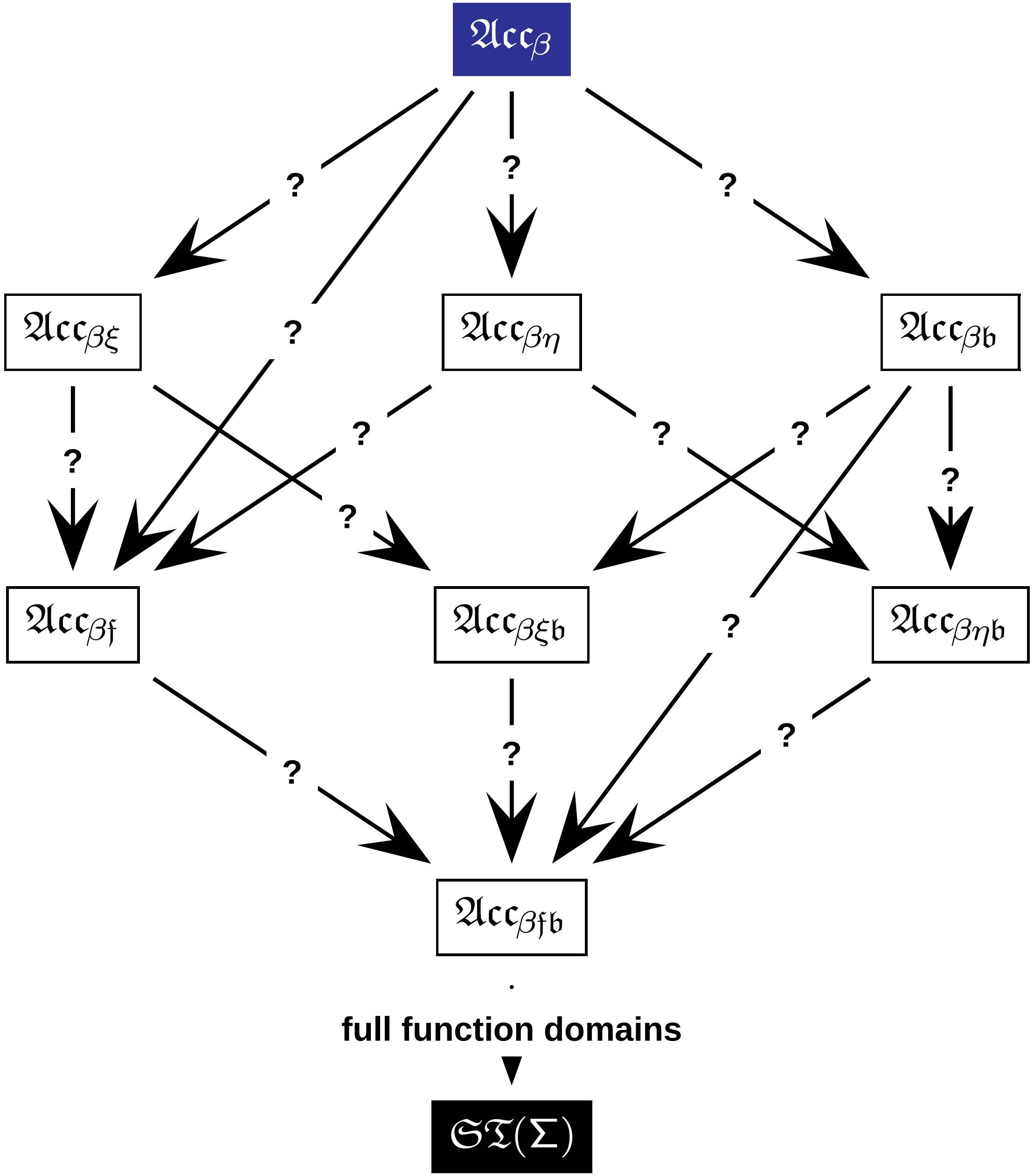
Abstract Consistency



Properties for $\mathfrak{Acc}_\beta$: (Γ_Σ is class of sets of formulas; $\Phi \in \Gamma_\Sigma$)

∇_c	If $\mathbf{A}$ is atomic, then $\mathbf{A} \notin \Phi$ or $\neg\mathbf{A} \notin \Phi$.
$\nabla_\neg$	If $\neg\neg\mathbf{A} \in \Phi$, then $\Phi,\mathbf{A} \in \Gamma_\Sigma$.
∇_β	If $\mathbf{A} =_\beta \mathbf{B}$ and $\mathbf{A} \in \Phi$, then $\Phi,\mathbf{B} \in \Gamma_\Sigma$.
$\nabla_\vee$	If $\mathbf{A} \vee \mathbf{B} \in \Phi$, then $\Phi,\mathbf{A} \in \Gamma_\Sigma$ or $\Phi,\mathbf{B} \in \Gamma_\Sigma$.
$\nabla_\wedge$	If $\neg(\mathbf{A} \vee \mathbf{B}) \in \Phi$, then $\Phi,\neg\mathbf{A},\neg\mathbf{B} \in \Gamma_\Sigma$.
$\nabla_\forall$	If $\Pi^\alpha\mathbf{F} \in \Phi$, then $\Phi,\mathbf{F}\mathbf{W} \in \Gamma_\Sigma$ for each $\mathbf{W} \in \mathit{cwff}_\alpha(\Sigma)$.
$\nabla_\exists$	If $\neg\Pi^\alpha\mathbf{F} \in \Phi$, then $\Phi,\neg(\mathbf{F}w) \in \Gamma_\Sigma$ for any parameter $w_\alpha \in \Sigma_\alpha$ which does not occur in any sentence of Φ.

Abstract Consistency

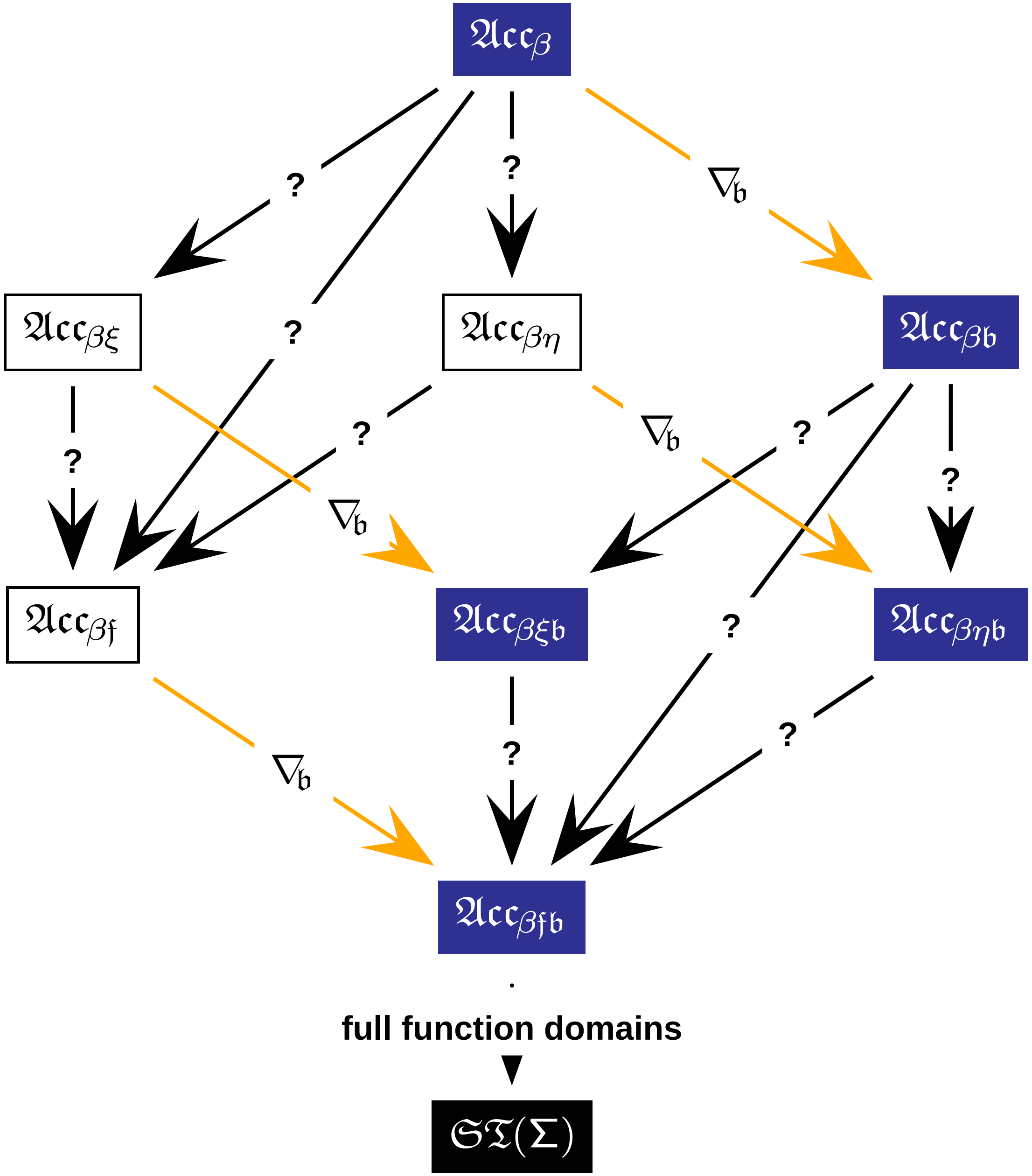


Properties for $\mathfrak{Acc}_{\beta}$

∇_c	...	$\nabla_\vee$	...
$\nabla_\neg$	...	$\nabla_\wedge$	...
∇_β	...	$\nabla_\forall$	...
		$\nabla_\exists$	...

Properties for Extensionality

Abstract Consistency



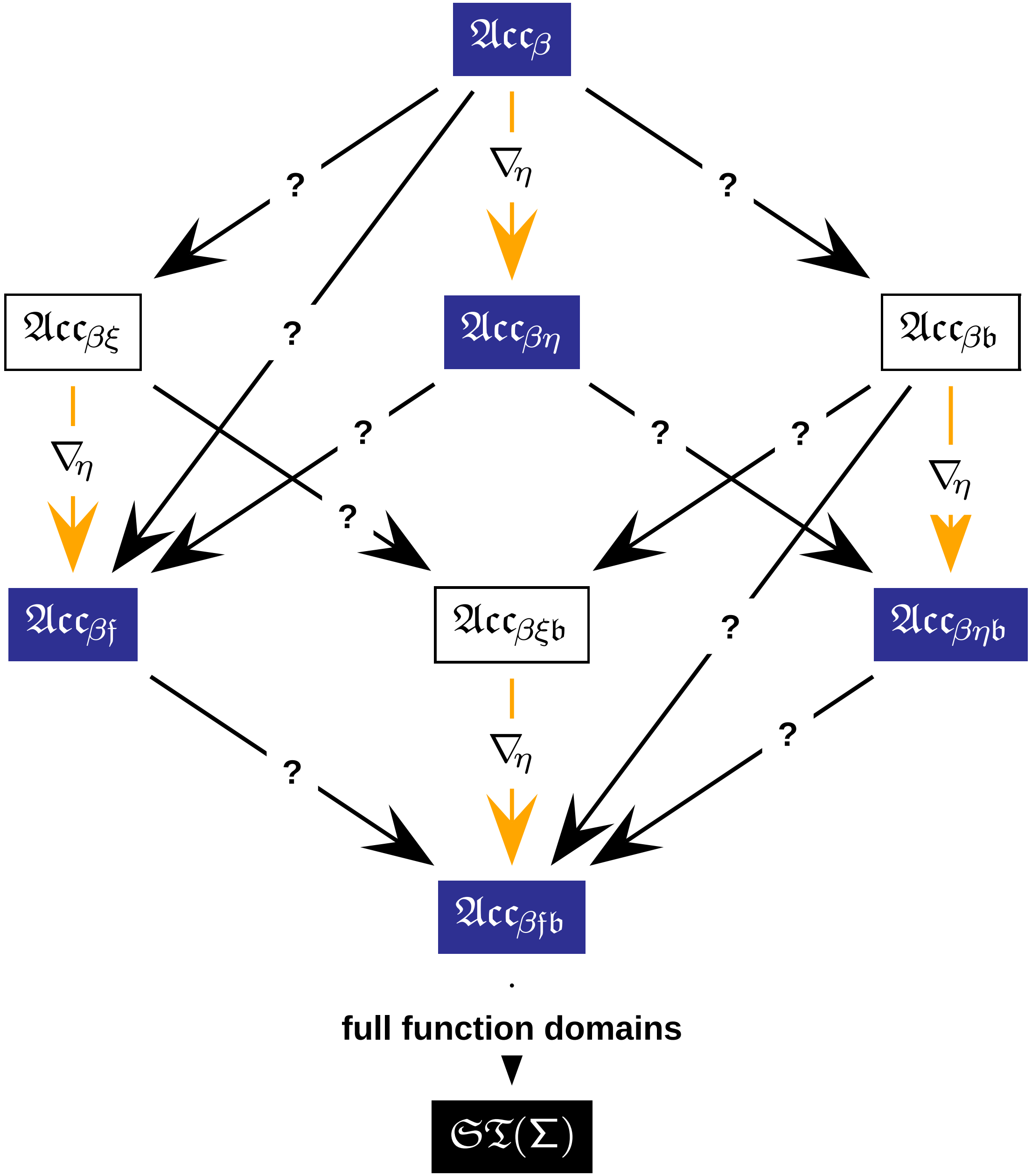
Properties for $\mathfrak{Acc}_\beta$

∇_c	…	$\nabla_\vee$	…
$\nabla_\neg$	…	$\nabla_\wedge$	…
∇_β	…	$\nabla_\forall$	…
		$\nabla_\exists$	…

Properties for Extensionality

$\nabla_\mathfrak{b}$ If $\neg(\mathbf{A} \doteq^\circ \mathbf{B}) \in \Phi$, then $\Phi, \mathbf{A}, \neg\mathbf{B} \in \Gamma_\Sigma$ or $\Phi, \neg\mathbf{A}, \mathbf{B} \in \Gamma_\Sigma$.

Abstract Consistency



Properties for $\mathfrak{Acc}_{\beta}$

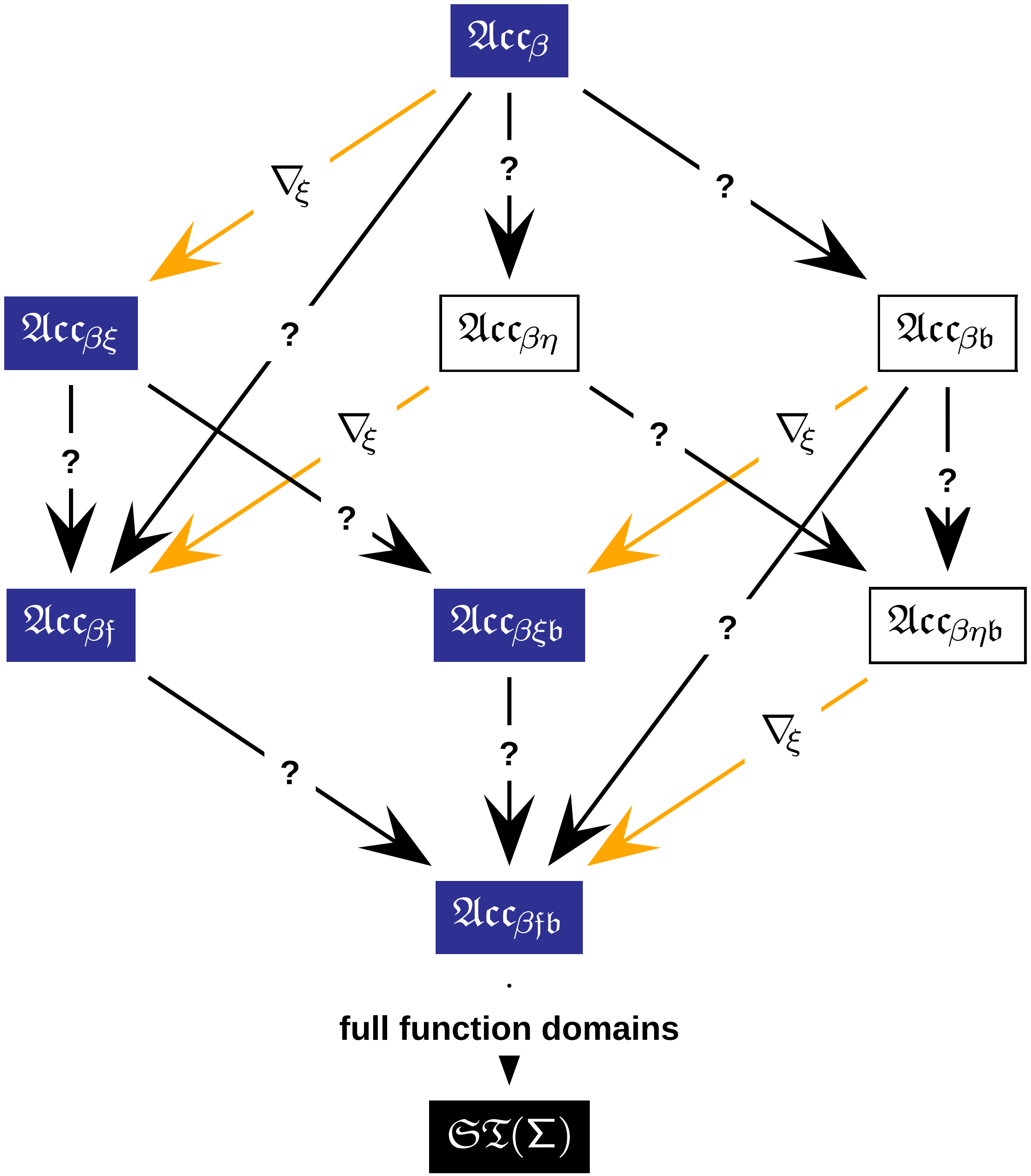
∇_c	...	$\nabla_\vee$	...
$\nabla_\neg$	...	$\nabla_\wedge$	...
∇_β	...	$\nabla_\forall$	...
		$\nabla_\exists$	...

Properties for Extensionality

$\nabla_{\mathfrak{b}}$ If $\neg(\mathbf{A} \doteq^{o} \mathbf{B}) \in \Phi$, then $\Phi, \mathbf{A}, \neg\mathbf{B} \in \Gamma_\Sigma$ or $\Phi, \neg\mathbf{A}, \mathbf{B} \in \Gamma_\Sigma$.

∇_η If $\mathbf{A} \stackrel{\beta\eta}{=} \mathbf{B}$ and $\mathbf{A} \in \Phi$, then $\Phi, \mathbf{B} \in \Gamma_\Sigma$.

Abstract Consistency



Properties for $\mathfrak{Acc}_\beta$

∇_c	...	$\nabla_\vee$	...
$\nabla_\neg$	...	$\nabla_\wedge$	...
∇_β	...	$\nabla_\forall$	...
		$\nabla_\exists$	...

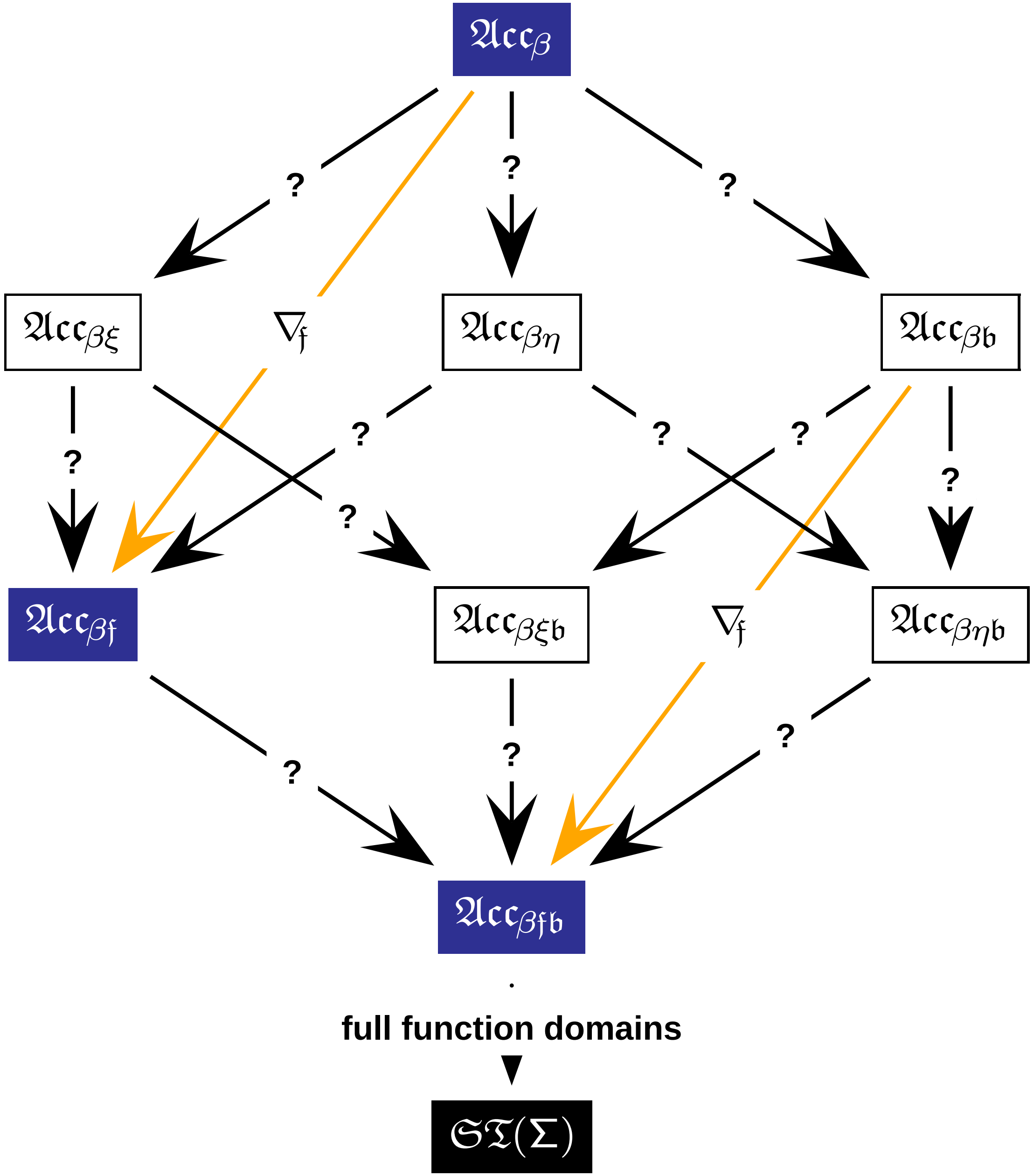
Properties for Extensionality

$\nabla_\mathfrak{b}$ If $\neg(\mathbf{A} \doteq^o \mathbf{B}) \in \Phi$, then $\Phi, \mathbf{A}, \neg\mathbf{B} \in \Gamma_\Sigma$ or $\Phi, \neg\mathbf{A}, \mathbf{B} \in \Gamma_\Sigma$.

∇_η If $\mathbf{A} \stackrel{\beta\eta}{=} \mathbf{B}$ and $\mathbf{A} \in \Phi$, then $\Phi, \mathbf{B} \in \Gamma_\Sigma$.

∇_ξ If $\neg(\lambda X_\alpha.\mathbf{M} \doteq^{\alpha\to\beta} \lambda X_\alpha.\mathbf{N}) \in \Phi$, then $\Phi, \neg([w/X]\mathbf{M} \doteq^\beta [w/X]\mathbf{N}) \in \Gamma_\Sigma$ for any new $w_\alpha \in \Sigma_\alpha$.

Abstract Consistency



Properties for $\mathfrak{Acc}_\beta$

∇_c	…	$\nabla_\vee$	…
$\nabla_\neg$	…	$\nabla_\wedge$	…
∇_β	…	$\nabla_\forall$	…
		$\nabla_\exists$	…

Properties for Extensionality

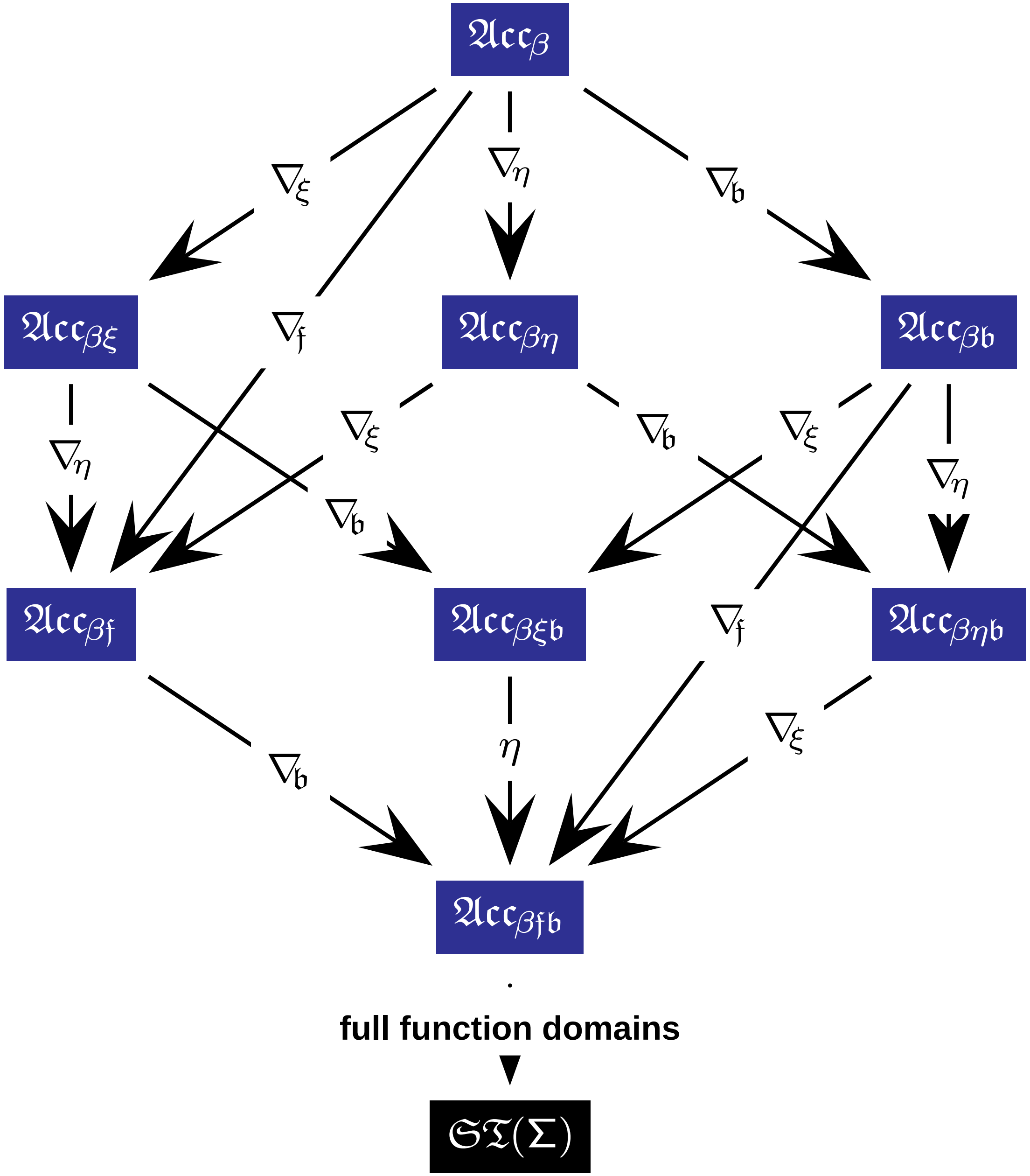
$\nabla_\mathfrak{b}$ If $\neg(\mathbf{A} \doteq^o \mathbf{B}) \in \Phi$, then $\Phi,\mathbf{A},\neg\mathbf{B} \in \Gamma_\Sigma$ or $\Phi,\neg\mathbf{A},\mathbf{B} \in \Gamma_\Sigma$.

∇_η If $\mathbf{A} \stackrel{\beta\eta}{=} \mathbf{B}$ and $\mathbf{A} \in \Phi$, then $\Phi,\mathbf{B} \in \Gamma_\Sigma$.

∇_ξ If $\neg(\lambda X_\alpha.\mathbf{M} \doteq^{\alpha\to\beta} \lambda X_\alpha.\mathbf{N}) \in \Phi$, then $\Phi,\neg([w/X]\mathbf{M} \doteq^\beta [w/X]\mathbf{N}) \in \Gamma_\Sigma$ for any new $w_\alpha \in \Sigma_\alpha$.

$\nabla_\mathfrak{f}$ If $\neg(\mathbf{G} \doteq^{\alpha\to\beta} \mathbf{H}) \in \Phi$, then $\Phi,\neg(\mathbf{G}w \doteq^\beta \mathbf{H}w) \in \Gamma_\Sigma$ for any new $w_\alpha \in \Sigma_\alpha$.

Abstract Consistency



Properties for $\mathfrak{Acc}_\beta$

∇_c	...	$\nabla_\vee$	...
$\nabla_\neg$	...	$\nabla_\wedge$	...
∇_β	...	$\nabla_\forall$	...
		$\nabla_\exists$	...

Properties for Extensionality

$\nabla_\mathfrak{b}$ If $\neg(\mathbf{A} \doteq^o \mathbf{B}) \in \Phi$, then $\Phi, \mathbf{A}, \neg\mathbf{B} \in \Gamma_\Sigma$ or $\Phi, \neg\mathbf{A}, \mathbf{B} \in \Gamma_\Sigma$.

∇_η If $\mathbf{A} \stackrel{\beta\eta}{=} \mathbf{B}$ and $\mathbf{A} \in \Phi$, then $\Phi, \mathbf{B} \in \Gamma_\Sigma$.

∇_ξ If $\neg(\lambda X_\alpha.\mathbf{M} \doteq^{\alpha\to\beta} \lambda X_\alpha.\mathbf{N}) \in \Phi$, then $\Phi, \neg([w/X]\mathbf{M} \doteq^\beta [w/X]\mathbf{N}) \in \Gamma_\Sigma$ for any new $w_\alpha \in \Sigma_\alpha$.

$\nabla_\mathfrak{f}$ If $\neg(\mathbf{G} \doteq^{\alpha\to\beta} \mathbf{H}) \in \Phi$, then $\Phi, \neg(\mathbf{G}w \doteq^\beta \mathbf{H}w) \in \Gamma_\Sigma$ for any new $w_\alpha \in \Sigma_\alpha$.

Abstract Consistency

Thm.: (Model Existence)
Saturated abstract consistency implies model existence

Appl.: (Completeness proofs by pure syntactical means)

$$\Gamma_\Sigma^{G} := \{\Phi \mid \Phi \text{ is } C\text{-consistent}\} \text{ is a saturated } \mathfrak{Acc}_*$$