Investigations of Proof Theory and Automated Reasoning for Non-classical Logics

Cosimo Perini Brogi

supervised by
Prof. Sara Negri
Prof. Giuseppe Rosolini

Università degli Studi di Genova
Dipartimento di Matematica
July 12, 2022

A theory of proofs

The proof theorist considers proofs as her study objects, and prove some properties about them.

Q: Right, but what is a proof then?

The physicist's A: A token of evidence!

The ordinary mathematician's A: A convincing *mathematical* argument!

The logician's A: A *logically sound* argument!

The proof theorist's A:

Let's say that proof system consists of a set of starting formal expressions together with inference rules. Its principal aim is to find proofs of valid expressions w.r.t. a given logic.

A proof (or derivation) in a proof system is obtained by application of the inference rules to starting expressions, followed by further application of the inference rules to the conclusion, and so on, recursively.

A theorem (or lemma) in such a system is the formal expression obtained after a finite run of the procedure just sketched.

A theory of proofs

The proof theorist considers proofs as her study objects, and prove some properties about them.

Q: Right, but what is a proof then?

The physicist's A: A token of evidence!

The ordinary mathematician's A: A convincing *mathematical* argument!

The logician's A: A *logically sound* argument!

The proof theorist's A:

Let's say that proof system consists of a set of starting formal expressions together with inference rules. Its principal aim is to find proofs of valid expressions w.r.t. a given logic.

A proof (or derivation) in a proof system is obtained by application of the inference rules to starting expressions, followed by further application of the inference rules to the conclusion, and so on, recursively.

A theorem (or lemma) in such a system is the formal expression obtained after a finite run of the procedure just sketched.

A theory of proofs

The proof theorist considers proofs as her study objects, and prove some properties about them.

Q: Right, but what is a proof then?

The physicist's A: A token of evidence!

The ordinary mathematician's A: A convincing *mathematical* argument!

The logician's A: A *logically sound* argument!

The proof theorist's A:

Let's say that proof system consists of a set of starting formal expressions together with inference rules. Its principal aim is to find proofs of valid expressions w.r.t. a given logic.

A proof (or derivation) in a proof system is obtained by application of the inference rules to starting expressions, followed by further application of the inference rules to the conclusion, and so on, recursively.

A theorem (or lemma) in such a system is the formal expression obtained after a finite run of the procedure just sketched.

A theory of proofs

The proof theorist considers proofs as her study objects, and prove some properties about them.

Q: Right, but what is a proof then?

The physicist's A: A token of evidence!

The ordinary mathematician's A: A convincing *mathematical* argument!

The logician's A: A *logically sound* argument!

The proof theorist's A:

Let's say that proof system consists of a set of starting formal expressions together with inference rules. Its principal aim is to find proofs of valid expressions w.r.t. a given logic.

A proof (or derivation) in a proof system is obtained by application of the inference rules to starting expressions, followed by further application of the inference rules to the conclusion, and so on, recursively.

A theorem (or lemma) in such a system is the formal expression obtained after a finite run of the procedure just sketched.

A theory of proofs

The proof theorist considers proofs as her study objects, and prove some properties about them.

Q: Right, but what is a proof then?

The physicist's A: A token of evidence!

The ordinary mathematician's A: A convincing *mathematical* argument!

The logician's A: A *logically sound* argument!

The proof theorist's A:

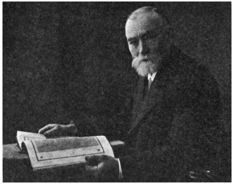
Let's say that proof system consists of a set of starting formal expressions together with inference rules. Its principal aim is to find proofs of valid expressions w.r.t. a given logic.

A proof (or derivation) in a proof system is obtained by application of the inference rules to starting expressions, followed by further application of the inference rules to the conclusion, and so on, recursively.

A theorem (or lemma) in such a system is the formal expression obtained after a finite run of the procedure just sketched.

Mathematical Paradigms

Axiomatic calculi

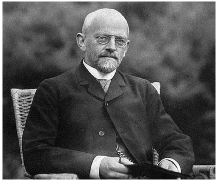


Starting points: Instances of axiom schemas

Rules: Two inference rules are enough (including *MP*)

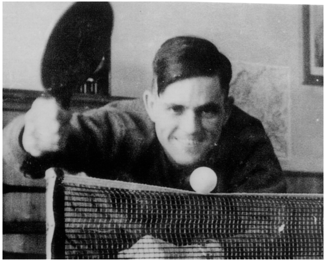
Identity law:

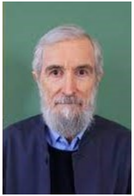
1.	$(A \to ((A \to A) \to A)) \to ((A \to (A \to A)) \to (A \to A))$	Frege
2.	$A \to ((A \to A) \to A)$	a fortiori
3.	$(A \to (A \to A)) \to (A \to A)$	$MP: 1, 2$
4.	$A \to (A \to A)$	a fortiori
5.	$A \to A$	$MP: 3, 4$



Starting points: Assumptions $\approx$ Leaves of a tree

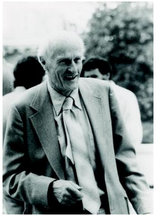
Rules: For each logical operator of the language, we have an introduction rule and an elimination rule $\approx$ Generating new tree nodes

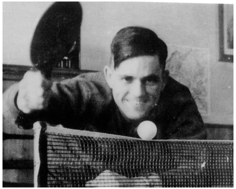




Identity Law:

$$\frac{[A]^1}{A \to A}\ {\to}\mathcal{I}{:}1$$





$\Gamma \Rightarrow \Delta$, where Γ, Δ are finite multisets of formulas.

Starting points: Initial sequents $\approx$ Trivial deductions

Rules: For each logical operator of the language, we have a right rule $\approx$ introduction rule in ND; and a left rule $\approx$ (generalised) elimination rule in ND

Identity law:

$$A \Rightarrow A$$

By adopting Gentzen's formalisms it is possible to perform a fine grained analysis of the structure of proofs. In particular, in each paradigm, it is possible to prove canonical form theorems for formal derivations:

- each classical or intuitionistic deduction can be effectively turned into a normal deduction, in which no detours occur

$$\dfrac{\dfrac{\begin{matrix}\vdots\\ B\end{matrix}}{A \to B}\to\mathcal{I} \qquad \begin{matrix}\vdots\\ A\end{matrix}}{B}\to\mathcal{E} \qquad \rightsquigarrow \qquad \begin{matrix}\vdots\\ B\end{matrix}$$

- the cut rule

$$\dfrac{\Gamma \Rightarrow \Delta, A \qquad A, \Gamma' \Rightarrow \Delta'}{\Gamma, \Gamma' \Rightarrow \Delta, \Delta'}\mathit{Cut}$$

  can be effectively removed from every derivation in the G3-style sequent calculi for classical and intuitionistic logics

By adopting Gentzen's formalisms it is possible to perform a fine grained analysis of the structure of proofs. In particular, in each paradigm, it is possible to prove canonical form theorems for formal derivations:

- each classical or intuitionistic deduction can be effectively turned into a normal deduction, in which no detours occur

$$\dfrac{\dfrac{\begin{array}{c}\vdots\\ B\end{array}}{A \to B}\to\mathcal{I} \qquad \begin{array}{c}\vdots\\ A\end{array}}{B}\to\mathcal{E} \qquad \rightsquigarrow \qquad \begin{array}{c}\vdots\\ B\end{array}$$

- the cut rule

$$\dfrac{\Gamma \Rightarrow \Delta, A \qquad A, \Gamma' \Rightarrow \Delta'}{\Gamma, \Gamma' \Rightarrow \Delta, \Delta'}\ \mathit{Cut}$$

can be effectively removed from every derivation in the G3-style sequent calculi for classical and intuitionistic logics

By adopting Gentzen's formalisms it is possible to perform a fine grained analysis of the structure of proofs. In particular, in each paradigm, it is possible to prove canonical form theorems for formal derivations:

- each classical or intuitionistic deduction can be effectively turned into a normal deduction, in which no detours occur

$$\dfrac{\dfrac{\begin{array}{c}\vdots\\ B\end{array}}{A \to B}\,{\to}\mathcal{I} \qquad \begin{array}{c}\vdots\\ A\end{array}}{B}\,{\to}\mathcal{E} \quad \rightsquigarrow \quad \begin{array}{c}\vdots\\ B\end{array}$$

- the cut rule

$$\dfrac{\Gamma \Rightarrow \Delta, A \qquad A, \Gamma' \Rightarrow \Delta'}{\Gamma, \Gamma' \Rightarrow \Delta, \Delta'}\,\mathit{Cut}$$

  can be effectively removed from every derivation in the G3-style sequent calculi for classical and intuitionistic logics

Normal form theorems for Gentzen's calculi have a direct application in the field of computerised reasoning.
In fact, theorem provers do work since they are based on logical calculi having good structural properties:

- Analyticity:
  ↬ *no guesses are required* to the prover when developing a formal proof;
- Avoiding of backtracking:
  ↬ *no bit of information gets lost* during the procedure;
- Termination:
  ↬ no loops of the prover

Normal form theorems for Gentzen's calculi have a direct application in the field of computerised reasoning.
In fact, theorem provers do work since they are based on logical calculi having good structural properties:

- Analyticity:
  ↬ *no guesses are required* to the prover when developing a formal proof;
- Avoiding of backtracking:
  ↬ *no bit of information gets lost* during the procedure;
- Termination:
  ↬ no loops of the prover

Normal form theorems for Gentzen's calculi have a direct application in the field of computerised reasoning.
In fact, theorem provers do work since they are based on logical calculi having good structural properties:

- Analyticity:
  ↬ *no guesses are required* to the prover when developing a formal proof;
- Avoiding of backtracking:
  ↬ *no bit of information gets lost* during the procedure;
- Termination:
  ↬ no loops of the prover

Normal form theorems for Gentzen's calculi have a direct application in the field of computerised reasoning.
In fact, theorem provers do work since they are based on logical calculi having good structural properties:

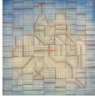
- Analyticity:
  ↬ *no guesses are required* to the prover when developing a formal proof;
- Avoiding of backtracking:
  ↬ *no bit of information gets lost* during the procedure;
- Termination:
  ↬ no loops of the prover

Overview

Introduction

Structural Proof Theory for Modal Logics
- Verification-based epistemic states
- Intuitionistic belief
- Intuitionistic knowledge
- Intuitionistic strong Löb logic
- Interpretability logics

Automated reasoning
- Gödel-Löb in HOL Light
- Universal algebra in UniMath



I. Structural Proof Theory for Modal Logics

Proof theory for modal logics

If we enrich our base syntax by modal operators, the design and the structural analysis of calculi for modal logics may become painful.

Nevertheless, it is not impossible to design well-behaved Gentzen-style systems for those logics

In this first part of the talk, I will propose

- “standard” natural deduction calculi for three intuitionistic modal logics;
- an “enriched” sequent calculus for a wide family of classical modal logics for arithmetical interpretability

and study their structural properties.

Proof theory for modal logics

If we enrich our base syntax by modal operators, the design and the structural analysis of calculi for modal logics may become painful.

Nevertheless, it is not impossible to design well-behaved Gentzen-style systems for those logics

In this first part of the talk, I will propose

- "standard" natural deduction calculi for three intuitionistic modal logics;
- an "enriched" sequent calculus for a wide family of classical modal logics for arithmetical interpretability

and study their structural properties.

Proof theory for modal logics

If we enrich our base syntax by modal operators, the design and the structural analysis of calculi for modal logics may become painful.

Nevertheless, it is not impossible to design well-behaved Gentzen-style systems for those logics

In this first part of the talk, I will propose

- "standard" natural deduction calculi for three intuitionistic modal logics;
- an "enriched" sequent calculus for a wide family of classical modal logics for arithmetical interpretability

and study their structural properties.

Proof theory for modal logics

If we enrich our base syntax by modal operators, the design and the structural analysis of calculi for modal logics may become painful.

Nevertheless, it is not impossible to design well-behaved Gentzen-style systems for those logics

In this first part of the talk, I will propose

- "standard" natural deduction calculi for three intuitionistic modal logics;
- an "enriched" sequent calculus for a wide family of classical modal logics for arithmetical interpretability

and study their structural properties.

Verification-based epistemic states

Brouwer-Heyting-Kolgomorov interpretation

We adopt the view that an ***an intuitionistic epistemic state (belief or knowledge) is the result of verification*** *where a verification is evidence considered sufficiently conclusive for practical purposes.*

(Artemov and Protopopescu 2016)

We can read any formula $\Box A$ as asserting that A has a proof which is not necessarily specified in the process of verification, or more generally that it is verified that A holds in some not specified constructive sense.

This allows to apply intuitionistic epistemic reasoning in various contexts which are not necessarily in the standard domain of BHK; for instance:

- Testimony of authority;
- Zero-knowledge protocols;
- Highly probable truth;

$\vdots$

Intuitionistic belief

(Artemov and Protopopescu 2016)

Axioms

1. Axioms of propositional intuitionistic logic;
2. $\Box(A \to B) \to \Box A \to \Box B$; (K-scheme)
3. $A \to \Box A$. (co-reflection)

Rules

$$\frac{A \to B \qquad A}{B}\, MP$$

A model for $\mathbb{IEL}^-$ is a quadruple $\langle W, \leq, v, E\rangle$ where

- $\langle W, \leq, v\rangle$ is a standard model for intuitionistic propositional logic;
- E is a binary 'knowledge' relation on W such that:
  - if xEy, then $x \leq y$; and
  - if $x \leq y$ and yEz, then xEz;
- v extends to a forcing relation $\Vdash$ such that
  - $x \Vdash \Box A$ iff $y \Vdash A$ for all y such that xEy.

Modal adequacy for $\mathbb{IEL}^-$

$\mathbb{IEL}^-$ is sound and complete w.r.t. $\mathbb{IEL}^-$ relational frames.

Natural deduction for intuitionistic belief

(PB 2021)

Let IEL^- be the calculus extending NJp by the following rule:

$$
\dfrac{\begin{matrix}\Gamma_1 \\ \vdots \\ \Box A_1\end{matrix} \quad \cdots \quad \begin{matrix}\Gamma_n \\ \vdots \\ \Box A_n\end{matrix} \qquad \begin{matrix}[A_1, \cdots, A_n], \Delta \\ \vdots \\ B\end{matrix}}{\Box B}\ \Box\mathcal{I}
$$

where Γ and Δ are *multisets of formulas*, and $A_1, \cdots, A_n$ are *all* discharged.[1]

We say that B is the major premise of the rule, and each $\Box A_i$ is a minor premise, whose corresponding discharged assumption is A_i.

[1] Notice that this calculus differs from the system introduced in (de Paiva and Ritter 2004) by allowing the set Δ of additional hypotheses in the subdeduction of B.

As for NJp, a modal λ-calculus is then obtained by decorating IEL^{-}-deductions with proof names.
The λ-term corresponding to $\Box\mathcal{I}$ is indeed ruled by the single constructor:

$$\frac{\Gamma_1 \vdash f_1 : \Box A_1 \qquad \cdots \qquad \Gamma_n \vdash f_n : \Box A_n \qquad x_1 : A_1, \cdots, x_n : A_n, \Delta \vdash g : B}{\Gamma_1, \cdots, \Gamma_n, \Delta \vdash (\mathrm{box}[x_1, \cdots, x_n].\, g\, \mathrm{with}\, f_1, \cdots, f_n) \ : \Box B}$$

$$\dfrac{\dfrac{\begin{array}{c}\Gamma\\ \vdots\\ \Box\vec{A}\end{array} \qquad \begin{array}{c}[\vec{A}]^1, \vec{C}\\ \vdots\\ B\end{array}}{\Box B}\,\Box\mathcal{I}{:}1 \qquad \begin{array}{c}\Delta\\ \vdots\\ \Box\vec{D}\end{array} \qquad \begin{array}{c}[B, \vec{D}]^2, \vec{E}\\ \vdots\\ F\end{array}}{\Box F}\,\Box\mathcal{I}{:}2 \quad \overset{\rho\Box}{\rightsquigarrow} \quad \dfrac{\begin{array}{c}\Gamma\\ \vdots\\ \Box\vec{A}\end{array} \qquad \begin{array}{c}\Delta\\ \vdots\\ \Box\vec{D}\end{array} \qquad \begin{array}{c}\begin{array}{c}[\vec{A}]^1, \vec{C}\\ \vdots\\ B\end{array} \quad [\vec{D}]^1, \vec{E}\\ \vdots\\ F\end{array}}{\Box F}\,\Box\mathcal{I}{:}1$$

Rewritings

$\rho\Box$

$$\cfrac{\cfrac{\begin{matrix}\Gamma\\ \vdots\\ \Box\vec{A}\end{matrix}\qquad \begin{matrix}[\vec{A}]^1,\vec{C}\\ \vdots\\ B\end{matrix}}{\Box B}\,\Box\mathcal{I}{:}1 \qquad \begin{matrix}\Delta\\ \vdots\\ \Box\vec{D}\end{matrix}\qquad \begin{matrix}[B,\vec{D}]^2,\vec{E}\\ \vdots\\ F\end{matrix}}{\Box F}\,\Box\mathcal{I}{:}2 \quad \overset{\rho\Box}{\rightsquigarrow} \quad \cfrac{\begin{matrix}\Gamma\\ \vdots\\ \Box\vec{A}\end{matrix}\qquad \begin{matrix}\Delta\\ \vdots\\ \Box\vec{D}\end{matrix}\qquad \begin{matrix}\begin{matrix}[\vec{A}]^1,\vec{C}\\ \vdots\\ B\end{matrix}\quad [\vec{D}]^1,\vec{E}\\ \vdots\\ F\end{matrix}}{\Box F}\,\Box\mathcal{I}{:}1$$

Rewritings

$\pi_\Box$

$$\dfrac{\begin{matrix}\Gamma_1\\ \vdots\, f_1\\ \Box A_1\end{matrix}\quad\cdots\quad \dfrac{\begin{matrix}\vdots\, t\end{matrix}\qquad \begin{matrix}\Gamma_i\\ \vdots\, f_i\\ \Box A_i\end{matrix}}{\Box A_i}\,\mathcal{E}+ \quad\cdots\quad \begin{matrix}\Gamma_n\\ \vdots\, f_n\\ \Box A_n\end{matrix}\qquad \begin{matrix}[A_1,\cdots,A_n],\Delta\\ \vdots\, g\\ B\end{matrix}}{\Box B}\,\Box\mathcal{I} \quad \overset{\pi_\Box}{\rightsquigarrow}$$

$$\overset{\pi_\Box}{\rightsquigarrow}\quad \dfrac{\begin{matrix}\vdots\, t\end{matrix}\qquad \dfrac{\begin{matrix}\Gamma_1\\ \vdots\, f_1\\ \Box A_1\end{matrix}\quad\cdots\quad \begin{matrix}\Gamma_i\\ \vdots\, f_i\\ \Box A_i\end{matrix}\quad\cdots\quad \begin{matrix}\Gamma_n\\ \vdots\, f_n\\ \Box A_n\end{matrix}\qquad \begin{matrix}[A_1,\cdots,A_n],\Delta\\ \vdots\, g\\ B\end{matrix}}{\Box B}\,\Box\mathcal{I}}{\Box B}\,\mathcal{E}+$$

where $\mathcal{E}+$ is $\vee\mathcal{E}$ or $\bot_J$.

Rewritings

$\pi_{\Box}$

$$
\dfrac{
\begin{array}{c}\Gamma_1\\ \vdots\, f_1\\ \Box A_1\end{array}
\quad \cdots \quad
\dfrac{\begin{array}{c}\vdots\, t\end{array} \quad \begin{array}{c}\Gamma_i\\ \vdots\, f_i\\ \Box A_i\end{array}}{\Box A_i}\,\mathcal{E}+
\quad \cdots \quad
\begin{array}{c}\Gamma_n\\ \vdots\, f_n\\ \Box A_n\end{array}
\quad
\begin{array}{c}[A_1, \cdots, A_n], \Delta\\ \vdots\, g\\ B\end{array}
}{\Box B}\,\Box\mathcal{I}
\quad \overset{\pi_{\Box}}{\rightsquigarrow}
$$

$$
\overset{\pi_{\Box}}{\rightsquigarrow} \quad
\dfrac{
\begin{array}{c}\vdots\, t\end{array}
\quad
\dfrac{
\begin{array}{c}\Gamma_1\\ \vdots\, f_1\\ \Box A_1\end{array}
\quad \cdots \quad
\begin{array}{c}\Gamma_i\\ \vdots\, f_i\\ \Box A_i\end{array}
\quad \cdots \quad
\begin{array}{c}\Gamma_n\\ \vdots\, f_n\\ \Box A_n\end{array}
\quad
\begin{array}{c}[A_1, \cdots, A_n], \Delta\\ \vdots\, g\\ B\end{array}
}{\Box B}\,\Box\mathcal{I}
}{\Box B}\,\mathcal{E}+
$$

where $\mathcal{E}+$ is $\vee\mathcal{E}$ or $\bot_J$.

Normalisation

Theorem (PB 2022)

Deductions in IEL$^-$ *strongly normalise w.r.t. the standard rewriting system for* NJp *extended by* $\rho_\Box + \pi_\Box$.

Proof Sketch. We define a translation $\langle - \rangle$ from our modal λ-calculus to simple type theory with products, sums, unit and empty types as follows:

$$
\begin{array}{lcl}
\langle \bot \rangle & := & \bot \\
\langle \top \rangle & := & \top \\
\langle p \rangle & := & p \\
\langle A \to B \rangle & := & \langle A \rangle \to \langle B \rangle \\
\langle A \wedge B \rangle & := & \langle A \rangle \wedge \langle B \rangle \\
\langle A \vee B \rangle & := & \langle A \rangle \vee \langle B \rangle \\
\langle \Box A \rangle & := & \langle A \rangle \vee q
\end{array}
$$

where q is an arbitrary atom.

Theorem (PB 2022)

Deductions in IEL^{-} *strongly normalise w.r.t. the standard rewriting system for* NJp *extended by* $\rho_\Box + \pi_\Box$.

Proof Sketch. We define a translation $\langle - \rangle$ from our modal λ-calculus to simple type theory with products, sums, unit and empty types as follows:

$$
\begin{array}{lcl}
\langle \bot \rangle & := & \bot \\
\langle \top \rangle & := & \top \\
\langle p \rangle & := & p \\
\langle A \to B \rangle & := & \langle A \rangle \to \langle B \rangle \\
\langle A \wedge B \rangle & := & \langle A \rangle \wedge \langle B \rangle \\
\langle A \vee B \rangle & := & \langle A \rangle \vee \langle B \rangle \\
\langle \Box A \rangle & := & \langle A \rangle \vee q
\end{array}
$$

where q is an arbitrary atom.

Theorem (PB 2022)

Deductions in IEL$^-$ *strongly normalise w.r.t. the standard rewriting system for* NJp *extended by* $\rho_\Box + \pi_\Box$.

Proof Sketch.

$$\dfrac{\begin{matrix}\Gamma_1\\ \vdots\, f_1\\ \Box A_1\end{matrix} \qquad \begin{matrix}\Gamma_2\\ \vdots\, f_2\\ \Box A_2\end{matrix} \qquad \begin{matrix}[A_1, A_2], \Delta\\ \vdots\, g\\ B\end{matrix}}{\Box B}\,\Box\mathcal{I} \quad \xmapsto{\langle\ \rangle}$$

$$\xmapsto{\langle\ \rangle} \dfrac{\begin{matrix}\langle\Gamma_2\rangle\\ \vdots\, \langle f_2\rangle\\ \langle A_2\rangle \vee q\end{matrix} \qquad \dfrac{\begin{matrix}\langle\Gamma_1\rangle\\ \vdots\, \langle f_1\rangle\\ \langle A_1\rangle \vee q\end{matrix} \qquad \dfrac{\begin{matrix}[\langle A_1\rangle]^1, [\langle A_2\rangle]^2, \langle\Delta\rangle\\ \vdots\, \langle g\rangle\\ \langle B\rangle\end{matrix}}{\langle B\rangle \vee q}\,\vee\mathcal{I}_1 \qquad \dfrac{[q]^1}{\langle B\rangle \vee q}\,\vee\mathcal{I}_2}{\langle B\rangle \vee q}\,\vee\mathcal{E}{:}1 \qquad \dfrac{[q]^2}{\langle B\rangle \vee q}\,\vee\mathcal{I}_2}{\langle B\rangle \vee q}\,\vee\mathcal{E}{:}2$$

$\boxtimes$

The real importance of cut-free proofs is not the elimination of cuts per se, but rather that such proofs obey the subformula principle.

(Smullyan 1968)

Theorem (Subformula principle, PB 2022)

Every formula B occurring in a normal IEL$^-$*-deduction f of A from assumptions Γ is a subformula of A or of some formula in Γ.*

Proof Sketch.

After (Prawitz 1971):

The real importance of cut-free proofs is not the elimination of cuts per se, but rather that such proofs obey the subformula principle.

(Smullyan 1968)

Theorem (Subformula principle, PB 2022)

Every formula B occurring in a normal IEL^--deduction f of A from assumptions Γ is a subformula of A or of some formula in Γ.

Proof Sketch.

After (Prawitz 1971):

The real importance of cut-free proofs is not the elimination of cuts per se, but rather that such proofs obey the subformula principle.

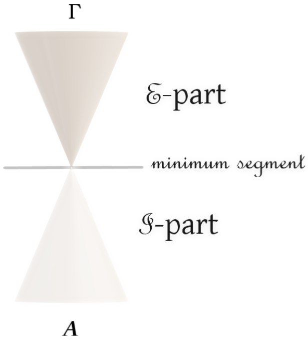
(Smullyan 1968)

Theorem (Subformula principle, PB 2022)

Every formula B occurring in a normal IEL^--deduction f of A from assumptions Γ is a subformula of A or of some formula in Γ.

Proof Sketch.

After (Prawitz 1971):



☒

Lemma

The following hold:

- *The reflection rule is admissible in* IEL^-.
- IEL^- *satisfies the disjunction property.*
- IEL^- *is* $\Box$*-prime.*
- *If* $\mathsf{IEL}^- \vdash \Box(A \vee B)$*, then* $\mathsf{IEL}^- \vdash \Box A$ *or* $\mathsf{IEL}^- \vdash \Box B$*.*
- IEL^- *is consistent.*
- IEL^- *is decidable.*

Proof.

These properties are proven in (Artemov and Protopopescu 2016) by semantic arguments.

Here we can rely on syntactic considerations involving (canonicity and) the subformula property of the calculus.

⊠

Computational trinitarism

Logic	Type Theory	Category Theory
proposition	type	object
proof	term	arrow
theorem	inhabitant	element-arrow
conjunction	product type	product
true	unit type	terminal object
implication	function type	exponential
disjunction	sum type	(weak) coproduct
false	empty type	(weak) initial object

Proof theoretic semantics for IEL^-

(PB 2021)

An IEL^--category is a bi-CCC $\mathscr{C}$, equipped with a pointed monoidal endofunctor $\Box : \mathscr{C} \to \mathscr{C}$ whose point is monoidal.

$\Rightarrow$ algebraic semantics of deductions in IEL^-

But what kind of identity of proofs does an IEL^--category capture?

$$
\dfrac{\begin{array}{c}\Gamma\\ \vdots\\ \Box A\end{array} \qquad [A]}{\Box A}\,\Box\mathcal{I}
\quad \overset{\eta_\Box}{\rightsquigarrow} \quad
\begin{array}{c}\Gamma\\ \vdots\\ \Box A\end{array}
$$

Theorem (PB 2022)

IEL^-*-deductions strongly normalise w.r.t. the standard rewriting system for* NJp *extended by* $\rho_\Box + \eta_\Box$.

$$
\dfrac{\begin{array}{c}\Gamma \\ \vdots \\ \Box A\end{array} \qquad [A]}{\Box A}\,\Box\mathcal{I}
\quad \overset{\eta_\Box}{\rightsquigarrow} \quad
\begin{array}{c}\Gamma \\ \vdots \\ \Box A\end{array}
$$

Theorem (PB 2022)

IEL$^-$*-deductions strongly normalise w.r.t. the standard rewriting system for* NJp *extended by* $\rho_\Box + \eta_\Box$.

Categorical interpretation

(PB 2021)

Theorem (Soundness)

Let $\mathscr{C}$ be an IEL^--category. Then there is a canonical interpretation $[\![-]\!]$ of IEL^- in $\mathscr{C}$ such that

- *a formula A is mapped to a $\mathscr{C}$-object $[\![A]\!]$;*
- *a deduction f of $A_1, \cdots, A_n \vdash_{\mathsf{IEL}^-} B$ is mapped to an arrow*

$$[\![f]\!] : [\![A_1]\!] \times \cdots \times [\![A_n]\!] \to [\![B]\!];$$

- *for any two deductions f and g which are equal modulo standard rewritings extended by $\rho_\Box + \eta_\Box$, we have $[\![f]\!] = [\![g]\!]$.*

Theorem (Completeness)

If the interpretation of two IEL^--deductions is equal in all IEL^--categories, then the two deductions are equal modulo standard+$\rho_\Box + \eta_\Box$-rewritings.

Intuitionistic factivity of knowledge

(Artemov and Protopopescu 2016)

Knowledge $\simeq$ Justified True Belief

Axioms

1. Axioms of propositional intuitionistic logic;
2. $\Box(A \to B) \to \Box A \to \Box B$; (K-scheme)
3. $A \to \Box A$. (co-reflection)
4. $\Box A \to \neg\neg A$ (intuitionistic factivity of knowledge)

Rules

$$\frac{A \to B \qquad A}{B}\ MP$$

A model for $\mathbb{IEL}$ is a model for $\mathbb{IEL}^-$ $\langle W, \leq, v, E\rangle$, where the relation E satisfies the seriality condition

- for any $x \in W$, there exists a $y \in W$ such that xEy.

Modal adequacy for $\mathbb{IEL}$

$\mathbb{IEL}$ is sound and complete w.r.t. $\mathbb{IEL}$ relational frames.

Natural deduction for intuitionistic knowledge

(PB 2022)

Let IEL be the system extending the natural deduction calculus IEL^{-} by the following elimination rule:

$$\dfrac{\begin{array}{c}\Gamma\\ \vdots\\ \Box A\end{array} \qquad \begin{array}{c}[A], \Delta\\ \vdots\\ \bot\end{array}}{\bot}\ \Box\mathcal{E}$$

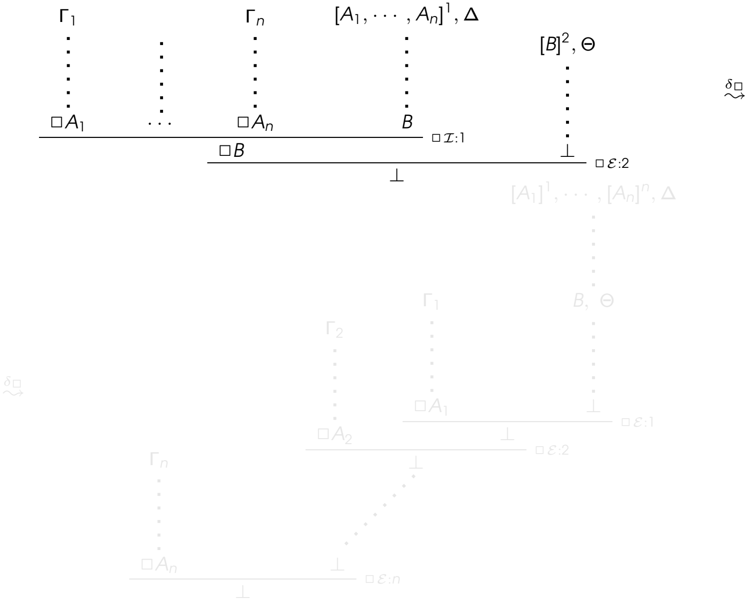
where Γ and Δ are *multisets of formulas*, and A is discharged by $\Box\mathcal{E}$.

It is straightforward to extend the modal λ-calculus for IEL^- by decorating $\Box\mathcal{E}$ with proof names.
The λ-term corresponding to $\Box\mathcal{E}$ gives the eliminator for modal terms, as expected:

$$\frac{\Gamma \vdash f : \Box A \qquad x : A, \Delta \vdash g : \bot}{\Gamma, \Delta \vdash \ (\mathsf{unbox}\, f\, \mathsf{with}\, x.g) \ : \bot}$$

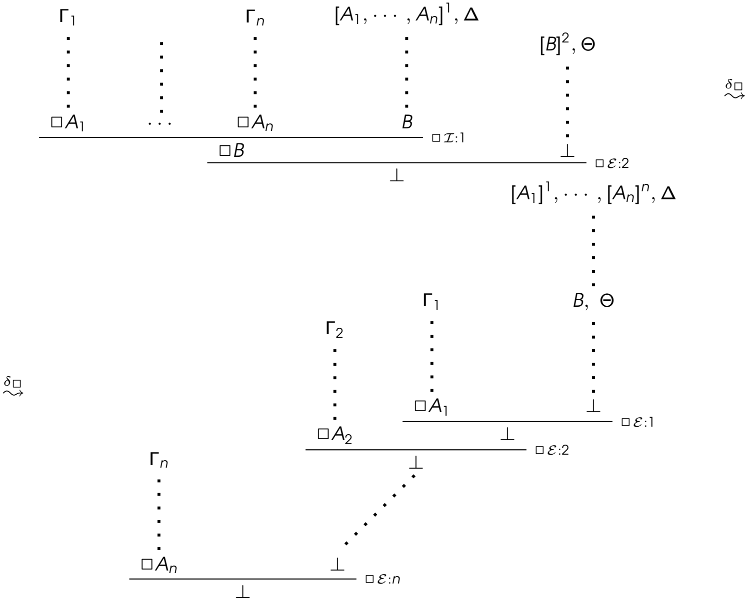
Rewritings

$\delta_\Box$



Rewritings

$\delta_\Box$



Normalisation

Theorem (PB 2022)

Deductions in IEL *strongly normalise w.r.t. the rewriting system for* NJp *extended by* $\rho_\Box + \pi_\Box + \delta_\Box$.

Proof Sketch.

Tweak the translation $\langle - \rangle$ in the proof of strong normalisation for IEL^- as follows:

$$
\begin{array}{lcl}
\langle \bot \rangle & := & \bot \\
\langle \top \rangle & := & \top \\
\langle p \rangle & := & p \\
\langle A \to B \rangle & := & \langle A \rangle \to \langle B \rangle \\
\langle A \wedge B \rangle & := & \langle A \rangle \wedge \langle B \rangle \\
\langle A \vee B \rangle & := & \langle A \rangle \vee \langle B \rangle \\
\langle \Box A \rangle & := & \langle A \rangle \vee \bot
\end{array}
$$

$$
\dfrac{\begin{array}{c}\Gamma \\ \vdots\, f \\ \Box A\end{array} \qquad \begin{array}{c}[A], \Delta \\ \vdots\, g \\ \bot\end{array}}{\bot}\,\Box\mathcal{E}
\quad \overset{\langle\,\rangle}{\longmapsto} \quad
\dfrac{\begin{array}{c}\langle \Gamma \rangle \\ \vdots\, \langle f \rangle \\ \langle A \rangle \vee \bot\end{array} \qquad \begin{array}{c}[\langle A \rangle]^1, \langle \Delta \rangle \\ \vdots\, \langle g \rangle \\ \bot\end{array} \qquad [\bot]^1}{\bot}\,\vee\mathcal{E}{:}1
$$

Theorem (PB 2022)

Deductions in IEL *strongly normalise w.r.t. the rewriting system for* NJp *extended by* $\rho_\Box + \pi_\Box + \delta_\Box$.

Proof Sketch.

Tweak the translation $\langle - \rangle$ in the proof of strong normalisation for IEL^- as follows:

$$
\begin{array}{lcl}
\langle \bot \rangle & := & \bot \\
\langle \top \rangle & := & \top \\
\langle p \rangle & := & p \\
\langle A \to B \rangle & := & \langle A \rangle \to \langle B \rangle \\
\langle A \wedge B \rangle & := & \langle A \rangle \wedge \langle B \rangle \\
\langle A \vee B \rangle & := & \langle A \rangle \vee \langle B \rangle \\
\langle \Box A \rangle & := & \langle A \rangle \vee \bot
\end{array}
$$

$$
\dfrac{\begin{array}{c}\Gamma \\ \vdots\, f \\ \Box A\end{array} \qquad \begin{array}{c}[A], \Delta \\ \vdots\, g \\ \bot\end{array}}{\bot}\,\Box\mathcal{E}
\quad \stackrel{\langle\,\rangle}{\longmapsto} \quad
\dfrac{\begin{array}{c}\langle\Gamma\rangle \\ \vdots\, \langle f\rangle \\ \langle A\rangle \vee \bot\end{array} \qquad \begin{array}{c}[\langle A\rangle]^1, \langle\Delta\rangle \\ \vdots\, \langle g\rangle \\ \bot\end{array} \qquad [\bot]^1}{\bot}\,\vee\mathcal{E}{:}1
$$

$\boxtimes$

All the results about IEL^- are modularly extended to IEL:

Theorem (PB 2022)

The following hold:

- *The subformula property holds for normal* IEL*-deductions.*
- *In any normal* IEL*-deduction of A, the last rule applied is the introduction rule for the main connective of A.*
- *The reflection rule is admissible in* IEL.
- IEL *satisfies the disjunction property.*
- IEL *is* $\Box$*-prime.*
- *If* $\mathsf{IEL} \vdash \Box(A \vee B)$*, then* $\mathsf{IEL} \vdash \Box A$ *or* $\mathsf{IEL} \vdash \Box B$.
- IEL *is consistent.*
- IEL *is decidable.*
- $\mathsf{IEL}^- \subsetneq \mathsf{IEL}$.
- *For* $\mathsf{L} \in \{\mathsf{IEL}^-, \mathsf{IEL}\}$, $\mathsf{L} \nvdash \Box(A \vee B) \to \Box A \vee \Box B$.

Logics for provability $\simeq$ modal systems capturing the abstract and structural properties of the provability predicate used in Gödel's incompleteness results

For classical arithmetical theories, Gödel-Löb logic does the job.

For intuitionistic arithmetical theories, the situation is less clear,[2] but intuitionistic provability logics have become relevant tools for studying fix-point and guarded recursion operators.

[2] Refer however to (Mojtahedi 2022) for a promising candidate for provability in Heyting arithmetic.

Provability logics

Logics for provability $\simeq$ modal systems capturing the abstract and structural properties of the provability predicate used in Gödel's incompleteness results

For classical arithmetical theories, Gödel-Löb logic does the job.

For intuitionistic arithmetical theories, the situation is less clear,[2] but intuitionistic provability logics have become relevant tools for studying fix-point and guarded recursion operators.

[2]Refer however to (Mojtahedi 2022) for a promising candidate for provability in Heyting arithmetic.

Intuitionistic strong Löb logic

(Visser and Zoethout 2018)

Axioms

1. Axioms of propositional intuitionistic logic;
2. $\Box(A \to B) \to \Box A \to \Box B$; (K-scheme)
3. $A \to \Box A$. (co-reflection)
4. $\Box(\Box A \to A) \to \Box A$ (GL-scheme)

Rules

$$\frac{A \to B \qquad A}{B}\, MP$$

A model for $\mathbb{ISL}$ is a quadruple $\langle W, \leq, v, R\rangle$ where

- $\langle W, \leq, v\rangle$ is a standard model for intuitionistic propositional logic;
- R is a binary relation on W such that:
  - if xRy, then $x \leq y$; and
  - if $x \leq y$ and yRz, then xRz
  - R is transitive;
  - R is Noetherian;
- v extends to a forcing relation $\Vdash$ such that
  - $x \Vdash \Box A$ iff $y \Vdash A$ for all y such that xRy.

Modal adequacy for $\mathbb{ISL}$

$\mathbb{ISL}$ is sound and complete w.r.t. $\mathbb{ISL}$ relational frames.

Natural deduction for iSL

(PB 2022)

Let ISL be the system extending the natural deduction calculus IEL^- by the following elimination rule:

$$\dfrac{\begin{array}{c}\Gamma, [\Box A]^\star \\ \vdots \\ A\end{array} \qquad \begin{array}{c}[A]^\star, \Delta \\ \vdots \\ C\end{array}}{C}\ \Box\mathcal{E}{:}\star$$

where Γ and Δ are *multisets of formulas*, and $\Box\mathcal{E}$ allows both multiple and vacuous discharge.

The λ-term corresponding to $\Box\mathcal{E}$ gives the eliminator for modal *variables*:

$$\frac{x : \Box A, \Gamma \vdash f : A \qquad y : A, \Delta \vdash g : C}{\Gamma, \Delta \vdash \ (\text{löb}\, x.f \text{ with } y.g) \ : C}$$

$$\dfrac{\dfrac{\begin{matrix}\Gamma_1\\ \vdots\\ \Box A_1\end{matrix} \quad \begin{matrix}\vdots\\ \cdots\end{matrix} \quad \begin{matrix}\Gamma_n\\ \vdots\\ \Box A_n\end{matrix} \qquad \begin{matrix}[A_1,\cdots,A_n]^1, \Delta, [\Box\Box B]^2\\ \vdots\\ B\end{matrix}}{\Box B}\,\Box\mathcal{I}{:}1 \qquad \begin{matrix}[\Box B]^2, \Theta\\ \vdots\\ C\end{matrix}}{C}\,\Box\mathcal{E}{:}2$$

$$\overset{o^1_\Box}{\rightsquigarrow}$$

$$\begin{matrix}\dfrac{\begin{matrix}\Gamma_1\\ \vdots\\ \Box A_1\end{matrix} \quad \begin{matrix}\vdots\\ \cdots\end{matrix} \quad \begin{matrix}\Gamma_n\\ \vdots\\ \Box A_n\end{matrix} \qquad \dfrac{\begin{matrix}\dfrac{[\Box B]^1}{\Box\Box B}\,\Box\mathcal{I} \quad [A_1,\cdots,A_n]^2, \Delta\\ \vdots\\ B\end{matrix} \qquad [B]^1}{B}\,\Box\mathcal{E}{:}1}{\Box B}\,\Box\mathcal{I}{:}2\\ \vdots\\ C\end{matrix}$$

$$
\dfrac{
\dfrac{
\begin{matrix}\Gamma_1\\ \vdots \\ \Box A_1\end{matrix}
\quad
\begin{matrix}\\ \vdots \\ \cdots\end{matrix}
\quad
\begin{matrix}\Gamma_n\\ \vdots \\ \Box A_n\end{matrix}
\quad
\begin{matrix}[A_1, \cdots, A_n]^1, \Delta, [\Box\Box B]^2\\ \vdots \\ B\end{matrix}
}{\Box B}\,\Box\mathcal{I}{:}1
\qquad
\begin{matrix}[\Box B]^2, \Theta\\ \vdots \\ C\end{matrix}
}{C}\,\Box\mathcal{E}{:}2
\qquad \overset{o^1_\Box}{\leadsto}
$$

$$
\overset{o^1_\Box}{\leadsto}\qquad
\begin{matrix}
\dfrac{
\begin{matrix}\Gamma_1\\ \vdots \\ \Box A_1\end{matrix}
\quad
\begin{matrix}\\ \vdots \\ \cdots\end{matrix}
\quad
\begin{matrix}\Gamma_n\\ \vdots \\ \Box A_n\end{matrix}
\quad
\begin{matrix}\dfrac{[\Box B]^1}{\Box\Box B}\,\Box\mathcal{I}\\ \\ \\ \end{matrix}
\quad
\dfrac{\begin{matrix}[A_1, \cdots, A_n]^2, \Delta\\ \vdots \\ B\end{matrix}\qquad [B]^1}{B}\,\Box\mathcal{E}{:}1
}{\Box B}\,\Box\mathcal{I}{:}2
\\ \vdots \\ C
\end{matrix}
$$

$$\dfrac{\dfrac{\begin{matrix}\Gamma_1\\ \vdots\\ \Box A_1\end{matrix} \quad \begin{matrix}[\Box\Box B]^2,\Gamma_i\\ \vdots \\ \Box A_i\end{matrix} \quad \begin{matrix}\Gamma_n\\ \vdots\\ \Box A_n\end{matrix} \quad \begin{matrix}[A_1,\cdots,A_n]^1,\Delta\\ \vdots\\ B\end{matrix}}{\Box B}\,\Box\mathcal{I}{:}1 \qquad \begin{matrix}[\Box B]^2,\Theta\\ \vdots\\ C\end{matrix}}{\Box B}\,\Box\mathcal{E}{:}2 \qquad \overset{o^2_\Box}{\rightsquigarrow}$$

$$\overset{o^2_\Box}{\rightsquigarrow} \quad \dfrac{\dfrac{\begin{matrix}[\Box\Box B]^3,\Gamma_i\\ \vdots\\ \Box A_i\end{matrix} \qquad \dfrac{\begin{matrix}\Gamma_1\\ \vdots\\ \Box A_1\end{matrix} \quad [\Box A_i]^2 \quad \begin{matrix}\Gamma_n\\ \vdots\\ \Box A_n\end{matrix} \quad \begin{matrix}[A_1,\cdots,A_n]^1,\Delta\\ \vdots\\ B\end{matrix}}{\Box B}\,\Box\mathcal{I}{:}1}{\Box B}\,\Box\mathcal{E}{:}2 \qquad \begin{matrix}[\Box B]^3\\ \vdots\\ C\end{matrix}}{C}\,\Box\mathcal{E}{:}3$$

Rewritings

$o^2_\Box$

$$\cfrac{\cfrac{\begin{matrix}\Gamma_1\\ \vdots\\ \Box A_1\end{matrix} \quad \begin{matrix}[\Box\Box B]^2,\Gamma_i\\ \vdots \\ \Box A_i\end{matrix} \quad \begin{matrix}\Gamma_n\\ \vdots\\ \Box A_n\end{matrix}\quad \begin{matrix}[A_1,\cdots,A_n]^1,\Delta\\ \vdots\\ B\end{matrix}}{\Box B}\,\Box\mathcal{I}{:}1 \qquad \begin{matrix}[\Box B]^2,\Theta\\ \vdots\\ C\end{matrix}}{\Box B}\,\Box\mathcal{E}{:}2 \qquad \overset{o^2_\Box}{\rightsquigarrow}$$

$$\overset{o^2_\Box}{\rightsquigarrow} \qquad \cfrac{\cfrac{\begin{matrix}[\Box\Box B]^3,\Gamma_i\\ \vdots \\ \Box A_i\end{matrix} \quad \cfrac{\begin{matrix}\Gamma_1\\ \vdots\\ \Box A_1\end{matrix} \quad [\Box A_i]^2 \quad \begin{matrix}\Gamma_n\\ \vdots\\ \Box A_n\end{matrix}\quad \begin{matrix}[A_1,\cdots,A_n]^1,\Delta\\ \vdots\\ B\end{matrix}}{\Box B}\,\Box\mathcal{I}{:}1}{\Box B}\,\Box\mathcal{E}{:}2 \qquad \begin{matrix}[\Box B]^3\\ \vdots\\ C\end{matrix}}{C}\,\Box\mathcal{E}{:}3$$

Normalisation

Theorem (PB 2022)

Deductions in ISL *strongly normalise w.r.t. the rewriting system for* NJp *extended by* $\rho_\Box + \pi_\Box + o^1_\Box + o^2_\Box$.

Proof Sketch.

Tweak the translation $\langle - \rangle$ in the proof of strong normalisation for IEL^- as follows:

$$
\begin{array}{lcl}
\langle \bot \rangle & := & \bot \\
\langle \top \rangle & := & \top \\
\langle p \rangle & := & p \\
\langle A \to B \rangle & := & \langle A \rangle \to \langle B \rangle \\
\langle A \wedge B \rangle & := & \langle A \rangle \wedge \langle B \rangle \\
\langle A \vee B \rangle & := & \langle A \rangle \vee \langle B \rangle \\
\langle \Box A \rangle & := & \langle A \rangle \vee \top
\end{array}
$$

$$
\dfrac{\begin{array}{c} \Gamma, [\Box A]^\star \\ \vdots\, f \\ A \end{array} \qquad \begin{array}{c} [A]^\star, \Delta \\ \vdots\, g \\ C \end{array}}{C}\ \Box\mathcal{E}{:}\star
\quad \overset{\langle\ \rangle}{\longmapsto} \quad
\begin{array}{c}
\langle \Gamma \rangle \quad \dfrac{\dfrac{}{\top}\ \top_I}{\langle A \rangle \vee \top}\ \mathrm{in}_2 \\
\vdots\, \langle f \rangle \\
\langle A \rangle \qquad\qquad \langle \Delta \rangle \\
\vdots\, \langle g \rangle \\
\langle C \rangle
\end{array}
$$

Theorem (PB 2022)

Deductions in ISL *strongly normalise w.r.t. the rewriting system for* NJp *extended by* $\rho_\Box + \pi_\Box + o^1_\Box + o^2_\Box$.

Proof Sketch.

Tweak the translation $\langle - \rangle$ in the proof of strong normalisation for IEL^- as follows:

$$
\begin{array}{lcl}
\langle \bot \rangle & := & \bot \\
\langle \top \rangle & := & \top \\
\langle p \rangle & := & p \\
\langle A \to B \rangle & := & \langle A \rangle \to \langle B \rangle \\
\langle A \wedge B \rangle & := & \langle A \rangle \wedge \langle B \rangle \\
\langle A \vee B \rangle & := & \langle A \rangle \vee \langle B \rangle \\
\langle \Box A \rangle & := & \langle A \rangle \vee \top
\end{array}
$$

$$
\dfrac{\begin{matrix}\Gamma, [\Box A]^\star \\ \vdots\, f \\ A\end{matrix} \qquad \begin{matrix}[A]^\star, \Delta \\ \vdots\, g \\ C\end{matrix}}{C}\,\Box\mathcal{E}{:}\star
\qquad \overset{\langle\ \rangle}{\longmapsto} \qquad
\begin{matrix}
\langle \Gamma \rangle \qquad \dfrac{\dfrac{}{\top}\,\top_I}{\langle A \rangle \vee \top}\,\mathrm{in}_2 \\
\vdots\, \langle f \rangle \\
\langle A \rangle \qquad\qquad \langle \Delta \rangle \\
\vdots\, \langle g \rangle \\
\langle C \rangle
\end{matrix}
$$

$\boxtimes$

$$\dfrac{\dfrac{\begin{matrix}\Gamma\\ \vdots\\ A\end{matrix} \qquad \begin{matrix}[A]^{\star}, \Delta\\ \vdots\\ \Box B\end{matrix}}{\Box B}\,\Box\mathcal{E}{:}\star \qquad \begin{matrix}[B]^{1}, \Theta\\ \vdots\\ C\end{matrix}}{\Box C}\,\Box\mathcal{I}{:}1 \quad \overset{\kappa_{\Box}}{\rightsquigarrow} \quad \dfrac{\begin{matrix}\Gamma\\ \vdots\\ A\end{matrix} \qquad \dfrac{\begin{matrix}[A]^{\star}, \Delta\\ \vdots\\ \Box B\end{matrix} \qquad \begin{matrix}[B]^{1}, \Theta\\ \vdots\\ C\end{matrix}}{\Box C}\,\Box\mathcal{I}{:}1}{\Box C}\,\Box\mathcal{E}{:}\star$$

$$\dfrac{\dfrac{\begin{matrix}\Gamma\\ \vdots\\ A\end{matrix} \qquad \begin{matrix}[A]^{\star}, \Delta\\ \vdots\\ C\end{matrix}}{C}\,\Box\mathcal{E}{:}\star \qquad \begin{matrix}\vdots\end{matrix}}{D}\,\mathcal{E}-prop \quad \overset{\kappa_{\Box}}{\rightsquigarrow} \quad \dfrac{\begin{matrix}\Gamma\\ \vdots\\ A\end{matrix} \qquad \dfrac{\begin{matrix}[A]^{\star}, \Delta\\ \vdots\\ C\end{matrix} \qquad \begin{matrix}\vdots\end{matrix}}{D}\,\mathcal{E}-prop}{C}\,\Box\mathcal{E}{:}\star$$

Notice that the translation used in proving strong normalisation preserves the rewritings of $\kappa_{\Box}$ too, so that ISL strongly normalises w.r.t. the extended system.

Analyticity

Further rewritings $\kappa_\Box$

$$\dfrac{\dfrac{\begin{matrix}\Gamma\\ \vdots\\ A\end{matrix} \quad \begin{matrix}[A]^\star,\Delta\\ \vdots\\ \Box B\end{matrix}}{\Box B}\,\Box\mathcal{E}{:}\star \qquad \begin{matrix}[B]^1,\Theta\\ \vdots\\ C\end{matrix}}{\Box C}\,\Box\mathcal{I}{:}1 \quad \overset{\kappa_\Box}{\rightsquigarrow} \quad \dfrac{\begin{matrix}\Gamma\\ \vdots\\ A\end{matrix} \qquad \dfrac{\begin{matrix}[A]^\star,\Delta\\ \vdots\\ \Box B\end{matrix} \quad \begin{matrix}[B]^1,\Theta\\ \vdots\\ C\end{matrix}}{\Box C}\,\Box\mathcal{I}{:}1}{\Box C}\,\Box\mathcal{E}{:}\star$$

$$\dfrac{\dfrac{\begin{matrix}\Gamma\\ \vdots\\ A\end{matrix} \quad \begin{matrix}[A]^\star,\Delta\\ \vdots\\ C\end{matrix}}{C}\,\Box\mathcal{E}{:}\star \qquad \begin{matrix}\vdots\end{matrix}}{D}\,\mathcal{E}-prop \quad \overset{\kappa_\Box}{\rightsquigarrow} \quad \dfrac{\begin{matrix}\Gamma\\ \vdots\\ A\end{matrix} \qquad \dfrac{\begin{matrix}[A]^\star,\Delta\\ \vdots\\ C\end{matrix} \quad \begin{matrix}\vdots\end{matrix}}{D}\,\mathcal{E}-prop}{C}\,\Box\mathcal{E}{:}\star$$

Notice that the translation used in proving strong normalisation preserves the rewritings of $\kappa_\Box$ too, so that ISL strongly normalises w.r.t. the extended system.

$$
\dfrac{\dfrac{\begin{matrix}\Gamma\\ \vdots\\ A\end{matrix} \qquad \begin{matrix}[A]^\star, \Delta\\ \vdots\\ \Box B\end{matrix}}{\Box B}\,\Box\mathcal{E}{:}\star \qquad \begin{matrix}[B]^1, \Theta\\ \vdots\\ C\end{matrix}}{\Box C}\,\Box\mathcal{I}{:}1
\qquad \overset{\kappa_\Box}{\rightsquigarrow} \qquad
\dfrac{\begin{matrix}\Gamma\\ \vdots\\ A\end{matrix} \qquad \dfrac{\begin{matrix}[A]^\star, \Delta\\ \vdots\\ \Box B\end{matrix} \qquad \begin{matrix}[B]^1, \Theta\\ \vdots\\ C\end{matrix}}{\Box C}\,\Box\mathcal{I}{:}1}{\Box C}\,\Box\mathcal{E}{:}\star
$$

$$
\dfrac{\dfrac{\begin{matrix}\Gamma\\ \vdots\\ A\end{matrix} \qquad \begin{matrix}[A]^\star, \Delta\\ \vdots\\ C\end{matrix}}{C}\,\Box\mathcal{E}{:}\star \qquad \vdots}{D}\,\mathcal{E}-prop
\qquad \overset{\kappa_\Box}{\rightsquigarrow} \qquad
\dfrac{\begin{matrix}\Gamma\\ \vdots\\ A\end{matrix} \qquad \dfrac{\begin{matrix}[A]^\star, \Delta\\ \vdots\\ C\end{matrix} \qquad \vdots}{D}\,\mathcal{E}-prop}{C}\,\Box\mathcal{E}{:}\star
$$

Notice that the translation used in proving strong normalisation preserves the rewritings of $\kappa_\Box$ too, so that ISL strongly normalises w.r.t. the extended system.

$$\dfrac{\dfrac{\begin{matrix}\Gamma\\ \vdots\\ A\end{matrix} \qquad \begin{matrix}[A]^\star,\Delta\\ \vdots\\ \Box B\end{matrix}}{\Box B}\,\Box\mathcal{E}{:}\star \qquad \begin{matrix}[B]^1,\Theta\\ \vdots\\ C\end{matrix}}{\Box C}\,\Box\mathcal{I}{:}1 \qquad \overset{\kappa_\Box}{\rightsquigarrow} \qquad \dfrac{\begin{matrix}\Gamma\\ \vdots\\ A\end{matrix} \qquad \dfrac{\begin{matrix}[A]^\star,\Delta\\ \vdots\\ \Box B\end{matrix} \qquad \begin{matrix}[B]^1,\Theta\\ \vdots\\ C\end{matrix}}{\Box C}\,\Box\mathcal{I}{:}1}{\Box C}\,\Box\mathcal{E}{:}\star$$

$$\dfrac{\dfrac{\begin{matrix}\Gamma\\ \vdots\\ A\end{matrix} \qquad \begin{matrix}[A]^\star,\Delta\\ \vdots\\ C\end{matrix}}{C}\,\Box\mathcal{E}{:}\star \qquad \vdots}{D}\,\mathcal{E}-prop \qquad \overset{\kappa_\Box}{\rightsquigarrow} \qquad \dfrac{\begin{matrix}\Gamma\\ \vdots\\ A\end{matrix} \qquad \dfrac{\begin{matrix}[A]^\star,\Delta\\ \vdots\\ C\end{matrix} \qquad \vdots}{D}\,\mathcal{E}-prop}{C}\,\Box\mathcal{E}{:}\star$$

Notice that the translation used in proving strong normalisation preserves the rewritings of $\kappa_\Box$ too, so that ISL strongly normalises w.r.t. the extended system.

Analyticity

Further rewritings $\kappa_\Box$

$$\dfrac{\dfrac{\begin{matrix}\Gamma\\ \vdots\\ A\end{matrix} \quad \begin{matrix}[A]^\star,\Delta\\ \vdots\\ \Box B\end{matrix}}{\Box B}\,\Box\mathcal{E}{:}\star \qquad \begin{matrix}[B]^1,\Theta\\ \vdots\\ C\end{matrix}}{\Box C}\,\Box\mathcal{I}{:}1 \qquad \overset{\kappa_\Box}{\rightsquigarrow} \qquad \dfrac{\begin{matrix}\Gamma\\ \vdots\\ A\end{matrix} \quad \dfrac{\begin{matrix}[A]^\star,\Delta\\ \vdots\\ \Box B\end{matrix} \quad \begin{matrix}[B]^1,\Theta\\ \vdots\\ C\end{matrix}}{\Box C}\,\Box\mathcal{I}{:}1}{\Box C}\,\Box\mathcal{E}{:}\star$$

$$\dfrac{\dfrac{\begin{matrix}\Gamma\\ \vdots\\ A\end{matrix} \quad \begin{matrix}[A]^\star,\Delta\\ \vdots\\ C\end{matrix}}{C}\,\Box\mathcal{E}{:}\star \qquad \vdots}{D}\,\mathcal{E}-prop \qquad \overset{\kappa_\Box}{\rightsquigarrow} \qquad \dfrac{\begin{matrix}\Gamma\\ \vdots\\ A\end{matrix} \quad \dfrac{\begin{matrix}[A]^\star,\Delta\\ \vdots\\ C\end{matrix} \quad \vdots}{D}\,\mathcal{E}-prop}{C}\,\Box\mathcal{E}{:}\star$$

Notice that the translation used in proving strong normalisation preserves the rewritings of $\kappa_\Box$ too, so that ISL strongly normalises w.r.t. the extended system.

Conjecture

Every formula B occurring in a normal – w.r.t. the standard system extended by $\rho_\Box + \pi_\Box + o^1_\Box + o^2_\Box + \kappa_\Box$ – ISL-deduction f of A from assumptions Γ is a subformula of A or of some formula in Γ.

Interpretability logics

An interpretation of a theory T into a theory T′ is just a structure preserving translation t such that if $T \vdash A$ then $T' \vdash t(A)$.
Interpretations are ubiquitous in (meta-)mathematics:

- Faithful embeddings;
- Gödel numbering;
- Relative consistency proofs;

$\vdots$

Modal logics for interpretability are an extension of the language of provability logic by means of a binary modal operator $\rhd$ capturing the relation of (relative) interpretability between two arithmetical theories:

$$A \rhd B \stackrel{*}{\simeq} Int_{\mathrm{T}}(\ulcorner A^* \urcorner, \ulcorner B^* \urcorner)$$

where $Int_{\mathrm{T}}(x, y)$ is the formal predicate for relative interpretability over T – expressing the fact that the arithmetical theory T extended by A^* interprets the arithmetical theory T extended by B^*.

Interpretability logics

An interpretation of a theory T into a theory T′ is just a structure preserving translation t such that if $T \vdash A$ then $T' \vdash t(A)$.
Interpretations are ubiquitous in (meta-)mathematics:

- Faithful embeddings;
- Gödel numbering;
- Relative consistency proofs;

$\vdots$

Modal logics for interpretability are an extension of the language of provability logic by means of a binary modal operator $\rhd$ capturing the relation of (relative) interpretability between two arithmetical theories:

$$A \rhd B \overset{*}{\simeq} Int_{\mathrm{T}}(\ulcorner A^* \urcorner, \ulcorner B^* \urcorner)$$

where $Int_{\mathrm{T}}(x, y)$ is the formal predicate for relative interpretability over T – expressing the fact that the arithmetical theory T extended by A^* interprets the arithmetical theory T extended by B^*.

- Axiom schemas of $\mathbb{CPC}$;
- schema IL2 : $A \rhd B \to B \rhd C \to A \rhd C$;
- schema IL3 : $A \rhd C \to B \rhd C \to A \vee B \rhd C$;
- schema IL-Löb: $A \rhd (A \wedge (A \rhd \bot))$;
- MP Rule $\dfrac{A \to B \qquad A}{B}$;
- $\rhd$Rule $\dfrac{A \to B}{A \rhd B}$.

We define

$$\Box A := \neg A \rhd \bot, \text{ and } \Diamond A := \neg \Box \neg A.$$

Let us define as proper extensions of $\mathbb{IL}$

- $\mathbb{ILM} := \mathbb{IL} + \mathsf{M}$, where
$$\mathsf{M} := A \rhd B \to A \wedge \Box C \rhd B \wedge \Box C$$
is called the Montagna schema;
- $\mathbb{ILP} := \mathbb{IL} + \mathsf{P}$, where
$$\mathsf{P} := A \rhd B \to \Box(A \rhd B)$$
is called the persistence schema;
- $\mathbb{ILW} := \mathbb{IL} + \mathsf{W}$, where
$$\mathsf{W} := A \rhd B \to A \rhd B \wedge \Box\neg A$$
is called the de Jongh-Visser schema;
- $\mathbb{ILKM}1 := \mathbb{IL} + \mathsf{KM1}$, where
$$\mathsf{KM1} := A \rhd \Diamond\top \to \top \rhd \neg A;$$
- $\mathbb{ILM}_0 := \mathbb{IL} + \mathsf{M}_0$, where
$$\mathsf{M}_0 := A \rhd B \to \Diamond A \wedge \Box C \rhd B \wedge \Box C;$$

Each of these extensions can be characterised in terms of GVS semantics by imposing specific conditions to frames.

A generalised Veltman frame $\mathcal{F}$ consists of

- a finite set $W \neq \varnothing$;
- a binary relation $R \subseteq W \times W$ which is irreflexive and transitive;
- a W-indexed set of relations $S_x \subseteq R[x] \times (\wp(R[x]) \setminus \{\varnothing\})$;

satisfying the following conditions:

- Quasi-reflexivity: if xRy then $yS_x\{y\}$;
- Definiteness: if $xRyRz$ then $yS_x\{z\}$;
- Monotonicity: if yS_xa and $a \subseteq b \subseteq R[x]$ then yS_xb;
- Quasi-transitivity: if yS_xa and vS_xb_v for all $v \in a$, then $yS_x(\bigcup_{v\in a} b_v)$.

The forcing relation is defined as for standard relational semantics, with only one difference:

$$x \Vdash A \rhd B \text{ iff for all } y \text{ if } xRy \text{ and } y \Vdash A, \text{ then there exists an } a \text{ such that } yS_xa \text{ and } a \Vdash^{\forall} B,$$

where $a \Vdash^{\forall} B$ abbreviates the expression "for any $z \in a, z \Vdash B$".

A generalised Veltman frame $\mathcal{F}$ consists of

- a finite set $W \neq \varnothing$;
- a binary relation $R \subseteq W \times W$ which is irreflexive and transitive;
- a W-indexed set of relations $S_x \subseteq R[x] \times (\wp(R[x]) \setminus \{\varnothing\})$;

satisfying the following conditions:

- Quasi-reflexivity: if xRy then $yS_x\{y\}$;
- Definiteness: if $xRyRz$ then $yS_x\{z\}$;
- Monotonicity: if yS_xa and $a \subseteq b \subseteq R[x]$ then yS_xb;
- Quasi-transitivity: if yS_xa and vS_xb_v for all $v \in a$, then $yS_x(\bigcup_{v \in a} b_v)$.

The forcing relation is defined as for standard relational semantics, with only one difference:

$x \Vdash A \rhd B$ iff for all y if xRy and $y \Vdash A$, then there exists an a such that yS_xa and $a \Vdash^{\forall} B$,

where $a \Vdash^{\forall} B$ abbreviates the expression "for any $z \in a, z \Vdash B$".

In recent years, internalisation techniques of semantic notions in sequent calculi marked an event in proof theory for non-classical logics.

The starting point of that perspective is still the basic G3-paradigm, but the formalism of sequent systems is extended either by

- enriching the language of the calculi themselves (**explicit internalisation**);or by
- enriching the structure of sequents (**implicit internalisation**).

For interpretability logics, we adopted an explicit internalisation methodology, and developed labelled sequent calculi for IL and its extensions.

In recent years, internalisation techniques of semantic notions in sequent calculi marked an event in proof theory for non-classical logics.

The starting point of that perspective is still the basic G3-paradigm, but the formalism of sequent systems is extended either by

- enriching the language of the calculi themselves (**explicit internalisation**); or by
- enriching the structure of sequents (**implicit internalisation**).

For interpretability logics, we adopted an explicit internalisation methodology, and developed labelled sequent calculi for IL and its extensions.

In recent years, internalisation techniques of semantic notions in sequent calculi marked an event in proof theory for non-classical logics.

The starting point of that perspective is still the basic G3-paradigm, but the formalism of sequent systems is extended either by

- enriching the language of the calculi themselves (**explicit internalisation**);or by
- enriching the structure of sequents (**implicit internalisation**).

For interpretability logics, we adopted an explicit internalisation methodology, and developed labelled sequent calculi for IL and its extensions.

In recent years, internalisation techniques of semantic notions in sequent calculi marked an event in proof theory for non-classical logics.

The starting point of that perspective is still the basic G3-paradigm, but the formalism of sequent systems is extended either by

- enriching the language of the calculi themselves (**explicit internalisation**);or by
- enriching the structure of sequents (**implicit internalisation**).

For interpretability logics, we adopted an explicit internalisation methodology, and developed labelled sequent calculi for IL and its extensions.

(♯) $x \Vdash A \rhd B$ iff for all y, if xRy and $y \Vdash A$,
then there exists an a such that $yS_x a$ and $a \Vdash^\forall B$,

(♯♭) $x \Vdash A \rhd B$ iff for all y, if xRy and $y \Vdash A$,
then $y \Vdash \langle]_x B$.

Therefore

$$x \Vdash A \rhd B \quad \text{iff} \quad x \Vdash \Box(A \to \langle]_x B).$$

Moreover, in any irreflexive transitive *finite* frame

$$x \Vdash \Box A \quad \text{iff} \quad \text{for any } y,\ \text{if } xRy \text{ and } y \Vdash \Box A,\ \text{then } y \Vdash A.$$

Henceforth

(♮) $x \Vdash A \rhd_i B$ iff for all y, if xRy and $y \Vdash A \rhd_i B$,
then, if $y \Vdash A$, $y \Vdash \langle]_i B$.

Labelled sequent calculi for interpretability

Starting point

$(\sharp\flat)\quad x \Vdash A \rhd B \quad$ iff $\quad$ for all y, if xRy and $y \Vdash A$,
then $y \Vdash \langle]_x B$.

Therefore

$$x \Vdash A \rhd B \qquad \text{iff} \qquad x \Vdash \Box(A \to \langle]_x B).$$

Moreover, in any irreflexive transitive *finite* frame

$$x \Vdash \Box A \qquad \text{iff} \qquad \text{for any } y,\ \text{if } xRy \text{ and } y \Vdash \Box A,\ \text{then } y \Vdash A.$$

Henceforth

$(\natural)\quad x \Vdash A \rhd_i B \quad$ iff $\quad$ for all y, if xRy and $y \Vdash A \rhd_i B$,
then, if $y \Vdash A$, $\ y \Vdash \langle]_i B$.

($\sharp\flat$) $\quad x \Vdash A \rhd B \quad$ iff $\quad$ for all y, if xRy and $y \Vdash A$, then $y \Vdash \langle]_x B$.

Therefore

$$x \Vdash A \rhd B \qquad \text{iff} \qquad x \Vdash \Box(A \to \langle]_x B).$$

Moreover, in any irreflexive transitive *finite* frame

$$x \Vdash \Box A \qquad \text{iff} \qquad \text{for any } y,\ \text{if } xRy \text{ and } y \Vdash \Box A,\ \text{then } y \Vdash A.$$

Henceforth

($\natural$) $\quad x \Vdash A \rhd_i B \quad$ iff $\quad$ for all y, if xRy and $y \Vdash A \rhd_i B$, then, if $y \Vdash A$, $y \Vdash \langle]_i B$.

G3IL*
Basic calculus

Initial sequents

$x : p, \Gamma \Rightarrow \Delta, x : p$

$x : A \rhd_i B, \Gamma \Rightarrow \Delta, x : A \rhd_i B$

Classical propositional rules: the usual ones, refer to Figure 1.2

Local forcing rules

$$\frac{x : A, x \in A, a \Vdash^{\forall} A, \Gamma \Rightarrow \Delta}{x \in A, a \Vdash^{\forall} A, \Gamma \Rightarrow \Delta}\ \mathcal{L}\Vdash^{\forall}$$

$$\frac{x \in a, \Gamma \Rightarrow \Delta, x : A}{\Gamma \Rightarrow \Delta, a \Vdash^{\forall} A}\ \mathcal{R}\Vdash^{\forall}_{(x!)}$$

Intermediate modality rules

$$\frac{yS_x a, a \Vdash^{\forall} A, \Gamma \Rightarrow \Delta}{y : \langle]_x A, \Gamma \Rightarrow \Delta}\ \mathcal{L}\langle]_{(a!)}$$

$$\frac{yS_x a, \Gamma \Rightarrow \Delta, y : \langle]_x A, a \Vdash^{\forall} A}{yS_x a, \Gamma \Rightarrow \Delta, y : \langle]_x A}\ \mathcal{R}\langle]$$

Interpretability modality rules

$$\frac{y \in R[x], x : A \rhd_i B, \Gamma \Rightarrow \Delta, y : A \qquad y : \langle]_i B, y \in R[x], x : A \rhd_i B, \Gamma \Rightarrow \Delta \qquad y \in R[x], x : A \rhd_i B, \Gamma \Rightarrow \Delta, y : A \rhd_i B}{y \in R[x], x : A \rhd_i B, \Gamma \Rightarrow \Delta}\ \mathcal{L}\rhd_i$$

$$\frac{y \in R[x], y : A, \Gamma, y : A \rhd_i B \Rightarrow \Delta, y : \langle]_i B}{\Gamma \Rightarrow \Delta, x : A \rhd_i B}\ \mathcal{R}\rhd_{i\,(y!)}$$

Rules for GVS

$$\frac{a \subseteq a, \Gamma \Rightarrow \Delta}{\Gamma \Rightarrow \Delta}\ \mathit{Refl}\subseteq \qquad \frac{a \subseteq c, a \subseteq b, b \subseteq c, \Gamma \Rightarrow \Delta}{a \subseteq b, b \subseteq c, \Gamma \Rightarrow \Delta}\ \mathit{Trans}\subseteq$$

$$\frac{x \in b, x \in a, a \subseteq b, \Gamma \Rightarrow \Delta}{x \in a, a \subseteq b, \Gamma \Rightarrow \Delta}\ \mathcal{L}\subseteq$$

$$\frac{x \in \{x\}, \Gamma \Rightarrow \Delta}{\Gamma \Rightarrow \Delta}\ \mathit{Sing}$$

$$\frac{Atm(y), Atm(x), y \in \{x\}, \Gamma \Rightarrow \Delta}{Atm(x), y \in \{x\}, \Gamma \Rightarrow \Delta}\ \mathit{Repl}_1 \qquad \frac{Atm(x), Atm(y), y \in \{x\}, \Gamma \Rightarrow \Delta}{Atm(y), y \in \{x\}, \Gamma \Rightarrow \Delta}\ \mathit{Repl}_2$$

where $Atm(x)$ has one of the following forms: $x : p, x \in a, x \in \{z\}, x \in R[z], z \in R[x], xS_z a, zS_x a$.

$$\frac{}{x \in R[x], \Gamma \Rightarrow \Delta}\ \mathit{Irrefl} \qquad \frac{z \in R[x], y \in R[x], z \in R[y], \Gamma \Rightarrow \Delta}{y \in R[x], z \in R[y], \Gamma \Rightarrow \Delta}\ \mathit{Trans}$$

$$\frac{z \in a, yS_x a, \Gamma \Rightarrow \Delta}{yS_x a, \Gamma \Rightarrow \Delta}\ \mathit{NE}_{(z!)} \qquad \frac{y \in R[x], a \subseteq R[x], yS_x a, \Gamma \Rightarrow \Delta}{yS_x a, \Gamma \Rightarrow \Delta}\ \mathit{DefS1}$$

$$\frac{yS_x\{z\}, y \in R[x], z \in R[y], \Gamma \Rightarrow \Delta}{y \in R[x], z \in R[y], \Gamma \Rightarrow \Delta}\ \mathit{DefS2} \qquad \frac{yS_x b, yS_x a, a \subseteq b, b \subseteq R[x], \Gamma \Rightarrow \Delta}{yS_x a, a \subseteq b, b \subseteq R[x], \Gamma \Rightarrow \Delta}\ \mathit{Mono}$$

$$\frac{yS_x\{y\}, y \in R[x], \Gamma \Rightarrow \Delta}{y \in R[x], \Gamma \Rightarrow \Delta}\ \mathit{Qrefl} \qquad \frac{yS_x b, yS_x a, z \in a, zS_x b, \Gamma \Rightarrow \Delta}{yS_x a, z \in a, zS_x b, \Gamma \Rightarrow \Delta}\ \mathit{Qtrans6}$$

Additional rules for GVS

$$\frac{x \in a, y \in R[x], y \in R[a], \Gamma \Rightarrow \Delta}{y \in R[a], \Gamma \Rightarrow \Delta}\ \mathit{Rset1}_{(x!)} \qquad \frac{y \in R[a], x \in a, y \in R[x], \Gamma \Rightarrow \Delta}{x \in a, y \in R[x], \Gamma \Rightarrow \Delta}\ \mathit{Rset2}$$

$$\frac{yS_x a, y \in S_x^{-1}a, \Gamma \Rightarrow \Delta}{y \in S_x^{-1}a, \Gamma \Rightarrow \Delta}\ \mathit{Sset1} \qquad \frac{y \in S_x^{-1}a, yS_x a, \Gamma \Rightarrow \Delta}{yS_x a, \Gamma \Rightarrow \Delta}\ \mathit{Sset2}$$

$$\frac{c \subseteq a, c \subseteq b, c \subseteq a \cap b, \Gamma \Rightarrow \Delta}{c \subseteq a \cap b, \Gamma \Rightarrow \Delta}\ \cap_1 \qquad \frac{c \subseteq a \cap b, c \subseteq a, c \subseteq b, \Gamma \Rightarrow \Delta}{c \subseteq a, c \subseteq b, \Gamma \Rightarrow \Delta}\ \cap_2$$

$$\frac{}{x \in \varnothing, \Gamma \Rightarrow \Delta}\ \mathcal{L}\varnothing$$

Rules for interpretability principles

$$\frac{b \subseteq a, yS_x b, R[b] \subseteq R[y], yS_x a, \Gamma \Rightarrow \Delta}{yS_x a, \Gamma \Rightarrow \Delta}\ M_{(b!)} \qquad \frac{z \in a, R_z \subseteq R[y], yS_x a, \Gamma \Rightarrow \Delta}{yS_x a, \Gamma \Rightarrow \Delta}\ KM1_{(z!)}$$

$$\frac{b \subseteq a, zS_y b, y \in R[x], z \in R[y], zS_x a, \Gamma \Rightarrow \Delta}{y \in R[x], z \in R[y], zS_x a, \Gamma \Rightarrow \Delta}\ P_{(b!)}$$

$$\frac{b \subseteq a, yS_x b, R[b] \cap S_x^{-1}a \subseteq \varnothing, yS_x a, \Gamma \Rightarrow \Delta}{yS_x a, \Gamma \Rightarrow \Delta}\ W_{(b!)}$$

$$\frac{b \subseteq a, yS_x b, R[b] \subseteq R[y], y \in R[x], z \in R[y], zS_x a, \Gamma \Rightarrow \Delta}{y \in R[x], z \in R[y], zS_x a, \Gamma \Rightarrow \Delta}\ M_{0(b!)}$$

Theorem (PB 2022)

Any calculus in the family $\mathsf{G3IL}^{\star}$ *satisfies the following properties:*

- *Generalised initial sequents are derivable;*
- *Substitution rules for worlds and neighbourhoods are hp-admissible;*
- *Weakening rules are hp-admissible;*
- *All the rules are invertible;*
- *Contraction rules are admissible;*
- *Cut-elimination holds.*

Some care is needed for proving cut elimination:
We had to generalise the strategy by (Negri 2005), and proceed by ternary transfinite induction – main induction on the size of the cut formula, secondary induction on the range of the cut label and tertiary induction on the cut height.

Theorem (PB 2022)

Any calculus in the family $\mathsf{G3IL}^\star$ *satisfies the following properties:*

- *Generalised initial sequents are derivable;*
- *Substitution rules for worlds and neighbourhoods are hp-admissible;*
- *Weakening rules are hp-admissible;*
- *All the rules are invertible;*
- *Contraction rules are admissible;*
- *Cut-elimination holds.*

Some care is needed for proving cut elimination:
We had to generalise the strategy by (Negri 2005), and proceed by ternary transfinite induction – main induction on the size of the cut formula, secondary induction on the range of the cut label and tertiary induction on the cut height.

Each calculus in the family of $G3IL^\star$ is sound and complete w.r.t. the appropriate class of Verbrugge frames: This is shown by interpreting derivations in frames – soundness – and, indirectly, by proving the interpretability principles of each axiomatic calculus – completeness.

After settling

Conjecture

There exists a strategy making proof search in $G3KIL^\star$ for a sequent of the form $\Rightarrow x : A$ always terminate in a finite number of steps. Moreover, from a failed proof search, it is possible to extract a countermodel to A belonging to appropriate class of generalised Veltman frames.

a more direct proof of completeness might be given, by the Tait-Schutte-Takeuti methodology.

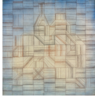
Each calculus in the family of G3IL* is sound and complete w.r.t. the appropriate class of Verbrugge frames: This is shown by interpreting derivations in frames – soundness – and, indirectly, by proving the interpretability principles of each axiomatic calculus – completeness.

After settling

Conjecture

There exists a strategy making proof search in G3KIL* for a sequent of the form $\Rightarrow x : A$ always terminate in a finite number of steps. Moreover, from a failed proof search, it is possible to extract a countermodel to A belonging to appropriate class of generalised Veltman frames.

a more direct proof of completeness might be given, by the Tait-Schutte-Takeuti methodology.



II.
Automated Reasoning

Formalisation and theorem provers

> *My intellect never quite recovered from the strain of writing (Principia Mathematica). I have been ever since definitely less capable of dealing with difficult abstractions than I was before.*

(Russell 1971)

Nowadays, contemporary proof assistants are capable to help the mathematician in formalising substantial bodies of advanced mathematics, and symbolism and formal reasoning do not drive anyone mad – in principle.

In this second part of the talk, I will propose two experiments in automated reasoning, namely

- an implementation in HOL Light of a theorem prover and countermodel constructor for provability logic;
- a formalisation in UniMath of the basics of universal algebra, with an eye at automated computations

and show how the current versions of that code work.

Formalisation and theorem provers

> *My intellect never quite recovered from the strain of writing (Principia Mathematica). I have been ever since definitely less capable of dealing with difficult abstractions than I was before.*

(Russell 1971)

Nowadays, contemporary proof assistants are capable to help the mathematician in formalising substantial bodies of advanced mathematics, and symbolism and formal reasoning do not drive anyone mad – in principle.

In this second part of the talk, I will propose two experiments in automated reasoning, namely

- an implementation in HOL Light of a theorem prover and countermodel constructor for provability logic;
- a formalisation in UniMath of the basics of universal algebra, with an eye at automated computations

and show how the current versions of that code work.

Formalisation and theorem provers

> *My intellect never quite recovered from the strain of writing (Principia Mathematica). I have been ever since definitely less capable of dealing with difficult abstractions than I was before.*

(Russell 1971)

Nowadays, contemporary proof assistants are capable to help the mathematician in formalising substantial bodies of advanced mathematics, and symbolism and formal reasoning do not drive anyone mad – in principle.

In this second part of the talk, I will propose two experiments in automated reasoning, namely

- an implementation in HOL Light of a theorem prover and countermodel constructor for provability logic;
- a formalisation in UniMath of the basics of universal algebra, with an eye at automated computations

and show how the current versions of that code work.

Formalisation and theorem provers

> *My intellect never quite recovered from the strain of writing (Principia Mathematica). I have been ever since definitely less capable of dealing with difficult abstractions than I was before.*

(Russell 1971)

Nowadays, contemporary proof assistants are capable to help the mathematician in formalising substantial bodies of advanced mathematics, and symbolism and formal reasoning do not drive anyone mad – in principle.

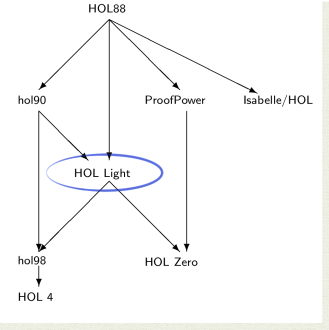
In this second part of the talk, I will propose two experiments in automated reasoning, namely

- an implementation in HOL Light of a theorem prover and countermodel constructor for provability logic;
- a formalisation in UniMath of the basics of universal algebra, with an eye at automated computations

and show how the current versions of that code work.

HOL Light

A brief glance

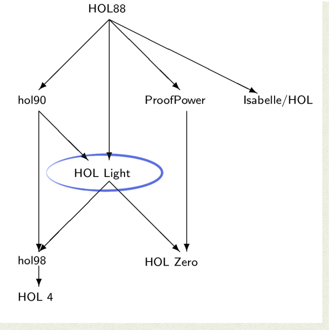


- Clean logical foundations ≈ *Principia Mathematica*
- LCF-style proof checker based on polymorphic simple type theory ≈ small class of *primitive inference rules* for creating theorems + *derived inference rules* to be programmed on top
  - ⇒ 10 primitive rules
  - ⇒ 2 conservative extension principles
  - ⇒ Axioms of choice, extensionality, and infinity
- Written as an OCaml program ≈ three datatypes for the logic: `hol_type`, `term`, and `thm`
- Goal-directed proof development ≈ tactic(al)s + automated methods (in the appropriate domains)

Originally designed and currently maintained by John Harrison, HOL Light is a flexible theorem prover used both in academic and industrial research: formalised incompleteness theorems, topology, floating point algorithms, modules for FlySpeck project

HOL Light

A brief glance

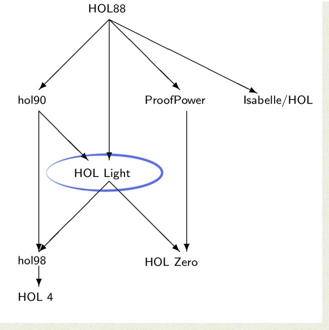


- Clean logical foundations ≈ *Principia Mathematica*
- LCF-style proof checker based on polymorphic simple type theory ≈ small class of *primitive inference rules* for creating theorems + *derived inference rules* to be programmed on top
  - ⇒ 10 primitive rules
  - ⇒ 2 conservative extension principles
  - ⇒ Axioms of choice, extensionality, and infinity
- Written as an OCaml program ≈ three datatypes for the logic: `hol_type`, `term`, and `thm`
- Goal-directed proof development ≈ tactic(al)s + automated methods (in the appropriate domains)

Originally designed and currently maintained by John Harrison, HOL Light is a flexible theorem prover used both in academic and industrial research: formalised incompleteness theorems, topology, floating point algorithms, modules for FlySpeck project

HOL Light

A brief glance

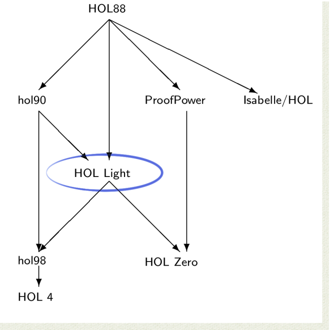


- Clean logical foundations ≈ *Principia Mathematica*
- LCF-style proof checker based on polymorphic simple type theory ≈ small class of *primitive inference rules* for creating theorems + *derived inference rules* to be programmed on top
  - ⇒ 10 primitive rules
  - ⇒ 2 conservative extension principles
  - ⇒ Axioms of choice, extensionality, and infinity
- Written as an OCaml program ≈ three datatypes for the logic: `hol_type`, `term`, and `thm`
- Goal-directed proof development ≈ tactic(al)s + automated methods (in the appropriate domains)

Originally designed and currently maintained by John Harrison, HOL Light is a flexible theorem prover used both in academic and industrial research: formalised incompleteness theorems, topology, floating point algorithms, modules for FlySpeck project

HOL Light

A brief glance

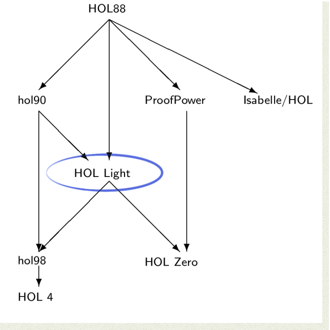


- Clean logical foundations ≈ *Principia Mathematica*
- LCF-style proof checker based on polymorphic simple type theory ≈ small class of *primitive inference rules* for creating theorems + *derived inference rules* to be programmed on top
  - ⇒ 10 primitive rules
  - ⇒ 2 conservative extension principles
  - ⇒ Axioms of choice, extensionality, and infinity
- Written as an OCaml program ≈ three datatypes for the logic: `hol_type`, `term`, and `thm`
- Goal-directed proof development ≈ tactic(al)s + automated methods (in the appropriate domains)

Originally designed and currently maintained by John Harrison, HOL Light is a flexible theorem prover used both in academic and industrial research: formalised incompleteness theorems, topology, floating point algorithms, modules for FlySpeck project

HOL Light

A brief glance



- Clean logical foundations ≈ *Principia Mathematica*
- LCF-style proof checker based on polymorphic simple type theory ≈ small class of *primitive inference rules* for creating theorems + *derived inference rules* to be programmed on top
  - ⇒ 10 primitive rules
  - ⇒ 2 conservative extension principles
  - ⇒ Axioms of choice, extensionality, and infinity
- Written as an OCaml program ≈ three datatypes for the logic: `hol_type`, `term`, and `thm`
- Goal-directed proof development ≈ tactic(al)s + automated methods (in the appropriate domains)

Originally designed and currently maintained by John Harrison, HOL Light is a flexible theorem prover used both in academic and industrial research: formalised incompleteness theorems, topology, floating point algorithms, modules for FlySpeck project

The abstract properties of the provability predicate of any "reasonable" arithmetical theory over a classical base are captured by the system $\mathbb{GL}$, that is made of:

- Axioms of classical propositional logic
- Axiom K : $\Box(A \to B) \to \Box A \to \Box B$
- Axiom GL : $\Box(\Box A \to A) \to \Box A$
- MP Rule $\dfrac{A \to B \qquad A}{B}$
- Nec Rule $\dfrac{A}{\Box A}$

Theorem (Modal adequacy)

$$\mathbb{GL} \vdash A \quad \textit{iff} \quad \mathit{TFT} \vDash A$$

where TFT is the class of relational frames $\mathcal{F} = \langle W, R\rangle$ where W is finite, $R \subseteq W \times W$ is transitive and $\langle W, R\rangle$ defines a tree.

In (Maggesi and PB 2021) we have presented a formalisation in HOL Light of that theorem, and adopted an hybrid proof strategy which considers the difficulties determined by the non-compactness of GL, without incurring in syntactic subtleties sketched in (Boolos 1995).

But we wanted something more. . .

Our theorem prover

G3KGL

Initial sequents:

$x : p, \Gamma \Rightarrow \Delta, x : p$

Propositional rules:

$$\frac{}{x : \bot, \Gamma \Rightarrow \Delta} \; L\bot$$

$$\frac{x : A, x : B, \Gamma \Rightarrow \Delta}{x : A \wedge B, \Gamma \Rightarrow \Delta} \; \mathcal{L}\wedge \qquad \frac{\Gamma \Rightarrow \Delta, x : A \qquad \Gamma \Rightarrow \Delta, x : B}{\Gamma \Rightarrow \Delta, x : A \wedge B} \; \mathcal{R}\wedge$$

$$\frac{x : A, \Gamma \Rightarrow \Delta \qquad x : B, \Gamma \Rightarrow \Delta}{x : A \vee B, \Gamma \Rightarrow \Delta} \; \mathcal{L}\vee \qquad \frac{\Gamma \Rightarrow \Delta, x : A, x : B}{\Gamma \Rightarrow \Delta, x : A \vee B} \; \mathcal{R}\vee$$

$$\frac{\Gamma \Rightarrow \Delta, x : A}{x : \neg A, \Gamma \Rightarrow \Delta} \; \mathcal{L}\neg \qquad \frac{x : A, \Gamma \Rightarrow \Delta}{\Gamma \Rightarrow \Delta, x : \neg A} \; \mathcal{R}\neg$$

$$\frac{\Gamma \Rightarrow \Delta, x : A \qquad x : B, \Gamma \Rightarrow \Delta}{x : A \to B, \Gamma \Rightarrow \Delta} \; \mathcal{L}\to \qquad \frac{x : A, \Gamma \Rightarrow \Delta, x : B}{\Gamma \Rightarrow \Delta, x : A \to B} \; \mathcal{R}\to$$

Modal rules:

$$\frac{y : A, xRy, x : \Box A, \Gamma \Rightarrow \Delta}{xRy, x : \Box A, \Gamma \Rightarrow \Delta} \; \mathcal{L}\Box \qquad \frac{xRy, y : \Box A, \Gamma \Rightarrow \Delta, y : A}{\Gamma \Rightarrow \Delta, x : \Box A} \; \mathcal{R}\Box^{\text{Löb}}_{(y!)}$$

Semantic rules:

$$\frac{}{xRx, \Gamma \Rightarrow \Delta} \; \mathit{Irref} \qquad \frac{xRz, xRy, yRz, \Gamma \Rightarrow \Delta}{xRy, yRz, \Gamma \Rightarrow \Delta} \; \mathit{Trans}$$

It is not hard to see how to use both our formalisation of modal completeness and the already known proof theory for G3KGL to the aim of implementing a decision algorithm in HOL Light for GL: Our predicate `holds (W,R) V A x` corresponds exactly to the labelled formula $x : A$.

Thus we have three different ways of expressing the fact that a world x forces A in a given model $\langle W, R, v\rangle$:

SEMANTIC NOTATION	$x \Vdash A$
LABELLED SEQUENT CALCULUS NOTATION	$x : A$
HOL LIGHT NOTATION	`holds (W,R) V A x`

It is not hard to see how to use both our formalisation of modal completeness and the already known proof theory for G3KGL to the aim of implementing a decision algorithm in HOL Light for GL: Our predicate `holds (W,R) V A x` corresponds exactly to the labelled formula $x : A$.
Thus we have three different ways of expressing the fact that a world x forces A in a given model $\langle W, R, v \rangle$:

SEMANTIC NOTATION	$x \Vdash A$
LABELLED SEQUENT CALCULUS NOTATION	$x : A$
HOL LIGHT NOTATION	`holds (W,R) V A x`

Our tactic `GL_TAC` works as expected:

1. Given a formula A of $\mathcal{L}$, OCaml `let`-terms are rewritten together with definable modal operators, and the goal is set to `|-- A`;
2. A model $\langle W, R, v\rangle$ and a world $w \in W$ – where W sits on the type `num` – are introduced. The main goal is now `holds (W,R) V A w`;
3. Meta-hypotheses `trans boxr1 boxr2 boxl1 boxl2 w` are introduced to be able to handle modal and relational rules;
4. All possible propositional rules are applied after unfolding the predicate `holds`. This assures that, at each step of the proof search, the goal term is a finite conjunction of disjunctions of positive and negative `holds`-propositions. As usual, priority is given to non-branching rules, i.e. to those that do not generate subgoals. Furthermore, the hypothesis list is checked, and `trans` is applied whenever possible; the same holds for $\mathcal{L}\Box$, which is applied to any appropriate hypothesis after the tactic triggering `trans`. Each new goal term is reordered by `SORT_BOX_TAC`, which always precedes the implementation of $\mathcal{R}\Box^{Löb}$.

The procedure is repeated starting from step 2. The tactic ruling it is

```
FIRST o map CHANGED_TAC,
```

which triggers the correct non-failing tactic.
By calling `ASM_REWRITE_TAC`, an additional condition states that the current branch is closed, i.e. an initial sequent has been reached, or the sequent currently analysed has a labelled formula $x : \bot$ in the antecedent.

Moreover, if the proof search fails, a countermodel to the input formula is stored in the HOL Light proof development process, and the computer returns all the necessary information to the user.

Our theorem prover

Design of the proof search

Our tactic `GL_TAC` works as expected:

1. Given a formula A of $\mathcal{L}$, OCaml `let`-terms are rewritten together with definable modal operators, and the goal is set to `|-- A`;
2. A model $\langle W, R, v\rangle$ and a world $w \in W$ – where W sits on the type `num` – are introduced. The main goal is now `holds (W,R) V A w`;
3. Meta-hypotheses `trans boxr1 boxr2 boxl1 boxl2 w` are introduced to be able to handle modal and relational rules;
4. All possible propositional rules are applied after unfolding the predicate `holds`. This assures that, at each step of the proof search, the goal term is a finite conjunction of disjunctions of positive and negative `holds`-propositions. As usual, priority is given to non-branching rules, i.e. to those that do not generate subgoals. Furthermore, the hypothesis list is checked, and `trans` is applied whenever possible; the same holds for $\mathcal{L}\Box$, which is applied to any appropriate hypothesis after the tactic triggering `trans`. Each new goal term is reordered by `SORT_BOX_TAC`, which always precedes the implementation of $\mathcal{R}\Box^{Löb}$.

The procedure is repeated starting from step 2. The tactic ruling it is

```
FIRST o map CHANGED_TAC,
```

which triggers the correct non-failing tactic.
By calling `ASM_REWRITE_TAC`, an additional condition states that the current branch is closed, i.e. an initial sequent has been reached, or the sequent currently analysed has a labelled formula $x : \bot$ in the antecedent.

Moreover, if the proof search fails, a countermodel to the input formula is stored in the HOL Light proof development process, and the computer returns all the necessary information to the user.

Our theorem prover

Design of the proof search

Our tactic `GL_TAC` works as expected:

1. Given a formula A of $\mathcal{L}$, OCaml `let`-terms are rewritten together with definable modal operators, and the goal is set to `|-- A`;
2. A model $\langle W, R, v \rangle$ and a world $w \in W$ – where W sits on the type `num` – are introduced. The main goal is now `holds (W,R) V A w`;
3. Meta-hypotheses `trans boxr1 boxr2 boxl1 boxl2 w` are introduced to be able to handle modal and relational rules;
4. All possible propositional rules are applied after unfolding the predicate `holds`. This assures that, at each step of the proof search, the goal term is a finite conjunction of disjunctions of positive and negative `holds`-propositions. As usual, priority is given to non-branching rules, i.e. to those that do not generate subgoals. Furthermore, the hypothesis list is checked, and `trans` is applied whenever possible; the same holds for $\mathcal{L}\Box$, which is applied to any appropriate hypothesis after the tactic triggering `trans`. Each new goal term is reordered by `SORT_BOX_TAC`, which always precedes the implementation of $\mathcal{R}\Box^{Löb}$.

The procedure is repeated starting from step 2. The tactic ruling it is

```
FIRST o map CHANGED_TAC,
```

which triggers the correct non-failing tactic.
By calling `ASM_REWRITE_TAC`, an additional condition states that the current branch is closed, i.e. an initial sequent has been reached, or the sequent currently analysed has a labelled formula $x : \bot$ in the antecedent.

Moreover, if the proof search fails, a countermodel to the input formula is stored in the HOL Light proof development process, and the computer returns all the necessary information to the user.

Our tactic `GL_TAC` works as expected:

1. Given a formula A of $\mathcal{L}$, OCaml `let`-terms are rewritten together with definable modal operators, and the goal is set to `|-- A`;
2. A model $\langle W, R, v\rangle$ and a world $w \in W$ – where W sits on the type `num` – are introduced. The main goal is now `holds (W,R) V A w`;
3. Meta-hypotheses `trans boxr1 boxr2 boxl1 boxl2 w` are introduced to be able to handle modal and relational rules;
4. All possible propositional rules are applied after unfolding the predicate `holds`. This assures that, at each step of the proof search, the goal term is a finite conjunction of disjunctions of positive and negative `holds`-propositions. As usual, priority is given to non-branching rules, i.e. to those that do not generate subgoals. Furthermore, the hypothesis list is checked, and `trans` is applied whenever possible; the same holds for $\mathcal{L}\Box$, which is applied to any appropriate hypothesis after the tactic triggering `trans`. Each new goal term is reordered by `SORT_BOX_TAC`, which always precedes the implementation of $\mathcal{R}\Box^{Löb}$.

The procedure is repeated starting from step 2. The tactic ruling it is

```
FIRST o map CHANGED_TAC,
```

which triggers the correct non-failing tactic.
By calling `ASM_REWRITE_TAC`, an additional condition states that the current branch is closed, i.e. an initial sequent has been reached, or the sequent currently analysed has a labelled formula $x : \bot$ in the antecedent.

Moreover, if the proof search fails, a countermodel to the input formula is stored in the HOL Light proof development process, and the computer returns all the necessary information to the user.

Our tactic `GL_TAC` works as expected:

1. Given a formula A of $\mathcal{L}$, OCaml `let`-terms are rewritten together with definable modal operators, and the goal is set to `|-- A`;
2. A model $\langle W, R, v\rangle$ and a world $w \in W$ – where W sits on the type `num` – are introduced. The main goal is now `holds (W,R) V A w`;
3. Meta-hypotheses `trans boxr1 boxr2 boxl1 boxl2 w` are introduced to be able to handle modal and relational rules;
4. All possible propositional rules are applied after unfolding the predicate `holds`. This assures that, at each step of the proof search, the goal term is a finite conjunction of disjunctions of positive and negative `holds`-propositions. As usual, priority is given to non-branching rules, i.e. to those that do not generate subgoals. Furthermore, the hypothesis list is checked, and `trans` is applied whenever possible; the same holds for $\mathcal{L}\Box$, which is applied to any appropriate hypothesis after the tactic triggering `trans`. Each new goal term is reordered by `SORT_BOX_TAC`, which always precedes the implementation of $\mathcal{R}\Box^{Löb}$.

The procedure is repeated starting from step 2. The tactic ruling it is

```
FIRST o map CHANGED_TAC,
```

which triggers the correct non-failing tactic.
By calling `ASM_REWRITE_TAC`, an additional condition states that the current branch is closed, i.e. an initial sequent has been reached, or the sequent currently analysed has a labelled formula $x : \bot$ in the antecedent.

Moreover, if the proof search fails, a countermodel to the input formula is stored in the HOL Light proof development process, and the computer returns all the necessary information to the user.

Our tactic `GL_TAC` works as expected:

1. Given a formula A of $\mathcal{L}$, OCaml `let`-terms are rewritten together with definable modal operators, and the goal is set to `|-- A`;
2. A model $\langle W, R, v\rangle$ and a world $w \in W$ – where W sits on the type `num` – are introduced. The main goal is now `holds (W,R) V A w`;
3. Meta-hypotheses `trans boxr1 boxr2 boxl1 boxl2 w` are introduced to be able to handle modal and relational rules;
4. All possible propositional rules are applied after unfolding the predicate `holds`. This assures that, at each step of the proof search, the goal term is a finite conjunction of disjunctions of positive and negative `holds`-propositions. As usual, priority is given to non-branching rules, i.e. to those that do not generate subgoals. Furthermore, the hypothesis list is checked, and `trans` is applied whenever possible; the same holds for $\mathcal{L}\Box$, which is applied to any appropriate hypothesis after the tactic triggering `trans`. Each new goal term is reordered by `SORT_BOX_TAC`, which always precedes the implementation of $\mathcal{R}\Box^{Löb}$.

The procedure is repeated starting from step 2. The tactic ruling it is

```
FIRST o map CHANGED_TAC,
```

which triggers the correct non-failing tactic.
By calling `ASM_REWRITE_TAC`, an additional condition states that the current branch is closed, i.e. an initial sequent has been reached, or the sequent currently analysed has a labelled formula $x : \bot$ in the antecedent.

Moreover, if the proof search fails, a countermodel to the input formula is stored in the HOL Light proof development process, and the computer returns all the necessary information to the user.

Our theorem prover

At work

Our code is integrated in the official HOL Light distribution
`https://github.com/jrh13/hol-light`
and is surveyed in (Maggesi and PB 2022) – under review.

Let's have a run on it now

Our theorem prover

At work

Our code is integrated in the official HOL Light distribution

`https://github.com/jrh13/hol-light`

and is surveyed in (Maggesi and PB 2022) – under review.

Let's have a run on it now

UniMath origin dates back to 2014 when three Coq libraries were combined:

- Foundations (Voevodsky, 2010)
- RezkCompletion (Ahrens, Kapulkin, Shulman, 2013)
- Ktheory (Grayson, 2013)

Martin-Löf Type Theory / subsystem of Coq:

- no `record` types
- no `inductive` types
- no `match` construct

Extended by:

- Univalence (and Function Extensionality) Axiom(s)
- Propositional Resizing

UniMath origin dates back to 2014 when three Coq libraries were combined:

- Foundations (Voevodsky, 2010)
- RezkCompletion (Ahrens, Kapulkin, Shulman, 2013)
- Ktheory (Grayson, 2013)

Martin-Löf Type Theory / subsystem of Coq:

- no `record` types
- no `inductive` types
- no `match` construct

Extended by:

- Univalence (and Function Extensionality) Axiom(s)
- Propositional Resizing

Universal algebra in UniMath

(Amato, Maggesi, PB 2021), (Amato, Maggesi, Parton, PB 2020)

We introduced the basic notions for developing investigations in universal algebra from the univalent perspective:

- multi-sorted signatures
- algebras and their univalent category
- free algebras
- theories and their univalent category
- terms

$\rightsquigarrow$ no inductive types in UniMath

Sketch of our implementation

- $t \in T(sigma, V)$ $\rightsquigarrow$ list of function symbols (and variables)
- Lists are executed by a stack machine (`status` monad on natural numbers)
  - Status `n` $\rightsquigarrow$ remaining elements after execution
  - Status `error` $\rightsquigarrow$ stack underflow
- At the end of execution, a w.f. term always has status `1`
- Induction principle in order to reason on terms

```
Theorem term_ind (P: term sigma → UU)
  (R: term_ind_HP P) (t: term sigma): P t.
```

`R` states that for any symbol `nm` of σ, if `P` holds for any elements of the list corresponding to `nm`, then `P` holds for the whole term.

$\rightsquigarrow$ UniMath is able to use that to perform computations autonomously

$\rightsquigarrow$ both proofs *and* computations concerning terms in UniMath environment $\hookrightarrow$ Poincaré Principle

Universal algebra in UniMath

(Amato, Maggesi, PB 2021), (Amato, Maggesi, Parton, PB 2020)

We introduced the basic notions for developing investigations in universal algebra from the univalent perspective:

- multi-sorted signatures
- algebras and their univalent category
- free algebras
- theories and their univalent category
- terms

$\rightsquigarrow$ no inductive types in UniMath

Sketch of our implementation

- $t \in T(sigma, V)$ $\rightsquigarrow$ list of function symbols (and variables)
- Lists are executed by a stack machine (`status` monad on natural numbers)
  - Status `n` $\rightsquigarrow$ remaining elements after execution
  - Status `error` $\rightsquigarrow$ stack underflow
- At the end of execution, a w.f. term always has status `1`
- Induction principle in order to reason on terms

```
Theorem term_ind (P: term sigma → UU)
  (R: term_ind_HP P) (t: term sigma): P t.
```

`R` states that for any symbol `nm` of σ, if `P` holds for any elements of the list corresponding to `nm`, then `P` holds for the whole term.

$\rightsquigarrow$ UniMath is able to use that to perform computations autonomously

$\rightsquigarrow$ both proofs *and* computations concerning terms in UniMath environment $\hookrightarrow$ Poincaré Principle

Universal algebra in UniMath

(Amato, Maggesi, PB 2021), (Amato, Maggesi, Parton, PB 2020)

We introduced the basic notions for developing investigations in universal algebra from the univalent perspective:

- multi-sorted signatures
- algebras and their univalent category
- free algebras
- theories and their univalent category
- terms

$\rightsquigarrow$ no inductive types in UniMath

Sketch of our implementation

- $t \in T(sigma, V)$ $\rightsquigarrow$ list of function symbols (and variables)
- Lists are executed by a stack machine (`status` monad on natural numbers)
  - Status `n` $\rightsquigarrow$ remaining elements after execution
  - Status `error` $\rightsquigarrow$ stack underflow
- At the end of execution, a w.f. term always has status `1`
- Induction principle in order to reason on terms

```
Theorem term_ind (P: term sigma → UU)
  (R: term_ind_HP P) (t: term sigma): P t.
```

`R` states that for any symbol `nm` of σ, if `P` holds for any elements of the list corresponding to `nm`, then `P` holds for the whole term.

$\rightsquigarrow$ UniMath is able to use that to perform computations autonomously

$\rightsquigarrow$ both proofs *and* computations concerning terms in UniMath environment ↬ Poincaré Principle

Universal algebra in UniMath

(Amato, Maggesi, PB 2021), (Amato, Maggesi, Parton, PB 2020)

We introduced the basic notions for developing investigations in universal algebra from the univalent perspective:

- multi-sorted signatures
- algebras and their univalent category
- free algebras
- theories and their univalent category
- terms

$\rightsquigarrow$ no inductive types in UniMath

Sketch of our implementation

- $t \in T(sigma, V)$ $\rightsquigarrow$ list of function symbols (and variables)
- Lists are executed by a stack machine (`status` monad on natural numbers)
  - Status `n` $\rightsquigarrow$ remaining elements after execution
  - Status `error` $\rightsquigarrow$ stack underflow
- At the end of execution, a w.f. term always has status `1`
- Induction principle in order to reason on terms

```
Theorem term_ind (P: term sigma → UU)
  (R: term_ind_HP P) (t: term sigma): P t.
```

`R` states that for any symbol `nm` of σ, if `P` holds for any elements of the list corresponding to `nm`, then `P` holds for the whole term.

$\rightsquigarrow$ UniMath is able to use that to perform computations autonomously

$\rightsquigarrow$ both proofs *and* computations concerning terms in UniMath environment $\looparrowright$ Poincaré Principle

Universal algebra in UniMath

(Amato, Maggesi, PB 2021), (Amato, Maggesi, Parton, PB 2020)

We introduced the basic notions for developing investigations in universal algebra from the univalent perspective:

- multi-sorted signatures
- algebras and their univalent category
- free algebras
- theories and their univalent category
- terms

$\rightsquigarrow$ no inductive types in UniMath

Sketch of our implementation

- $t \in T(sigma, V)$ $\rightsquigarrow$ list of function symbols (and variables)
- Lists are executed by a stack machine (`status` monad on natural numbers)
  - Status `n` $\rightsquigarrow$ remaining elements after execution
  - Status `error` $\rightsquigarrow$ stack underflow
- At the end of execution, a w.f. term always has status `1`
- Induction principle in order to reason on terms

```
Theorem term_ind (P: term sigma → UU)
  (R: term_ind_HP P) (t: term sigma): P t.
```

`R` states that for any symbol `nm` of σ, if `P` holds for any elements of the list corresponding to `nm`, then `P` holds for the whole term.

$\rightsquigarrow$ UniMath is able to use that to perform computations autonomously

$\rightsquigarrow$ both proofs *and* computations concerning terms in UniMath environment $\looparrowright$ Poincaré Principle

Our code is integrated in the official UniMath repository at
`https://github.com/UniMath/UniMath/tree/master/UniMath/Algebra/Universal`
and it is under further development at
`https://github.com/amato-gianluca/UniMath/tree/wtypes/UniMath`

Let's see how it works by now

Our code is integrated in the official UniMath repository at
https://github.com/UniMath/UniMath/tree/master/UniMath/Algebra/Universal
and it is under further development at
https://github.com/amato-gianluca/UniMath/tree/wtypes/UniMath

Let's see how it works by now

Many thanks
for your attention