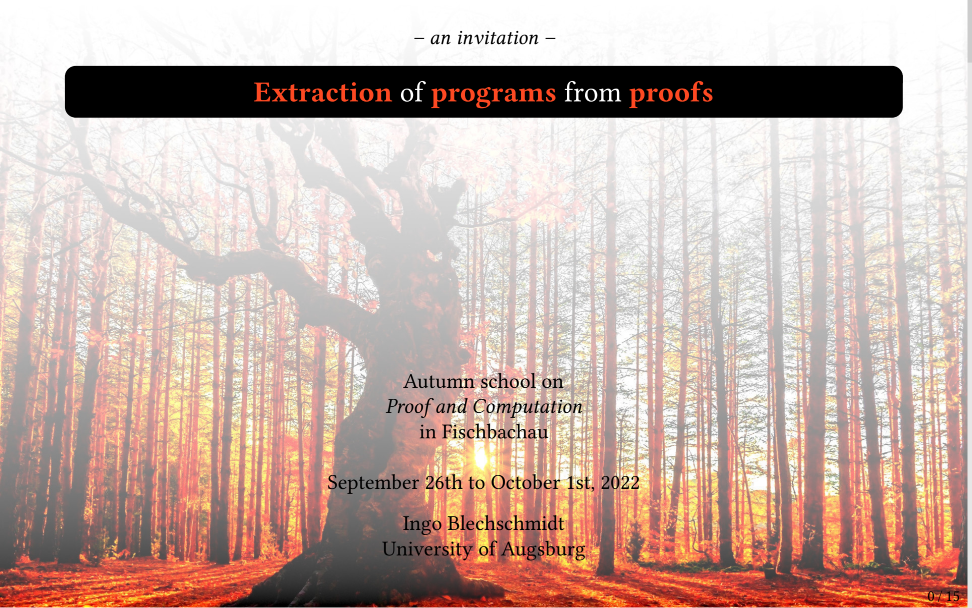


– an invitation –

Extraction of programs from proofs

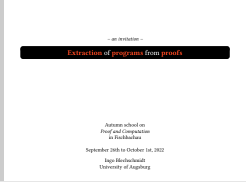
Autumn school on
Proof and Computation
in Fischbachau

September 26th to October 1st, 2022

Ingo Blechschmidt
University of Augsburg

2022-10-04

Extraction of programs from proofs



– an invitation –

Extraction of programs from proofs

Autumn school on
Proof and Computation
in Fischbachau

September 26th to October 1st, 2022

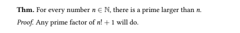
Ingo Blechschmidt
University of Augsburg

Thm. For every number $n \in \mathbb{N}$, there is a prime larger than n.

Proof. Any prime factor of $n! + 1$ will do.

Extraction of programs from proofs

2022-10-04


Thm. For every number $n \in \mathbb{N}$, there is a prime larger than n.

Proof. Any prime factor of $n! + 1$ will do.

From the displayed proof of Euclid's theorem, we can read off an algorithm for computing arbitrarily large primes. There is a deeper reason to that: The proof is *constructive*, and from *every* constructive proof we can extract a corresponding program. One way to formally state and prove this meta-statement is by *realizability theory*, the subject of the first lecture.

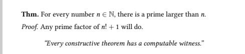
Thm. For every number $n \in \mathbb{N}$, there is a prime larger than n.

Proof. Any prime factor of $n! + 1$ will do.

“Every constructive theorem has a computable witness.”

2022-10-04

Extraction of programs from proofs



Thm. For every number $n \in \mathbb{N}$, there is a prime larger than n.

Proof. Any prime factor of $n! + 1$ will do.

“Every constructive theorem has a computable witness.”

From the displayed proof of Euclid’s theorem, we can read off an algorithm for computing arbitrarily large primes. There is a deeper reason to that: The proof is *constructive*, and from *every* constructive proof we can extract a corresponding program. One way to formally state and prove this meta-statement is by *realizability theory*, the subject of the first lecture.

Thm. For every number $n \in \mathbb{N}$, there is a prime larger than n.

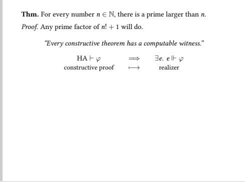
Proof. Any prime factor of $n! + 1$ will do.

"Every constructive theorem has a computable witness."

$$\begin{array}{ccc} \mathrm{HA} \vdash \varphi & \Longrightarrow & \exists e.\ e \Vdash \varphi \\ \text{constructive proof} & \longmapsto & \text{realizer} \end{array}$$

Extraction of programs from proofs

2022-10-04



Thm. For every number $n \in \mathbb{N}$, there is a prime larger than n.

Proof. Any prime factor of $n! + 1$ will do.

"Every constructive theorem has a computable witness."

$$\begin{array}{ccc} \mathrm{HA} \vdash \varphi & \Longrightarrow & \exists e.\ e \Vdash \varphi \\ \text{constructive proof} & \longmapsto & \text{realizer} \end{array}$$

From the displayed proof of Euclid's theorem, we can read off an algorithm for computing arbitrarily large primes. There is a deeper reason to that: The proof is *constructive*, and from *every* constructive proof we can extract a corresponding program. One way to formally state and prove this meta-statement is by *realizability theory*, the subject of the first lecture.

Thm. For every number $n \in \mathbb{N}$, there is a prime larger than n.

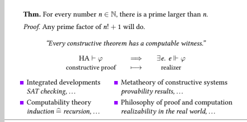
Proof. Any prime factor of $n! + 1$ will do.

"Every constructive theorem has a computable witness."

$$\begin{array}{ccc} \mathrm{HA} \vdash \varphi & \Longrightarrow & \exists e.\ e \Vdash \varphi \\ \text{constructive proof} & \longmapsto & \text{realizer} \end{array}$$

- Integrated developments
  SAT checking, …
- Computability theory
  induction $\widehat{=}$ recursion, …
- Metatheory of constructive systems
  provability results, …
- Philosophy of proof and computation
  realizability in the real world, …

Extraction of programs from proofs


Thm. For every number $n \in \mathbb{N}$, there is a prime larger than n.

Proof. Any prime factor of $n! + 1$ will do.

"Every constructive theorem has a computable witness."

$$\begin{array}{ccc} \mathrm{HA} \vdash \varphi & \Longrightarrow & \exists e.\ e \Vdash \varphi \\ \text{constructive proof} & \longmapsto & \text{realizer} \end{array}$$

- Integrated developments *SAT checking, …*
- Computability theory *induction $\widehat{=}$ recursion, …*
- Metatheory of constructive systems *provability results, …*
- Philosophy of proof and computation *realizability in the real world, …*

From the displayed proof of Euclid's theorem, we can read off an algorithm for computing arbitrarily large primes. There is a deeper reason to that: The proof is *constructive*, and from *every* constructive proof we can extract a corresponding program. One way to formally state and prove this meta-statement is by *realizability theory*, the subject of the first lecture.

In addition to the displayed applications of realizability theory, personally I'm intrigued by it for mostly the following reasons: (1) Realizability elucidates the interplay between constructive and computable mathematics. (2) Realizability is a useful guide for pursuing the question whether two given proofs are "secretly the same": Do they have the same computational content? (3) Realizability theory provides us with a host of tantalizing anti-classical models of constructive mathematics.

Thm. For every number $n \in \mathbb{N}$, there is a prime larger than n.

Proof. Any prime factor of $n! + 1$ will do.

"*Every constructive theorem has a computable witness.*"

$$\begin{array}{ccc} \mathrm{HA} \vdash \varphi & \Longrightarrow & \exists e.\ e \Vdash \varphi \\ \text{constructive proof} & \longmapsto & \text{realizer} \end{array}$$

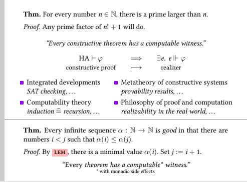
- Integrated developments
  SAT checking, …
- Metatheory of constructive systems
  provability results, …
- Computability theory
  induction $\widehat{=}$ *recursion, …*
- Philosophy of proof and computation
  realizability in the real world, …

Thm. Every infinite sequence $\alpha : \mathbb{N} \to \mathbb{N}$ is *good* in that there are numbers $i < j$ such that $\alpha(i) \leq \alpha(j)$.

Proof. By LEM, there is a minimal value $\alpha(i)$. Set $j := i + 1$.

"Every *theorem has a computable** *witness.*"
* with monadic side effects

Extraction of programs from proofs


Thm. For every number $n \in \mathbb{N}$, there is a prime larger than n.
Proof. Any prime factor of $n! + 1$ will do.
"*Every constructive theorem has a computable witness.*"
$\mathrm{HA} \vdash \varphi \Longrightarrow \exists e.\ e \Vdash \varphi$ constructive proof $\longmapsto$ realizer
Integrated developments *SAT checking, …* Metatheory of constructive systems *provability results, …* Computability theory *induction* $\widehat{=}$ *recursion, …* Philosophy of proof and computation *realizability in the real world, …*
Thm. Every infinite sequence $\alpha : \mathbb{N} \to \mathbb{N}$ is *good* in that there are numbers $i < j$ such that $\alpha(i) \leq \alpha(j)$.
Proof. By LEM, there is a minimal value $\alpha(i)$. Set $j := i + 1$.
"Every *theorem has a computable** *witness.*" * with monadic side effects

From the displayed proof of Euclid's theorem, we can read off an algorithm for computing arbitrarily large primes. There is a deeper reason to that: The proof is *constructive*, and from *every* constructive proof we can extract a corresponding program. One way to formally state and prove this meta-statement is by *realizability theory*, the subject of the first lecture.

In addition to the displayed applications of realizability theory, personally I'm intrigued by it for mostly the following reasons: (1) Realizability elucidates the interplay between constructive and computable mathematics. (2) Realizability is a useful guide for pursuing the question whether two given proofs are "secretly the same": Do they have the same computational content? (3) Realizability theory provides us with a host of tantalizing anti-classical models of constructive mathematics.

In the second lecture, we will turn to extracting programs from *classical* proofs. We will do so by transforming classical proofs into constructive ones and then applying the tools of the first lecture. Amazingly, the displayed classical proof and others like it do have constructive content—even though it is constructively and computably impossible to determine minimal values of infinite sequences.

Thm. For every number $n \in \mathbb{N}$, there is a prime larger than n.

Proof. Any prime factor of $n! + 1$ will do.

“Every constructive theorem has a computable witness.”

$$\begin{array}{ccc} \mathrm{HA} \vdash \varphi & \Longrightarrow & \exists e.\ e \Vdash \varphi \\ \text{constructive proof} & \longmapsto & \text{realizer} \end{array}$$

- Integrated developments
  SAT checking, …
- Computability theory
  induction $\mathrel{\widehat{=}}$ *recursion, …*
- Metatheory of constructive systems
  provability results, …
- Philosophy of proof and computation
  realizability in the real world, …

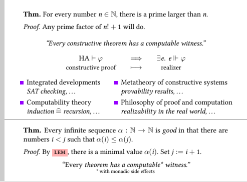
Thm. Every infinite sequence $\alpha : \mathbb{N} \to \mathbb{N}$ is *good* in that there are numbers $i < j$ such that $\alpha(i) \leq \alpha(j)$.

Proof. By LEM, there is a minimal value $\alpha(i)$. Set $j := i + 1$.

“Every *theorem has a computable** *witness.*”
* with monadic side effects

2022-10-04

Extraction of programs from proofs


Thm. For every number $n \in \mathbb{N}$, there is a prime larger than n.
Proof. Any prime factor of $n! + 1$ will do.
“Every constructive theorem has a computable witness.”
HA ⊢ φ ⟹ ∃e. e ⊩ φ
constructive proof ⟼ realizer
Integrated developments *SAT checking, …* Metatheory of constructive systems *provability results, …*
Computability theory *induction ≘ recursion, …* Philosophy of proof and computation *realizability in the real world, …*
Thm. Every infinite sequence $\alpha : \mathbb{N} \to \mathbb{N}$ is *good* in that there are numbers $i < j$ such that $\alpha(i) \leq \alpha(j)$.
Proof. By LEM, there is a minimal value $\alpha(i)$. Set $j := i + 1$.
“Every *theorem has a computable** *witness.*”
* with monadic side effects

Finally, in the third lecture we will learn how to extract constructive content from certain kinds of *invalid* proofs—those which use the preposterous assumption that every set is countable.

The methods presented in the second and in the third lecture are deeply related to the *dynamical approach to algebra* reported on in Stefan Neuwirth’s course. Coherent (and geometric) logic as in Marc Bezem’s course also plays an important role in this toolbox. It is greatly informed by a categorical analysis as provided in Steve Awodey’s course, particularly so for the third lecture. The first and the second lecture overlap with Chuangjie Xu’s course, particularly regarding the double-negation translation.

This course is set in an informal constructive metatheory. Formalization would both be possible in type theory as in Fredrik Nordvall-Forsberg’s course or in constructive set theory as in Hajime Ishihare’s course. All appeals to the transfinite such as by the law of excluded middle or by the axiom of choice will be explicitly pointed out. For primers to constructive mathematics, enjoy Andrej Bauer’s 2013 IAS talk, its written version or these course notes.

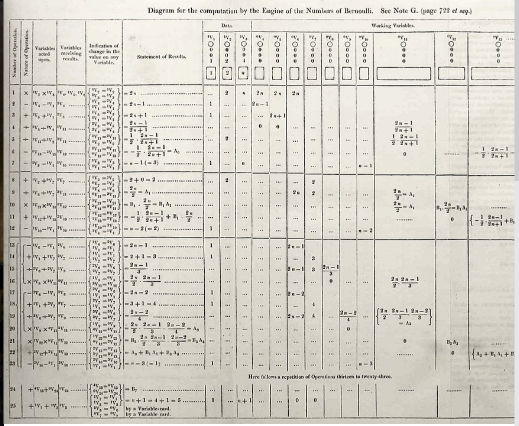


Diagram for the computation by the Engine of the Numbers of Bernoulli. See Note G. (page 722 *et seq.*)

Here follows a repetition of Operations thirteen to twenty-three.

Ada Lovelace,
the world's first
computer programmer

* 1815 † 1852

2022-10-04

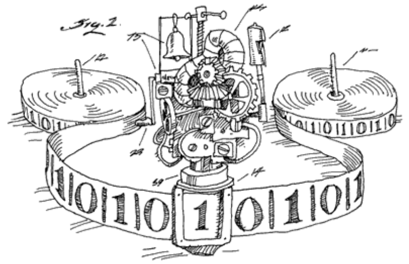
Extraction of programs from proofs

Ada Lovelace,
the world's first
computer programmer
* 1815 † 1852

This year we will celebrate the 207th birthday of Ada Lovelace, pioneer in computing.

It is astonishing what she started and what long way we have come!

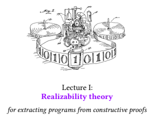
Perhaps you would enjoy the graphic novel *The Thrilling Adventures of Lovelace and Babbage* in her honor.



Lecture I:
Realizability theory

for extracting programs from constructive proofs

Extraction of programs from proofs
└─Realizability theory



Monika Seisenberger has many and more detailed slides on this topic.

For a written primer to realizability theory, see Andrej Bauer's course notes and the notes by Thomas Streicher.

Heyting arithmetic

The **language of arithmetic** has

- as its single sort: N
- as function symbols: 0, S, $+$, $\cdot$
- as its single relation symbol: $=$

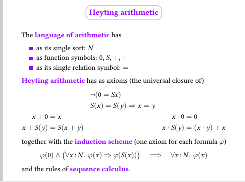
Heyting arithmetic has as axioms (the universal closure of)

$$\neg(0 = Sx)$$
$$S(x) = S(y) \Rightarrow x = y$$
$$x + 0 = x \qquad\qquad x \cdot 0 = 0$$
$$x + S(y) = S(x + y) \qquad\qquad x \cdot S(y) = (x \cdot y) + x$$

together with the **induction scheme** (one axiom for each formula φ)

$$\varphi(0) \wedge (\forall x{:}N.\ \varphi(x) \Rightarrow \varphi(S(x))) \quad\Longrightarrow\quad \forall x{:}N.\ \varphi(x)$$

and the rules of **sequence calculus**.


Heyting arithmetic

The language of arithmetic has

- as its single sort: N
- as function symbols: 0, S, $+$, $\cdot$
- as its single relation symbol: $=$

Heyting arithmetic has as axioms (the universal closure of)

$$\neg(0 = Sx)$$
$$S(x) = S(y) \Rightarrow x = y$$
$$x + 0 = x \qquad x \cdot 0 = 0$$
$$x + S(y) = S(x + y) \qquad x \cdot S(y) = (x \cdot y) + x$$

together with the induction scheme (one axiom for each formula φ)

$$\varphi(0) \wedge (\forall x{:}N.\ \varphi(x) \Rightarrow \varphi(S(x))) \quad\Longrightarrow\quad \forall x{:}N.\ \varphi(x)$$

and the rules of sequence calculus.

Heyting arithmetic is a convenient base theory for a constructive analysis of arithmetic. It has exactly the same axioms as Peano arithmetic, only that HA is set in intuitionistic logic while PA adds the law of excluded middle.

As is common, we define negation "$\neg\varphi$" as a shorthand for the implication "$\varphi \Rightarrow \bot$".

HA is often expanded to HA^ω, *higher-order Heyting arithmetic*, which includes sorts, term constructors and appropriate rules for function types such as N^N and $N^{(N^N)}$.

In its original form, realizability theory is only concernced with extracting computational witnesses from HA-proofs; however it is fruitfully extended to all of HA^ω, and we will also glimpse into this higher-order extension. (NB: HA^ω is conservative over HA, and one way to show this is by realizability.)

Sequence calculus

$$\frac{}{\varphi \vdash_{\vec{x}} \varphi} \qquad \frac{\varphi \vdash_{\vec{x}} \psi}{\varphi[\vec{s}/\vec{x}] \vdash_{\vec{y}} \psi[\vec{s}/\vec{x}]} \qquad \frac{\varphi \vdash_{\vec{x}} \psi \qquad \psi \vdash_{\vec{x}} \chi}{\varphi \vdash_{\vec{x}} \chi}$$

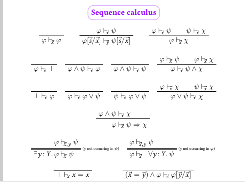
$$\frac{}{\varphi \vdash_{\vec{x}} \top} \qquad \frac{}{\varphi \wedge \psi \vdash_{\vec{x}} \varphi} \qquad \frac{}{\varphi \wedge \psi \vdash_{\vec{x}} \psi} \qquad \frac{\varphi \vdash_{\vec{x}} \psi \qquad \varphi \vdash_{\vec{x}} \chi}{\varphi \vdash_{\vec{x}} \psi \wedge \chi}$$

$$\frac{}{\bot \vdash_{\vec{x}} \varphi} \qquad \frac{}{\varphi \vdash_{\vec{x}} \varphi \vee \psi} \qquad \frac{}{\psi \vdash_{\vec{x}} \varphi \vee \psi} \qquad \frac{\varphi \vdash_{\vec{x}} \chi \qquad \psi \vdash_{\vec{x}} \chi}{\varphi \vee \psi \vdash_{\vec{x}} \chi}$$

$$\begin{array}{c} \varphi \wedge \psi \vdash_{\vec{x}} \chi \\ \hline\hline \varphi \vdash_{\vec{x}} \psi \Rightarrow \chi \end{array}$$

$$\begin{array}{c} \varphi \vdash_{\vec{x},y} \psi \\ \hline\hline \exists y : Y.\, \varphi \vdash_{\vec{x}} \psi \end{array} \ (y \text{ not occurring in } \psi) \qquad \begin{array}{c} \varphi \vdash_{\vec{x},y} \psi \\ \hline\hline \varphi \vdash_{\vec{x}} \ \forall y : Y.\, \psi \end{array} \ (y \text{ not occurring in } \varphi)$$

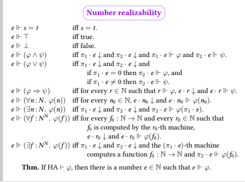
$$\frac{}{\top \vdash_{x} x = x} \qquad \frac{}{(\vec{x} = \vec{y}) \wedge \varphi \vdash_{\vec{z}} \varphi[\vec{y}/\vec{x}]}$$



Number realizability

$e \Vdash s = t$	iff $s = t$.
$e \Vdash \top$	iff true.
$e \Vdash \bot$	iff false.
$e \Vdash (\varphi \wedge \psi)$	iff $\pi_1 \cdot e \downarrow$ and $\pi_2 \cdot e \downarrow$ and $\pi_1 \cdot e \Vdash \varphi$ and $\pi_2 \cdot e \Vdash \psi$.
$e \Vdash (\varphi \vee \psi)$	iff $\pi_1 \cdot e \downarrow$ and $\pi_2 \cdot e \downarrow$ and
	if $\pi_1 \cdot e = 0$ then $\pi_2 \cdot e \Vdash \varphi$, and
	if $\pi_1 \cdot e \neq 0$ then $\pi_2 \cdot e \Vdash \psi$.
$e \Vdash (\varphi \Rightarrow \psi)$	iff for every $r \in \mathbb{N}$ such that $r \Vdash \varphi$, $e \cdot r \downarrow$ and $e \cdot r \Vdash \psi$.
$e \Vdash (\forall n : N.\ \varphi(n))$	iff for every $n_0 \in \mathbb{N}$, $e \cdot n_0 \downarrow$ and $e \cdot n_0 \Vdash \varphi(n_0)$.
$e \Vdash (\exists n : N.\ \varphi(n))$	iff $\pi_1 \cdot e \downarrow$ and $\pi_2 \cdot e \downarrow$ and $\pi_2 \cdot e \Vdash \varphi(\pi_1 \cdot e)$.
$e \Vdash (\forall f : N^N.\ \varphi(f))$	iff for every $f_0 : \mathbb{N} \to \mathbb{N}$ and every $r_0 \in \mathbb{N}$ such that
	f_0 is computed by the r_0-th machine,
	$e \cdot r_0 \downarrow$ and $e \cdot r_0 \Vdash \varphi(f_0)$.
$e \Vdash (\exists f : N^N.\ \varphi(f))$	iff $\pi_1 \cdot e \downarrow$ and $\pi_2 \cdot e \downarrow$ and the $(\pi_1 \cdot e)$-th machine
	computes a function $f_0 : \mathbb{N} \to \mathbb{N}$ and $\pi_2 \cdot e \Vdash \varphi(f_0)$.

Thm. If HA $\vdash \varphi$, then there is a number $e \in \mathbb{N}$ such that $e \Vdash \varphi$.



Number realizability

$e \Vdash s = t$	iff $s = t$.
$e \Vdash \top$	iff true.
$e \Vdash \bot$	iff false.
$e \Vdash (\varphi \wedge \psi)$	iff $\pi_1 \cdot e \downarrow$ and $\pi_2 \cdot e \downarrow$ and $\pi_1 \cdot e \Vdash \varphi$ and $\pi_2 \cdot e \Vdash \psi$.
$e \Vdash (\varphi \vee \psi)$	iff $\pi_1 \cdot e \downarrow$ and $\pi_2 \cdot e \downarrow$ and
	if $\pi_1 \cdot e = 0$ then $\pi_2 \cdot e \Vdash \varphi$, and
	if $\pi_1 \cdot e \neq 0$ then $\pi_2 \cdot e \Vdash \psi$.
$e \Vdash (\varphi \Rightarrow \psi)$	iff for every $r \in \mathbb{N}$ such that $r \Vdash \varphi$, $e \cdot r \downarrow$ and $e \cdot r \Vdash \psi$.
$e \Vdash (\forall n : N.\ \varphi(n))$	iff for every $n_0 \in \mathbb{N}$, $e \cdot n_0 \downarrow$ and $e \cdot n_0 \Vdash \varphi(n_0)$.
$e \Vdash (\exists n : N.\ \varphi(n))$	iff $\pi_1 \cdot e \downarrow$ and $\pi_2 \cdot e \downarrow$ and $\pi_2 \cdot e \Vdash \varphi(\pi_1 \cdot e)$.
$e \Vdash (\forall f : N^N.\ \varphi(f))$	iff for every $f_0 : \mathbb{N} \to \mathbb{N}$ and every $r_0 \in \mathbb{N}$ such that
	f_0 is computed by the r_0-th machine,
	$e \cdot r_0 \downarrow$ and $e \cdot r_0 \Vdash \varphi(f_0)$.
$e \Vdash (\exists f : N^N.\ \varphi(f))$	iff $\pi_1 \cdot e \downarrow$ and $\pi_2 \cdot e \downarrow$ and the $(\pi_1 \cdot e)$-th machine
	computes a function $f_0 : \mathbb{N} \to \mathbb{N}$ and $\pi_2 \cdot e \Vdash \varphi(f_0)$.

Thm. If HA $\vdash \varphi$, then there is a number $e \in \mathbb{N}$ such that $e \Vdash \varphi$.

A statement φ is *realizable*, written "$\Vdash \varphi$", iff it has a *realizer*, a number $e \in \mathbb{N}$ such that $e \Vdash \varphi$. The recursive rules governing which numbers e are deemed to be a realizer of φ make use of *Kleene's original partial combinatory algebra*, the natural numbers equipped with the following partial binary operation $(\cdot)$: $e \cdot n$ is the result of applying the input n to the e-th Turing machine (in some effective enumeration of all Turing machines). We write "$e \cdot n \downarrow$" to signify that this computation terminates.

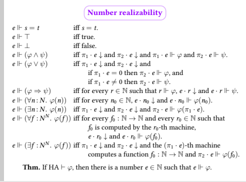
Instead of Turing machines, we can also study other deterministic models of computation by using different partial combinatory algebras, or even nondeterministic and stateful models by a recent tantalizing generalization due to Liron Cohen and her coauthors Sofia Abreu Faro and Ross Tate.

Realizability theory provides one way of formalizing the informal Brouwer–Heyting–Kolmogorov interpretation of constructive mathematics. For instance, that interpretation states that a witness of an implication $\varphi \Rightarrow \psi$ is a "method" for transforming witnesses for φ into witnesses for ψ. Realizability spells out what "method" should mean: Turing machine.

Number realizability

$e \Vdash s = t$	iff $s = t$.
$e \Vdash \top$	iff true.
$e \Vdash \bot$	iff false.
$e \Vdash (\varphi \wedge \psi)$	iff $\pi_1 \cdot e \downarrow$ and $\pi_2 \cdot e \downarrow$ and $\pi_1 \cdot e \Vdash \varphi$ and $\pi_2 \cdot e \Vdash \psi$.
$e \Vdash (\varphi \vee \psi)$	iff $\pi_1 \cdot e \downarrow$ and $\pi_2 \cdot e \downarrow$ and
	if $\pi_1 \cdot e = 0$ then $\pi_2 \cdot e \Vdash \varphi$, and
	if $\pi_1 \cdot e \neq 0$ then $\pi_2 \cdot e \Vdash \psi$.
$e \Vdash (\varphi \Rightarrow \psi)$	iff for every $r \in \mathbb{N}$ such that $r \Vdash \varphi$, $e \cdot r \downarrow$ and $e \cdot r \Vdash \psi$.
$e \Vdash (\forall n : N.\ \varphi(n))$	iff for every $n_0 \in \mathbb{N}$, $e \cdot n_0 \downarrow$ and $e \cdot n_0 \Vdash \varphi(n_0)$.
$e \Vdash (\exists n : N.\ \varphi(n))$	iff $\pi_1 \cdot e \downarrow$ and $\pi_2 \cdot e \downarrow$ and $\pi_2 \cdot e \Vdash \varphi(\pi_1 \cdot e)$.
$e \Vdash (\forall f : N^N.\ \varphi(f))$	iff for every $f_0 : \mathbb{N} \to \mathbb{N}$ and every $r_0 \in \mathbb{N}$ such that
	f_0 is computed by the r_0-th machine,
	$e \cdot r_0 \downarrow$ and $e \cdot r_0 \Vdash \varphi(f_0)$.
$e \Vdash (\exists f : N^N.\ \varphi(f))$	iff $\pi_1 \cdot e \downarrow$ and $\pi_2 \cdot e \downarrow$ and the $(\pi_1 \cdot e)$-th machine
	computes a function $f_0 : \mathbb{N} \to \mathbb{N}$ and $\pi_2 \cdot e \Vdash \varphi(f_0)$.

Thm. If HA $\vdash \varphi$, then there is a number $e \in \mathbb{N}$ such that $e \Vdash \varphi$.



Number realizability

$e \Vdash s = t$ iff $s = t$.
$e \Vdash \top$ iff true.
$e \Vdash \bot$ iff false.
$e \Vdash (\varphi \wedge \psi)$ iff $\pi_1 \cdot e \downarrow$ and $\pi_2 \cdot e \downarrow$ and $\pi_1 \cdot e \Vdash \varphi$ and $\pi_2 \cdot e \Vdash \psi$.
$e \Vdash (\varphi \vee \psi)$ iff $\pi_1 \cdot e \downarrow$ and $\pi_2 \cdot e \downarrow$ and
if $\pi_1 \cdot e = 0$ then $\pi_2 \cdot e \Vdash \varphi$, and
if $\pi_1 \cdot e \neq 0$ then $\pi_2 \cdot e \Vdash \psi$.
$e \Vdash (\varphi \Rightarrow \psi)$ iff for every $r \in \mathbb{N}$ such that $r \Vdash \varphi$, $e \cdot r \downarrow$ and $e \cdot r \Vdash \psi$.
$e \Vdash (\forall n : N.\ \varphi(n))$ iff for every $n_0 \in \mathbb{N}$, $e \cdot n_0 \downarrow$ and $e \cdot n_0 \Vdash \varphi(n_0)$.
$e \Vdash (\exists n : N.\ \varphi(n))$ iff $\pi_1 \cdot e \downarrow$ and $\pi_2 \cdot e \downarrow$ and $\pi_2 \cdot e \Vdash \varphi(\pi_1 \cdot e)$.
$e \Vdash (\forall f : N^N.\ \varphi(f))$ iff for every $f_0 : \mathbb{N} \to \mathbb{N}$ and every $r_0 \in \mathbb{N}$ such that
f_0 is computed by the r_0-th machine,
$e \cdot r_0 \downarrow$ and $e \cdot r_0 \Vdash \varphi(f_0)$.
$e \Vdash (\exists f : N^N.\ \varphi(f))$ iff $\pi_1 \cdot e \downarrow$ and $\pi_2 \cdot e \downarrow$ and the $(\pi_1 \cdot e)$-th machine
computes a function $f_0 : \mathbb{N} \to \mathbb{N}$ and $\pi_2 \cdot e \Vdash \varphi(f_0)$.

Thm. If HA $\vdash \varphi$, then there is a number $e \in \mathbb{N}$ such that $e \Vdash \varphi$.

The clauses for disjunction and existential quantification require pairing and unpairing. Given two numbers a and b, there is a Turing machine $p_{a,b}$ which outputs a or b depending on whether its input is zero or not zero. By π_1 and π_2, we mean (indices of) Turing machines which, when called on input (an index of) $p_{a,b}$, extract a repectively b by simulating its input on the input 0 or 1.

The soundness theorem is proven by an instructive induction on the structure of HA-proofs, verifying that if HA proves a sequent $\varphi \vdash_{\vec{x}} \psi$, then there is a realizer for $\forall x_1. \ldots \forall x_n.\ (\varphi \Rightarrow \psi)$. The core idea of the proof is to verify that every axiom and every rule of HA is realized. In this way, computational content of every axiom and every rule is explicated.

For instance, the statement (with no free variables)

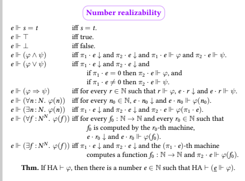
$$\bot \Rightarrow \varphi$$

is realized by any number $e \in \mathbb{N}$ such that for every $r \in \mathbb{N}$ with $r \Vdash \varphi$ (this condition is never satisfied), $e \cdot r \downarrow$ and $e \cdot r \Vdash \varphi$, so by any number whatsoever.

Number realizability

$$
\begin{array}{ll}
e \Vdash s = t & \text{iff } s = t. \\
e \Vdash \top & \text{iff true.} \\
e \Vdash \bot & \text{iff false.} \\
e \Vdash (\varphi \wedge \psi) & \text{iff } \pi_1 \cdot e \downarrow \text{ and } \pi_2 \cdot e \downarrow \text{ and } \pi_1 \cdot e \Vdash \varphi \text{ and } \pi_2 \cdot e \Vdash \psi. \\
e \Vdash (\varphi \vee \psi) & \text{iff } \pi_1 \cdot e \downarrow \text{ and } \pi_2 \cdot e \downarrow \text{ and} \\
 & \qquad \text{if } \pi_1 \cdot e = 0 \text{ then } \pi_2 \cdot e \Vdash \varphi, \text{ and} \\
 & \qquad \text{if } \pi_1 \cdot e \neq 0 \text{ then } \pi_2 \cdot e \Vdash \psi. \\
e \Vdash (\varphi \Rightarrow \psi) & \text{iff for every } r \in \mathbb{N} \text{ such that } r \Vdash \varphi,\ e \cdot r \downarrow \text{ and } e \cdot r \Vdash \psi. \\
e \Vdash (\forall n{:}N.\ \varphi(n)) & \text{iff for every } n_0 \in \mathbb{N},\ e \cdot n_0 \downarrow \text{ and } e \cdot n_0 \Vdash \varphi(n_0). \\
e \Vdash (\exists n{:}N.\ \varphi(n)) & \text{iff } \pi_1 \cdot e \downarrow \text{ and } \pi_2 \cdot e \downarrow \text{ and } \pi_2 \cdot e \Vdash \varphi(\pi_1 \cdot e). \\
e \Vdash (\forall f : N^N.\ \varphi(f)) & \text{iff for every } f_0 : \mathbb{N} \to \mathbb{N} \text{ and every } r_0 \in \mathbb{N} \text{ such that} \\
 & \qquad f_0 \text{ is computed by the } r_0\text{-th machine,} \\
 & \qquad e \cdot r_0 \downarrow \text{ and } e \cdot r_0 \Vdash \varphi(f_0). \\
e \Vdash (\exists f : N^N.\ \varphi(f)) & \text{iff } \pi_1 \cdot e \downarrow \text{ and } \pi_2 \cdot e \downarrow \text{ and the } (\pi_1 \cdot e)\text{-th machine} \\
 & \qquad \text{computes a function } f_0 : \mathbb{N} \to \mathbb{N} \text{ and } \pi_2 \cdot e \Vdash \varphi(f_0).
\end{array}
$$

Thm. If HA $\vdash \varphi$, then there is a number $e \in \mathbb{N}$ such that HA $\vdash (\underline{e} \Vdash \varphi)$.

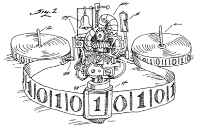


In the form "(HA $\vdash \varphi$) $\Rightarrow$ ($\Vdash \varphi$)", the soundness theorem can be stated and proved in most contexts in which the natural numbers exist as a complete entity, such as constructive or classical set or type theories.

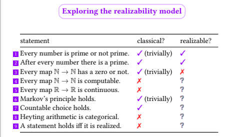
But in a sense, this is misleading: The mapping from proofs to realizers is a computationally simple syntactical transformation. As such, already PRA can prove the soundness theorem if we formulate it as "(HA $\vdash \varphi$) $\Rightarrow$ ($\exists e.$ HA $\vdash (e \Vdash \varphi)$)".

NB: Some formulations of realizability state the clauses for disjunction and existential quantification in a slighter simpler way, directly using pairing and unpairing functions on the naturals. The price for this simplification is that then the soundness theorem has to be formulated as"(HA $\vdash \varphi$) $\Rightarrow$ ($\exists e.$ HA $\vdash (e \Vdash (\top \Rightarrow \varphi))$)".

Exploring the realizability model

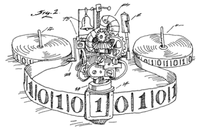


	statement	classical?	realizable?
1	Every number is prime or not prime.	✓ (trivially)	✓
2	After every number there is a prime.	✓	✓
3	Every map $\mathbb{N} \to \mathbb{N}$ has a zero or not.	✓ (trivially)	✗
4	Every map $\mathbb{N} \to \mathbb{N}$ is computable.	✗	?
5	Every map $\mathbb{R} \to \mathbb{R}$ is continuous.	✗	?
6	Markov's principle holds.	✓ (trivially)	?
7	Countable choice holds.	✓	?
8	Heyting arithmetic is categorical.	✗	?
9	A statement holds iff it is realized.	✗	?


Exploring the realizability model

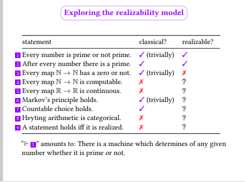
	statement	classical?	realizable?
1	Every number is prime or not prime.	✓ (trivially)	✓
2	After every number there is a prime.	✓	✓
3	Every map $\mathbb{N} \to \mathbb{N}$ has a zero or not.	✓ (trivially)	✗
4	Every map $\mathbb{N} \to \mathbb{N}$ is computable.	✗	?
5	Every map $\mathbb{R} \to \mathbb{R}$ is continuous.	✗	?
6	Markov's principle holds.	✓ (trivially)	?
7	Countable choice holds.	✓	?
8	Heyting arithmetic is categorical.	✗	?
9	A statement holds iff it is realized.	✗	?

Exploring the realizability model



	statement	classical?	realizable?
1	Every number is prime or not prime.	✓ (trivially)	✓
2	After every number there is a prime.	✓	✓
3	Every map $\mathbb{N} \to \mathbb{N}$ has a zero or not.	✓ (trivially)	✗
4	Every map $\mathbb{N} \to \mathbb{N}$ is computable.	✗	?
5	Every map $\mathbb{R} \to \mathbb{R}$ is continuous.	✗	?
6	Markov's principle holds.	✓ (trivially)	?
7	Countable choice holds.	✓	?
8	Heyting arithmetic is categorical.	✗	?
9	A statement holds iff it is realized.	✗	?

"$\Vdash$ 1" amounts to: There is a machine which determines of any given number whether it is prime or not.

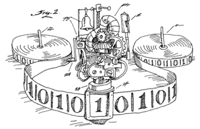


Exploring the realizability model

	statement	classical?	realizable?
1	Every number is prime or not prime.	✓ (trivially)	✓
2	After every number there is a prime.	✓	✓
3	Every map $\mathbb{N} \to \mathbb{N}$ has a zero or not.	✓ (trivially)	✗
4	Every map $\mathbb{N} \to \mathbb{N}$ is computable.	✗	?
5	Every map $\mathbb{R} \to \mathbb{R}$ is continuous.	✗	?
6	Markov's principle holds.	✓ (trivially)	?
7	Countable choice holds.	✓	?
8	Heyting arithmetic is categorical.	✗	?
9	A statement holds iff it is realized.	✗	?

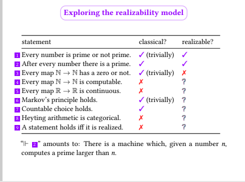
"$\Vdash$ 1" amounts to: There is a machine which determines of any given number whether it is prime or not.

Exploring the realizability model



statement	classical?	realizable?
1 Every number is prime or not prime.	✓ (trivially)	✓
2 After every number there is a prime.	✓	✓
3 Every map $\mathbb{N} \to \mathbb{N}$ has a zero or not.	✓ (trivially)	✗
4 Every map $\mathbb{N} \to \mathbb{N}$ is computable.	✗	?
5 Every map $\mathbb{R} \to \mathbb{R}$ is continuous.	✗	?
6 Markov's principle holds.	✓ (trivially)	?
7 Countable choice holds.	✓	?
8 Heyting arithmetic is categorical.	✗	?
9 A statement holds iff it is realized.	✗	?

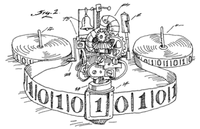
"$\Vdash$ 2" amounts to: There is a machine which, given a number n, computes a prime larger than n.


Exploring the realizability model

statement classical? realizable?
1 Every number is prime or not prime. ✓ (trivially) ✓
2 After every number there is a prime. ✓ ✓
3 Every map $\mathbb{N} \to \mathbb{N}$ has a zero or not. ✓ (trivially) ✗
4 Every map $\mathbb{N} \to \mathbb{N}$ is computable. ✗ ?
5 Every map $\mathbb{R} \to \mathbb{R}$ is continuous. ✗ ?
6 Markov's principle holds. ✓ (trivially) ?
7 Countable choice holds. ✓ ?
8 Heyting arithmetic is categorical. ✗ ?
9 A statement holds iff it is realized. ✗ ?

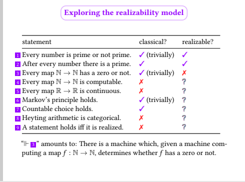
"$\Vdash$ 2" amounts to: There is a machine which, given a number n, computes a prime larger than n.

Exploring the realizability model



statement	classical?	realizable?
1 Every number is prime or not prime.	✓ (trivially)	✓
2 After every number there is a prime.	✓	✓
3 Every map $\mathbb{N} \to \mathbb{N}$ has a zero or not.	✓ (trivially)	✗
4 Every map $\mathbb{N} \to \mathbb{N}$ is computable.	✗	?
5 Every map $\mathbb{R} \to \mathbb{R}$ is continuous.	✗	?
6 Markov's principle holds.	✓ (trivially)	?
7 Countable choice holds.	✓	?
8 Heyting arithmetic is categorical.	✗	?
9 A statement holds iff it is realized.	✗	?

"$\Vdash$ 3" amounts to: There is a machine which, given a machine computing a map $f : \mathbb{N} \to \mathbb{N}$, determines whether f has a zero or not.

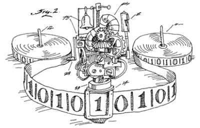


Exploring the realizability model

statement	classical?	realizable?
1 Every number is prime or not prime.	✓ (trivially)	✓
2 After every number there is a prime.	✓	✓
3 Every map $\mathbb{N} \to \mathbb{N}$ has a zero or not.	✓ (trivially)	✗
4 Every map $\mathbb{N} \to \mathbb{N}$ is computable.	✗	?
5 Every map $\mathbb{R} \to \mathbb{R}$ is continuous.	✗	?
6 Markov's principle holds.	✓ (trivially)	?
7 Countable choice holds.	✓	?
8 Heyting arithmetic is categorical.	✗	?
9 A statement holds iff it is realized.	✗	?

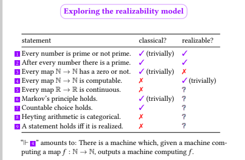
"$\Vdash$ 3" amounts to: There is a machine which, given a machine computing a map $f : \mathbb{N} \to \mathbb{N}$, determines whether f has a zero or not.

Exploring the realizability model



statement	classical?	realizable?
1 Every number is prime or not prime.	✓ (trivially)	✓
2 After every number there is a prime.	✓	✓
3 Every map $\mathbb{N} \to \mathbb{N}$ has a zero or not.	✓ (trivially)	✗
4 Every map $\mathbb{N} \to \mathbb{N}$ is computable.	✗	✓ (trivially)
5 Every map $\mathbb{R} \to \mathbb{R}$ is continuous.	✗	?
6 Markov's principle holds.	✓ (trivially)	?
7 Countable choice holds.	✓	?
8 Heyting arithmetic is categorical.	✗	?
9 A statement holds iff it is realized.	✗	?

“$\Vdash$ 4” amounts to: There is a machine which, given a machine computing a map $f : \mathbb{N} \to \mathbb{N}$, outputs a machine computing f.


Exploring the realizability model

statement	classical?	realizable?
1 Every number is prime or not prime.	✓ (trivially)	✓
2 After every number there is a prime.	✓	✓
3 Every map $\mathbb{N} \to \mathbb{N}$ has a zero or not.	✓ (trivially)	✗
4 Every map $\mathbb{N} \to \mathbb{N}$ is computable.	✗	✓ (trivially)
5 Every map $\mathbb{R} \to \mathbb{R}$ is continuous.	✗	?
6 Markov's principle holds.	✓ (trivially)	?
7 Countable choice holds.	✓	?
8 Heyting arithmetic is categorical.	✗	?
9 A statement holds iff it is realized.	✗	?

“$\Vdash$ 4” amounts to: There is a machine which, given a machine computing a map $f : \mathbb{N} \to \mathbb{N}$, outputs a machine computing f.

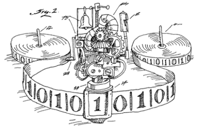
The statement that every function $\mathbb{N} \to \mathbb{N}$ is computable by a Turing machine (or equivalently, by a lambda term) is known as the *formal Church–Turing thesis*. It is an example of a statement which is realizable but not provable in HA^ω.

Many of the curious properties of the realizability model follow from the formal Church–Turing thesis. In fact, a slight generalization called the *extended Church thesis* suffices to completely characterize the (provably) realizable statements:

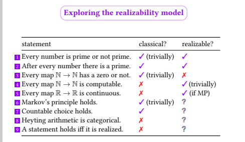
$$\mathrm{HA{+}ECT} \vdash \varphi \quad \text{iff} \quad \mathrm{HA} \vdash (\Vdash \varphi).$$

In the realizability model built using lambda terms instead of Turing machines, the formal Church–Turing thesis fails. This is because of changed calling conventions: In the model built using lambda terms, a realizer for a statement of the form “$\forall f : N^N.\ \varphi(f)$” is a lambda term e such that for every lambda term r computing a function $f : \mathbb{N} \to \mathbb{N}$, the term er is a realizer for $\varphi(f)$. However, the term e cannot inspect the form (“source code”) of its argument.

Exploring the realizability model



	statement	classical?	realizable?
1	Every number is prime or not prime.	✓ (trivially)	✓
2	After every number there is a prime.	✓	✓
3	Every map $\mathbb{N} \to \mathbb{N}$ has a zero or not.	✓ (trivially)	✗
4	Every map $\mathbb{N} \to \mathbb{N}$ is computable.	✗	✓ (trivially)
5	Every map $\mathbb{R} \to \mathbb{R}$ is continuous.	✗	✓ (if MP)
6	Markov's principle holds.	✓ (trivially)	?
7	Countable choice holds.	✓	?
8	Heyting arithmetic is categorical.	✗	?
9	A statement holds iff it is realized.	✗	?

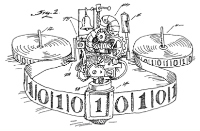

Exploring the realizability model

	statement	classical?	realizable?
1	Every number is prime or not prime.	✓ (trivially)	✓
2	After every number there is a prime.	✓	✓
3	Every map $\mathbb{N} \to \mathbb{N}$ has a zero or not.	✓ (trivially)	✗
4	Every map $\mathbb{N} \to \mathbb{N}$ is computable.	✗	✓ (trivially)
5	Every map $\mathbb{R} \to \mathbb{R}$ is continuous.	✗	✓ (if MP)
6	Markov's principle holds.	✓ (trivially)	?
7	Countable choice holds.	✓	?
8	Heyting arithmetic is categorical.	✗	?
9	A statement holds iff it is realized.	✗	?

Statement 5 is not a statement in the language of HA or of HA$^\omega$, but in an extension in which we can also support quotients (to make sense of the construction of the reals using equivalence classes of Cauchy sequences) or powersets (to support the construction using Dedekind cuts). A proper interpretation is possible in the category of assemblies or in the effective topos.

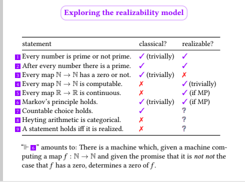
An exposition of this continuity phenomenon is provided in this survey paper (Example 6 there).

Exploring the realizability model



statement	classical?	realizable?
$\boxed{1}$ Every number is prime or not prime.	✓ (trivially)	✓
$\boxed{2}$ After every number there is a prime.	✓	✓
$\boxed{3}$ Every map $\mathbb{N} \to \mathbb{N}$ has a zero or not.	✓ (trivially)	✗
$\boxed{4}$ Every map $\mathbb{N} \to \mathbb{N}$ is computable.	✗	✓ (trivially)
$\boxed{5}$ Every map $\mathbb{R} \to \mathbb{R}$ is continuous.	✗	✓ (if MP)
$\boxed{6}$ Markov's principle holds.	✓ (trivially)	✓ (if MP)
$\boxed{7}$ Countable choice holds.	✓	?
$\boxed{8}$ Heyting arithmetic is categorical.	✗	?
$\boxed{9}$ A statement holds iff it is realized.	✗	?

"$\Vdash \boxed{6}$" amounts to: There is a machine which, given a machine computing a map $f : \mathbb{N} \to \mathbb{N}$ and given the promise that it is *not not* the case that f has a zero, determines a zero of f.



Exploring the realizability model

statement	classical?	realizable?
$\boxed{1}$ Every number is prime or not prime.	✓ (trivially)	✓
$\boxed{2}$ After every number there is a prime.	✓	✓
$\boxed{3}$ Every map $\mathbb{N} \to \mathbb{N}$ has a zero or not.	✓ (trivially)	✗
$\boxed{4}$ Every map $\mathbb{N} \to \mathbb{N}$ is computable.	✗	✓ (trivially)
$\boxed{5}$ Every map $\mathbb{R} \to \mathbb{R}$ is continuous.	✗	✓ (if MP)
$\boxed{6}$ Markov's principle holds.	✓ (trivially)	✓ (if MP)
$\boxed{7}$ Countable choice holds.	✓	?
$\boxed{8}$ Heyting arithmetic is categorical.	✗	?
$\boxed{9}$ A statement holds iff it is realized.	✗	?

"$\Vdash \boxed{6}$" amounts to: There is a machine which, given a machine computing a map $f : \mathbb{N} \to \mathbb{N}$ and given the promise that it is *not not* the case that f has a zero, determines a zero of f.

By *Markov's principle*, we mean the statement

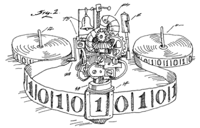
$$\forall f : N^N.\ (\neg\neg(\exists n : N.\ f(n) = 0)) \Rightarrow (\exists n : N.\ f(n) = 0).$$

By the clauses for implication and negation, a number e is a realizer for a negated statement $\neg\psi$ iff there is no realizer for ψ:

$$\begin{aligned} e \Vdash \neg\psi \quad & \text{iff} \quad \text{for every } r \in \mathbb{N} \text{ such that } r \Vdash \psi,\ e \cdot r\downarrow \text{ and } e \cdot r \Vdash \bot \\ & \text{iff} \quad \text{for every } r \in \mathbb{N} \text{ such that } r \Vdash \psi, \text{ falsum holds} \\ & \text{iff} \quad \text{there is no number } r \in \mathbb{N} \text{ such that } r \Vdash \psi \\ & \text{iff} \quad \psi \text{ is not realized} \end{aligned}$$

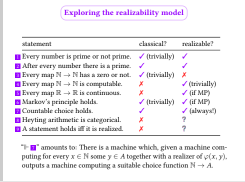
In particular, if there exists a realizer for a negated statement at all, every number whatsoever is a realizer. As a consequence, *realizers for negated statements are never informative;* and a number e is a realizer for $\neg\neg\varphi$ iff φ is *not not* realizable. Hence a realizer for $\neg\neg\varphi$ encodes the mere promise that somewhere, there is a realizer for φ, without giving any indication how to find it.

Exploring the realizability model



	statement	classical?	realizable?
1	Every number is prime or not prime.	✓ (trivially)	✓
2	After every number there is a prime.	✓	✓
3	Every map $\mathbb{N} \to \mathbb{N}$ has a zero or not.	✓ (trivially)	✗
4	Every map $\mathbb{N} \to \mathbb{N}$ is computable.	✗	✓ (trivially)
5	Every map $\mathbb{R} \to \mathbb{R}$ is continuous.	✗	✓ (if MP)
6	Markov's principle holds.	✓ (trivially)	✓ (if MP)
7	Countable choice holds.	✓	✓ (always!)
8	Heyting arithmetic is categorical.	✗	?
9	A statement holds iff it is realized.	✗	?

"$\Vdash$ 7" amounts to: There is a machine which, given a machine computing for every $x \in \mathbb{N}$ some $y \in A$ together with a realizer of $\varphi(x, y)$, outputs a machine computing a suitable choice function $\mathbb{N} \to A$.



Exploring the realizability model

	statement	classical?	realizable?
1	Every number is prime or not prime.	✓ (trivially)	✓
2	After every number there is a prime.	✓	✓
3	Every map $\mathbb{N} \to \mathbb{N}$ has a zero or not.	✓ (trivially)	✗
4	Every map $\mathbb{N} \to \mathbb{N}$ is computable.	✗	✓ (trivially)
5	Every map $\mathbb{R} \to \mathbb{R}$ is continuous.	✗	✓ (if MP)
6	Markov's principle holds.	✓ (trivially)	✓ (if MP)
7	Countable choice holds.	✓	✓ (always!)
8	Heyting arithmetic is categorical.	✗	?
9	A statement holds iff it is realized.	✗	?

"$\Vdash$ 7" amounts to: There is a machine which, given a machine computing for every $x \in \mathbb{N}$ some $y \in A$ together with a realizer of $\varphi(x, y)$, outputs a machine computing a suitable choice function $\mathbb{N} \to A$.

By *countable choice*, we mean the statement

$$(\forall x : N.\ \exists y : A.\ \varphi(x, y)) \Longrightarrow (\exists f : A^N.\ \forall x : N.\ \varphi(x, f(x))).$$

Up to some repackaging, this statement is realized by the identity machine which simply outputs its input unchanged.

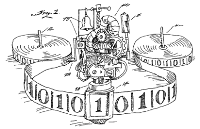
Choice for higher type fails in the realizability model. For instance, the statement

$$(\forall f : N^N.\ \exists y : A.\ \varphi(f, y)) \Longrightarrow (\exists \theta : A^{N^N}.\ \forall f : N^N.\ \varphi(f, \theta(f)))$$

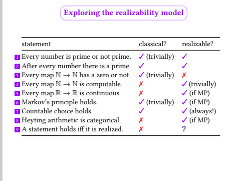
is not realized. A realizer for the antecedent would be a machine which, given an index for a Turing machine computing a total function $f : \mathbb{N} \to \mathbb{N}$, produces a code for a suitable element y. However, this element y might not only depend on the extensional input/output behavior of f, but also on the specific index (source code), hence wouldn't describe an actual function on the set of computable functions $\mathbb{N} \to \mathbb{N}$.

This issue does not arise with countable choice, as natural numbers have unique codes.

Exploring the realizability model



	statement	classical?	realizable?
1	Every number is prime or not prime.	✓ (trivially)	✓
2	After every number there is a prime.	✓	✓
3	Every map $\mathbb{N} \to \mathbb{N}$ has a zero or not.	✓ (trivially)	✗
4	Every map $\mathbb{N} \to \mathbb{N}$ is computable.	✗	✓ (trivially)
5	Every map $\mathbb{R} \to \mathbb{R}$ is continuous.	✗	✓ (if MP)
6	Markov's principle holds.	✓ (trivially)	✓ (if MP)
7	Countable choice holds.	✓	✓ (always!)
8	Heyting arithmetic is categorical.	✗	✓ (if MP)
9	A statement holds iff it is realized.	✗	?



Exploring the realizability model

	statement	classical?	realizable?
1	Every number is prime or not prime.	✓ (trivially)	✓
2	After every number there is a prime.	✓	✓
3	Every map $\mathbb{N} \to \mathbb{N}$ has a zero or not.	✓ (trivially)	✗
4	Every map $\mathbb{N} \to \mathbb{N}$ is computable.	✗	✓ (trivially)
5	Every map $\mathbb{R} \to \mathbb{R}$ is continuous.	✗	✓ (if MP)
6	Markov's principle holds.	✓ (trivially)	✓ (if MP)
7	Countable choice holds.	✓	✓ (always!)
8	Heyting arithmetic is categorical.	✗	✓ (if MP)
9	A statement holds iff it is realized.	✗	?

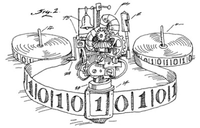
Similar to Statement 5, Statement 8 can only be formulated in an extension of the formal language used here.

It expresses that, up to unique isomorphism, there is just one model of Heyting arithmetic, namely the standard model. This is in stark contrast with the situation in classical mathematics, where Gödel's completeness theorem/Henkin term models can be used to concoct a host of nonstandard models.

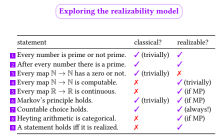
As a consequence, Peano arithmetic is "quasi-inconsistent" from the point of view of the realizability model, as it is consistent (being equiconsistent with HA) but does not admit a model (every model of PA is also a model of HA, but HA only has one model, and this does not validate the PA-theorem "every Turing machine terminates or does not terminate").

Pointers to relevant literature are in this survey paper (Example 8 there). Also see the 2022 paper by Marc Hermes and Dominik Kirst, particularly also the final paragraph of their Section 8.1 which alludes to a result in a different direction.

Exploring the realizability model



	statement	classical?	realizable?
1	Every number is prime or not prime.	✓ (trivially)	✓
2	After every number there is a prime.	✓	✓
3	Every map $\mathbb{N} \to \mathbb{N}$ has a zero or not.	✓ (trivially)	✗
4	Every map $\mathbb{N} \to \mathbb{N}$ is computable.	✗	✓ (trivially)
5	Every map $\mathbb{R} \to \mathbb{R}$ is continuous.	✗	✓ (if MP)
6	Markov's principle holds.	✓ (trivially)	✓ (if MP)
7	Countable choice holds.	✓	✓ (always!)
8	Heyting arithmetic is categorical.	✗	✓ (if MP)
9	A statement holds iff it is realized.	✗	✓


Exploring the realizability model

	statement	classical?	realizable?
1	Every number is prime or not prime.	✓ (trivially)	✓
2	After every number there is a prime.	✓	✓
3	Every map $\mathbb{N} \to \mathbb{N}$ has a zero or not.	✓ (trivially)	✗
4	Every map $\mathbb{N} \to \mathbb{N}$ is computable.	✗	✓ (trivially)
5	Every map $\mathbb{R} \to \mathbb{R}$ is continuous.	✗	✓ (if MP)
6	Markov's principle holds.	✓ (trivially)	✓ (if MP)
7	Countable choice holds.	✓	✓ (always!)
8	Heyting arithmetic is categorical.	✗	✓ (if MP)
9	A statement holds iff it is realized.	✗	✓

As the examples illustrate, realizability and truth do not at all coincide: There are many statements which are realizable but not true from the point of view of classical mathematics (such as the formal Church–Turing thesis) and vice versa (such as the statement that every function $\mathbb{N} \to \mathbb{N}$ has a zero or not).

Within the realizability model, the situation is radically different. For every statement φ, the statement "$\varphi \Leftrightarrow (\Vdash \varphi)$" is realizable.

The multiverse of models of constructive mathematics can be explored from the point of view of any base model, and from the point of view of the realizability model it looks quite different than from the point of view of classical mathematics.

Metatheory of Heyting arithmetic

1. **Unprovability results:**

   There are instances of LEM which HA does not prove, such as "every Turing machine terminates or does not terminate".

2. **Disjunction property:**

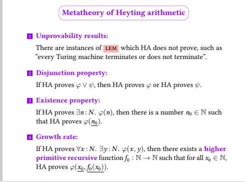
   If HA proves $\varphi \vee \psi$, then HA proves φ or HA proves ψ.

3. **Existence property:**

   If HA proves $\exists n : N.\ \varphi(n)$, then there is a number $n_0 \in \mathbb{N}$ such that HA proves $\varphi(\underline{n_0})$.

4. **Growth rate:**

   If HA proves $\forall x : N.\ \exists y : N.\ \varphi(x, y)$, then there exists a **higher primitive recursive** function $f_0 : \mathbb{N} \to \mathbb{N}$ such that for all $x_0 \in \mathbb{N}$, HA proves $\varphi(\underline{x_0}, \underline{f_0(x_0)})$.



Variants of the realizability model can be used to establish several metatheoretic properties of Heyting arithmetic. For the second and third properties, the keyword is "realizability with proof"; for the fourth, using the variant of realizability built using System T terms instead of Turing machines.

Range of machine models

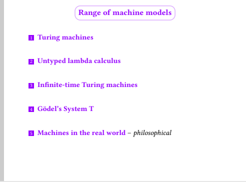
1 **Turing machines**

2 **Untyped lambda calculus**

3 **Infinite-time Turing machines**

4 **Gödel's System T**

5 **Machines in the real world** – *philosophical*



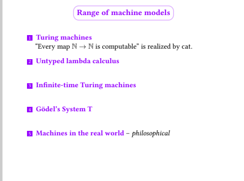
Infinite time Turing machines were introduced by Joel David Hamkins and Andy Lewis. Unlike ordinary Turing machines, they can carry out "more than infinitely many computational steps". Where an ordinary Turing machine fails to terminate, an infinite time Turing machine is put on day ω into a special limit state and can then meaningfully continue.

All functions in Gödel's System T are unconditionally total. Hence unbounded search cannot be implemented in System T.

The idea to apply realizability to machines in the real world is due to Andrej Bauer.

Range of machine models

1. **Turing machines**
   “Every map $\mathbb{N} \to \mathbb{N}$ is computable” is realized by cat.
2. **Untyped lambda calculus**
3. **Infinite-time Turing machines**
4. **Gödel’s System T**
5. **Machines in the real world** – *philosophical*


Range of machine models
1 Turing machines
“Every map $\mathbb{N} \to \mathbb{N}$ is computable” is realized by cat.
2 Untyped lambda calculus
3 Infinite-time Turing machines
4 Gödel’s System T
5 Machines in the real world – *philosophical*

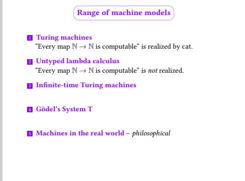
Infinite time Turing machines were introduced by Joel David Hamkins and Andy Lewis. Unlike ordinary Turing machines, they can carry out “more than infinitely many computational steps”. Where an ordinary Turing machine fails to terminate, an infinite time Turing machine is put on day ω into a special limit state and can then meaningfully continue.

All functions in Gödel’s System T are unconditionally total. Hence unbounded search cannot be implemented in System T.

The idea to apply realizability to machines in the real world is due to Andrej Bauer.

Range of machine models

1. **Turing machines**
   "Every map $\mathbb{N} \to \mathbb{N}$ is computable" is realized by cat.
2. **Untyped lambda calculus**
   "Every map $\mathbb{N} \to \mathbb{N}$ is computable" is *not* realized.
3. **Infinite-time Turing machines**
4. **Gödel's System T**
5. **Machines in the real world** – *philosophical*


Range of machine models

1. Turing machines
   "Every map $\mathbb{N} \to \mathbb{N}$ is computable" is realized by cat.
2. Untyped lambda calculus
   "Every map $\mathbb{N} \to \mathbb{N}$ is computable" is *not* realized.
3. Infinite-time Turing machines
4. Gödel's System T
5. Machines in the real world – *philosophical*

Infinite time Turing machines were introduced by Joel David Hamkins and Andy Lewis. Unlike ordinary Turing machines, they can carry out "more than infinitely many computational steps". Where an ordinary Turing machine fails to terminate, an infinite time Turing machine is put on day ω into a special limit state and can then meaningfully continue.

All functions in Gödel's System T are unconditionally total. Hence unbounded search cannot be implemented in System T.

The idea to apply realizability to machines in the real world is due to Andrej Bauer.

Range of machine models

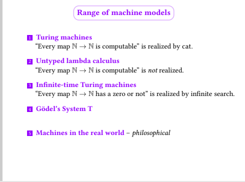
1. **Turing machines**
   "Every map $\mathbb{N} \to \mathbb{N}$ is computable" is realized by cat.

2. **Untyped lambda calculus**
   "Every map $\mathbb{N} \to \mathbb{N}$ is computable" is *not* realized.

3. **Infinite-time Turing machines**
   "Every map $\mathbb{N} \to \mathbb{N}$ has a zero or not" is realized by infinite search.

4. **Gödel's System T**

5. **Machines in the real world** – *philosophical*



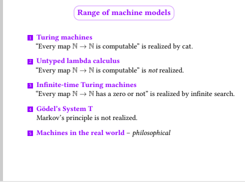
Infinite time Turing machines were introduced by Joel David Hamkins and Andy Lewis. Unlike ordinary Turing machines, they can carry out "more than infinitely many computational steps". Where an ordinary Turing machine fails to terminate, an infinite time Turing machine is put on day ω into a special limit state and can then meaningfully continue.

All functions in Gödel's System T are unconditionally total. Hence unbounded search cannot be implemented in System T.

The idea to apply realizability to machines in the real world is due to Andrej Bauer.

Range of machine models

1. **Turing machines**
   "Every map $\mathbb{N} \to \mathbb{N}$ is computable" is realized by cat.
2. **Untyped lambda calculus**
   "Every map $\mathbb{N} \to \mathbb{N}$ is computable" is *not* realized.
3. **Infinite-time Turing machines**
   "Every map $\mathbb{N} \to \mathbb{N}$ has a zero or not" is realized by infinite search.
4. **Gödel's System T**
   Markov's principle is not realized.
5. **Machines in the real world** – *philosophical*


Range of machine models

1. Turing machines
   "Every map $\mathbb{N} \to \mathbb{N}$ is computable" is realized by cat.
2. Untyped lambda calculus
   "Every map $\mathbb{N} \to \mathbb{N}$ is computable" is *not* realized.
3. Infinite-time Turing machines
   "Every map $\mathbb{N} \to \mathbb{N}$ has a zero or not" is realized by infinite search.
4. Gödel's System T
   Markov's principle is not realized.
5. Machines in the real world – *philosophical*

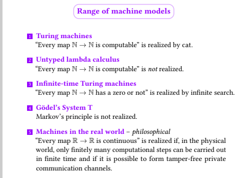
Infinite time Turing machines were introduced by Joel David Hamkins and Andy Lewis. Unlike ordinary Turing machines, they can carry out "more than infinitely many computational steps". Where an ordinary Turing machine fails to terminate, an infinite time Turing machine is put on day ω into a special limit state and can then meaningfully continue.

All functions in Gödel's System T are unconditionally total. Hence unbounded search cannot be implemented in System T.

The idea to apply realizability to machines in the real world is due to Andrej Bauer.

Range of machine models

1. **Turing machines**
   “Every map $\mathbb{N} \to \mathbb{N}$ is computable” is realized by cat.
2. **Untyped lambda calculus**
   “Every map $\mathbb{N} \to \mathbb{N}$ is computable” is *not* realized.
3. **Infinite-time Turing machines**
   “Every map $\mathbb{N} \to \mathbb{N}$ has a zero or not” is realized by infinite search.
4. **Gödel’s System T**
   Markov’s principle is not realized.
5. **Machines in the real world** – *philosophical*
   “Every map $\mathbb{R} \to \mathbb{R}$ is continuous” is realized if, in the physical world, only finitely many computational steps can be carried out in finite time and if it is possible to form tamper-free private communication channels.


Range of machine models

1. Turing machines
   “Every map $\mathbb{N} \to \mathbb{N}$ is computable” is realized by cat.
2. Untyped lambda calculus
   “Every map $\mathbb{N} \to \mathbb{N}$ is computable” is *not* realized.
3. Infinite-time Turing machines
   “Every map $\mathbb{N} \to \mathbb{N}$ has a zero or not” is realized by infinite search.
4. Gödel’s System T
   Markov’s principle is not realized.
5. Machines in the real world – *philosophical*
   “Every map $\mathbb{R} \to \mathbb{R}$ is continuous” is realized if, in the physical world, only finitely many computational steps can be carried out in finite time and if it is possible to form tamper-free private communication channels.

Infinite time Turing machines were introduced by Joel David Hamkins and Andy Lewis. Unlike ordinary Turing machines, they can carry out “more than infinitely many computational steps”. Where an ordinary Turing machine fails to terminate, an infinite time Turing machine is put on day ω into a special limit state and can then meaningfully continue.

All functions in Gödel’s System T are unconditionally total. Hence unbounded search cannot be implemented in System T.

The idea to apply realizability to machines in the real world is due to Andrej Bauer.

A classical logic fairy tale

Narrator. Once upon a time, in a kingdom far, far away, the queen of the land and of all Möbius strips called for her royal philosopher.

Queen. Philosopher! I ask you to carry out the following order. Get me the Philosopher's Stone, or alternatively find out how one could produce arbitrary amounts of gold with it!

Philosopher. But my queen! I haven't studied anything useful! How could I fulfill this order?

Queen. That is not my concern. I'll see you again tomorrow. Should you not accomplish the task, I will take your head off.

Narrator. After a long and wakeful night the philosopher was called to the queen again.

Queen. Tell me! What do you have to report?

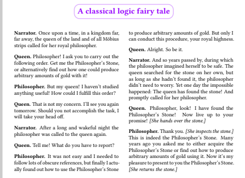
Philosopher. It was not easy and I needed to follow lots of obscure references, but finally I actually found out how to use the Philosopher's Stone to produce arbitrary amounts of gold. But only I can conduct this procedure, your royal highness.

Queen. Alright. So be it.

Narrator. And so years passed by, during which the philosopher imagined herself to be safe. The queen searched for the stone on her own, but as long as she hadn't found it, the philosopher didn't need to worry. Yet one day the impossible happened: The queen has found the stone! And promptly called for her philosopher.

Queen. Philosopher, look! I have found the Philosopher's Stone! Now live up to your promise! *[She hands over the stone.]*

Philosopher. Thank you. *[She inspects the stone.]* This is indeed the Philosopher's Stone. Many years ago you asked me to either acquire the Philosopher's Stone or find out how to produce arbitrary amounts of gold using it. Now it's my pleasure to present to you the Philosopher's Stone. *[She returns the stone.]*


A classical logic fairy tale

Narrator. Once upon a time, in a kingdom far, far away, the queen of the land and of all Möbius strips called for her royal philosopher.

Queen. Philosopher! I ask you to carry out the following order. Get me the Philosopher's Stone, or alternatively find out how one could produce arbitrary amounts of gold with it!

Philosopher. But my queen! I haven't studied anything useful! How could I fulfill this order?

Queen. That is not my concern. I'll see you again tomorrow. Should you not accomplish the task, I will take your head off.

Narrator. After a long and wakeful night the philosopher was called to the queen again.

Queen. Tell me! What do you have to report?

Philosopher. It was not easy and I needed to follow lots of obscure references, but finally I actually found out how to use the Philosopher's Stone to produce arbitrary amounts of gold. But only I can conduct this procedure, your royal highness.

Queen. Alright. So be it.

Narrator. And so years passed by, during which the philosopher imagined herself to be safe. The queen searched for the stone on her own, but as long as she hadn't found it, the philosopher didn't need to worry. Yet one day the impossible happened: The queen has found the stone! And promptly called for her philosopher.

Queen. Philosopher, look! I have found the Philosopher's Stone! Now live up to your promise! *[She hands over the stone.]*

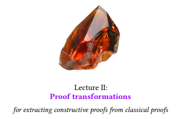
Philosopher. Thank you. *[She inspects the stone.]* This is indeed the Philosopher's Stone. Many years ago you asked me to either acquire the Philosopher's Stone or find out how to produce arbitrary amounts of gold using it. Now it's my pleasure to present to you the Philosopher's Stone. *[She returns the stone.]*

Adapted from Edward Yang's blog.

Lecture II:

Proof transformations

for extracting constructive proofs from classical proofs

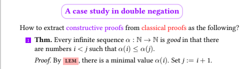


A case study in double negation

How to extract constructive proofs from classical proofs as the following?

1. **Thm.** Every infinite sequence $\alpha : \mathbb{N} \to \mathbb{N}$ is *good* in that there are numbers $i < j$ such that $\alpha(i) \leq \alpha(j)$.

   Proof. By LEM, there is a minimal value $\alpha(i)$. Set $j := i + 1$.


A case study in double negation

How to extract constructive proofs from classical proofs as the following?

1. **Thm.** Every infinite sequence $\alpha : \mathbb{N} \to \mathbb{N}$ is *good* in that there are numbers $i < j$ such that $\alpha(i) \leq \alpha(j)$.

   Proof. By LEM, there is a minimal value $\alpha(i)$. Set $j := i + 1$.

A case study in double negation

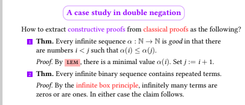
How to extract constructive proofs from classical proofs as the following?

1. **Thm.** Every infinite sequence $\alpha : \mathbb{N} \to \mathbb{N}$ is *good* in that there are numbers $i < j$ such that $\alpha(i) \leq \alpha(j)$.

   Proof. By LEM, there is a minimal value $\alpha(i)$. Set $j := i + 1$.

2. **Thm.** Every infinite binary sequence contains repeated terms.

   Proof. By the infinite box principle, infinitely many terms are zeros or are ones. In either case the claim follows.


A case study in double negation

How to extract constructive proofs from classical proofs as the following?

1. **Thm.** Every infinite sequence $\alpha : \mathbb{N} \to \mathbb{N}$ is *good* in that there are numbers $i < j$ such that $\alpha(i) \leq \alpha(j)$.

   Proof. By LEM, there is a minimal value $\alpha(i)$. Set $j := i + 1$.

2. **Thm.** Every infinite binary sequence contains repeated terms.

   Proof. By the infinite box principle, infinitely many terms are zeros or are ones. In either case the claim follows.

The two displayed proofs seem to be devoid of computational content: Computationally, it is neither possible to determine the minimal element of an infinite sequence nor to determine whether a binary sequence contains infinitely many zeros or infinitely many ones. These claims have no realizer.

However, contrary to expectations, the two proofs do contain an obscured constructive core, and this core can be uncovered by the two logical metatheorems presented in this lecture, the double-negation embedding and *Friedman's trick*. Details are in Thierry Coquand's Computational Content of Classical Logic.

A case study in double negation

How to extract constructive proofs from classical proofs as the following?

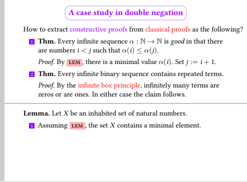
1. **Thm.** Every infinite sequence $\alpha : \mathbb{N} \to \mathbb{N}$ is *good* in that there are numbers $i < j$ such that $\alpha(i) \leq \alpha(j)$.

   Proof. By LEM, there is a minimal value $\alpha(i)$. Set $j := i + 1$.
2. **Thm.** Every infinite binary sequence contains repeated terms.

   Proof. By the infinite box principle, infinitely many terms are zeros or are ones. In either case the claim follows.

Lemma. Let X be an inhabited set of natural numbers.

1. Assuming LEM, the set X contains a minimal element.


A case study in double negation

How to extract constructive proofs from classical proofs as the following?

1. **Thm.** Every infinite sequence $\alpha : \mathbb{N} \to \mathbb{N}$ is *good* in that there are numbers $i < j$ such that $\alpha(i) \leq \alpha(j)$.

   Proof. By LEM, there is a minimal value $\alpha(i)$. Set $j := i + 1$.
2. **Thm.** Every infinite binary sequence contains repeated terms.

   Proof. By the infinite box principle, infinitely many terms are zeros or are ones. In either case the claim follows.

Lemma. Let X be an inhabited set of natural numbers.

1. Assuming LEM, the set X contains a minimal element.

The two displayed proofs seem to be devoid of computational content: Computationally, it is neither possible to determine the minimal element of an infinite sequence nor to determine whether a binary sequence contains infinitely many zeros or infinitely many ones. These claims have no realizer.

However, contrary to expectations, the two proofs do contain an obscured constructive core, and this core can be uncovered by the two logical metatheorems presented in this lecture, the double-negation embedding and *Friedman's trick*. Details are in Thierry Coquand's Computational Content of Classical Logic.

To set the stage, we analyze the lemma used in the first proof. The statement "every inhabited set of natural numbers contains a minimal element" is not realizable and hence does not directly have computational content. In fact, this statement is equivalent to the law of excluded middle. Correspondingly, it seems that proofs using it cannot ever be constructivized. However, as we will see, much more important for extraction of constructive content is the form of the asserted end results than the form of auxiliary lemmas used in a proof.

A case study in double negation

How to extract constructive proofs from classical proofs as the following?

1. **Thm.** Every infinite sequence $\alpha : \mathbb{N} \to \mathbb{N}$ is *good* in that there are numbers $i < j$ such that $\alpha(i) \leq \alpha(j)$.

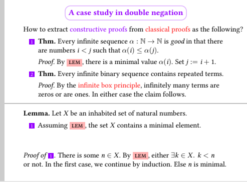
   Proof. By LEM, there is a minimal value $\alpha(i)$. Set $j := i + 1$.
2. **Thm.** Every infinite binary sequence contains repeated terms.

   Proof. By the infinite box principle, infinitely many terms are zeros or are ones. In either case the claim follows.

Lemma. Let X be an inhabited set of natural numbers.

1. Assuming LEM, the set X contains a minimal element.

Proof of 1. There is some $n \in X$. By LEM, either $\exists k \in X.\ k < n$ or not. In the first case, we continue by induction. Else n is minimal.


A case study in double negation
How to extract constructive proofs from classical proofs as the following?
1 Thm. Every infinite sequence $\alpha : \mathbb{N} \to \mathbb{N}$ is *good* in that there are numbers $i < j$ such that $\alpha(i) \leq \alpha(j)$.
Proof. By LEM, there is a minimal value $\alpha(i)$. Set $j := i + 1$.
2 Thm. Every infinite binary sequence contains repeated terms.
Proof. By the infinite box principle, infinitely many terms are zeros or are ones. In either case the claim follows.
Lemma. Let X be an inhabited set of natural numbers.
1 Assuming LEM, the set X contains a minimal element.
Proof of 1. There is some $n \in X$. By LEM, either $\exists k \in X.\ k < n$ or not. In the first case, we continue by induction. Else n is minimal.

The two displayed proofs seem to be devoid of computational content: Computationally, it is neither possible to determine the minimal element of an infinite sequence nor to determine whether a binary sequence contains infinitely many zeros or infinitely many ones. These claims have no realizer.

However, contrary to expectations, the two proofs do contain an obscured constructive core, and this core can be uncovered by the two logical metatheorems presented in this lecture, the double-negation embedding and *Friedman's trick*. Details are in Thierry Coquand's Computational Content of Classical Logic.

To set the stage, we analyze the lemma used in the first proof. The statement "every inhabited set of natural numbers contains a minimal element" is not realizable and hence does not directly have computational content. In fact, this statement is equivalent to the law of excluded middle. Correspondingly, it seems that proofs using it cannot ever be constructivized. However, as we will see, much more important for extraction of constructive content is the form of the asserted end results than the form of auxiliary lemmas used in a proof.

A case study in double negation

How to extract constructive proofs from classical proofs as the following?

1 **Thm.** Every infinite sequence $\alpha : \mathbb{N} \to \mathbb{N}$ is *good* in that there are numbers $i < j$ such that $\alpha(i) \leq \alpha(j)$.

Proof. By LEM, there is a minimal value $\alpha(i)$. Set $j := i + 1$.

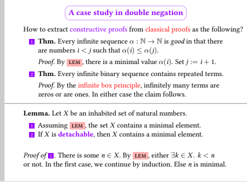
2 **Thm.** Every infinite binary sequence contains repeated terms.

Proof. By the infinite box principle, infinitely many terms are zeros or are ones. In either case the claim follows.

Lemma. Let X be an inhabited set of natural numbers.

1 Assuming LEM, the set X contains a minimal element.
2 If X is **detachable**, then X contains a minimal element.

Proof of 1. There is some $n \in X$. By LEM, either $\exists k \in X.\ k < n$ or not. In the first case, we continue by induction. Else n is minimal.


A case study in double negation

How to extract constructive proofs from classical proofs as the following?

1 **Thm.** Every infinite sequence $\alpha : \mathbb{N} \to \mathbb{N}$ is *good* in that there are numbers $i < j$ such that $\alpha(i) \leq \alpha(j)$.

Proof. By LEM, there is a minimal value $\alpha(i)$. Set $j := i + 1$.

2 **Thm.** Every infinite binary sequence contains repeated terms.

Proof. By the infinite box principle, infinitely many terms are zeros or are ones. In either case the claim follows.

Lemma. Let X be an inhabited set of natural numbers.

1 Assuming LEM, the set X contains a minimal element.
2 If X is detachable, then X contains a minimal element.

Proof of 1. There is some $n \in X$. By LEM, either $\exists k \in X.\ k < n$ or not. In the first case, we continue by induction. Else n is minimal.

A set X of natural numbers is *detachable* if and only if for every number $n \in \mathbb{N}$, either $n \in X$ or $n \notin X$. In classical mathematics, every set of natural numbers is detachable. Constructively, detachability is a nontrivial and computationally meaningful condition. A realizer for detachability of a set X is a machine which reads a number n as input and determines whether n belongs to X or not (outputting also corresponding realizers).

A case study in double negation

How to extract constructive proofs from classical proofs as the following?

1. **Thm.** Every infinite sequence $\alpha : \mathbb{N} \to \mathbb{N}$ is *good* in that there are numbers $i < j$ such that $\alpha(i) \leq \alpha(j)$.

   Proof. By LEM, there is a minimal value $\alpha(i)$. Set $j := i + 1$.

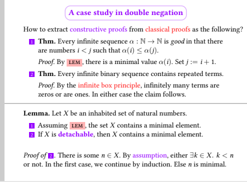
2. **Thm.** Every infinite binary sequence contains repeated terms.

   Proof. By the infinite box principle, infinitely many terms are zeros or are ones. In either case the claim follows.

Lemma. Let X be an inhabited set of natural numbers.

1. Assuming LEM, the set X contains a minimal element.
2. If X is **detachable**, then X contains a minimal element.

Proof of 2. There is some $n \in X$. By assumption, either $\exists k \in X.\ k < n$ or not. In the first case, we continue by induction. Else n is minimal.


A case study in double negation
How to extract constructive proofs from classical proofs as the following?
1 Thm. Every infinite sequence $\alpha : \mathbb{N} \to \mathbb{N}$ is *good* in that there are numbers $i < j$ such that $\alpha(i) \leq \alpha(j)$.
Proof. By LEM, there is a minimal value $\alpha(i)$. Set $j := i + 1$.
2 Thm. Every infinite binary sequence contains repeated terms.
Proof. By the infinite box principle, infinitely many terms are zeros or are ones. In either case the claim follows.
Lemma. Let X be an inhabited set of natural numbers.
1 Assuming LEM, the set X contains a minimal element.
2 If X is detachable, then X contains a minimal element.
Proof of 2. There is some $n \in X$. By assumption, either $\exists k \in X.\ k < n$ or not. In the first case, we continue by induction. Else n is minimal.

A set X of natural numbers is *detachable* if and only if for every number $n \in \mathbb{N}$, either $n \in X$ or $n \notin X$. In classical mathematics, every set of natural numbers is detachable. Constructively, detachability is a nontrivial and computationally meaningful condition. A realizer for detachability of a set X is a machine which reads a number n as input and determines whether n belongs to X or not (outputting also corresponding realizers).

A case study in double negation

How to extract constructive proofs from classical proofs as the following?

1. **Thm.** Every infinite sequence $\alpha : \mathbb{N} \to \mathbb{N}$ is *good* in that there are numbers $i < j$ such that $\alpha(i) \leq \alpha(j)$.

   Proof. By LEM, there is a minimal value $\alpha(i)$. Set $j := i + 1$.

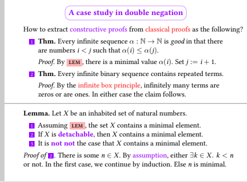
2. **Thm.** Every infinite binary sequence contains repeated terms.

   Proof. By the infinite box principle, infinitely many terms are zeros or are ones. In either case the claim follows.

Lemma. Let X be an inhabited set of natural numbers.

1. Assuming LEM, the set X contains a minimal element.
2. If X is **detachable**, then X contains a minimal element.
3. It is **not not** the case that X contains a minimal element.

Proof of 2. There is some $n \in X$. By assumption, either $\exists k \in X.\ k < n$ or not. In the first case, we continue by induction. Else n is minimal.


A case study in double negation

How to extract constructive proofs from classical proofs as the following?

1. **Thm.** Every infinite sequence $\alpha : \mathbb{N} \to \mathbb{N}$ is *good* in that there are numbers $i < j$ such that $\alpha(i) \leq \alpha(j)$.

   Proof. By LEM, there is a minimal value $\alpha(i)$. Set $j := i + 1$.

2. **Thm.** Every infinite binary sequence contains repeated terms.

   Proof. By the infinite box principle, infinitely many terms are zeros or are ones. In either case the claim follows.

Lemma. Let X be an inhabited set of natural numbers.

1. Assuming LEM, the set X contains a minimal element.
2. If X is **detachable**, then X contains a minimal element.
3. It is **not not** the case that X contains a minimal element.

Proof of 2. There is some $n \in X$. By assumption, either $\exists k \in X.\ k < n$ or not. In the first case, we continue by induction. Else n is minimal.

A set X of natural numbers is *detachable* if and only if for every number $n \in \mathbb{N}$, either $n \in X$ or $n \notin X$. In classical mathematics, every set of natural numbers is detachable. Constructively, detachability is a nontrivial and computationally meaningful condition. A realizer for detachability of a set X is a machine which reads a number n as input and determines whether n belongs to X or not (outputting also corresponding realizers).

Instead of strengthening the assumption, we can also weaken the conclusion. Statement 3 merely asserts that it is impossible that no minimum exists. The *double-negation embedding* reviewed on the next slide explains why it can be expected a priori that this weakening allows us to give a constructive proof.

A case study in double negation

How to extract constructive proofs from classical proofs as the following?

1. **Thm.** Every infinite sequence $\alpha : \mathbb{N} \to \mathbb{N}$ is *good* in that there are numbers $i < j$ such that $\alpha(i) \leq \alpha(j)$.

   Proof. By LEM, there is a minimal value $\alpha(i)$. Set $j := i + 1$.

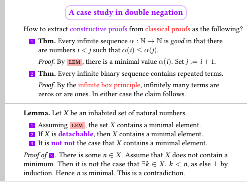
2. **Thm.** Every infinite binary sequence contains repeated terms.

   Proof. By the infinite box principle, infinitely many terms are zeros or are ones. In either case the claim follows.

Lemma. Let X be an inhabited set of natural numbers.

1. Assuming LEM, the set X contains a minimal element.
2. If X is **detachable**, then X contains a minimal element.
3. It is **not not** the case that X contains a minimal element.

Proof of 3. There is some $n \in X$. Assume that X does not contain a minimum. Then it is not the case that $\exists k \in X.\ k < n$, as else $\perp$ by induction. Hence n is minimal. This is a contradiction.


A case study in double negation
How to extract constructive proofs from classical proofs as the following?
1 Thm. Every infinite sequence $\alpha : \mathbb{N} \to \mathbb{N}$ is *good* in that there are numbers $i < j$ such that $\alpha(i) \leq \alpha(j)$.
Proof. By LEM, there is a minimal value $\alpha(i)$. Set $j := i + 1$.
2 Thm. Every infinite binary sequence contains repeated terms.
Proof. By the infinite box principle, infinitely many terms are zeros or are ones. In either case the claim follows.
Lemma. Let X be an inhabited set of natural numbers.
1 Assuming LEM, the set X contains a minimal element.
2 If X is detachable, then X contains a minimal element.
3 It is not not the case that X contains a minimal element.
Proof of 3. There is some $n \in X$. Assume that X does not contain a minimum. Then it is not the case that $\exists k \in X.\ k < n$, as else $\perp$ by induction. Hence n is minimal. This is a contradiction.

A set X of natural numbers is *detachable* if and only if for every number $n \in \mathbb{N}$, either $n \in X$ or $n \notin X$. In classical mathematics, every set of natural numbers is detachable. Constructively, detachability is a nontrivial and computationally meaningful condition. A realizer for detachability of a set X is a machine which reads a number n as input and determines whether n belongs to X or not (outputting also corresponding realizers).

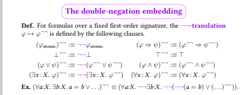
Instead of strengthening the assumption, we can also weaken the conclusion. Statement 3 merely asserts that it is impossible that no minimum exists. The *double-negation embedding* reviewed on the next slide explains why it can be expected a priori that this weakening allows us to give a constructive proof.

The double-negation embedding

Def. For formulas over a fixed first-order signature, the **$\neg\neg$-translation** $\varphi \mapsto \varphi^{\neg\neg}$ is defined by the following clauses.

$$
\begin{array}{rclrcl}
(\varphi_{\text{atomic}})^{\neg\neg} & :\equiv & \neg\neg\varphi_{\text{atomic}} & (\varphi \Rightarrow \psi)^{\neg\neg} & :\equiv & (\varphi^{\neg\neg} \Rightarrow \psi^{\neg\neg}) \\
\bot^{\neg\neg} & :\equiv & \neg\neg\bot & \top^{\neg\neg} & :\equiv & \top \\
(\varphi \vee \psi)^{\neg\neg} & :\equiv & \neg\neg(\varphi^{\neg\neg} \vee \psi^{\neg\neg}) & (\varphi \wedge \psi)^{\neg\neg} & :\equiv & (\varphi^{\neg\neg} \wedge \psi^{\neg\neg}) \\
(\exists x : X.\ \varphi)^{\neg\neg} & :\equiv & \neg\neg(\exists x : X.\ \varphi^{\neg\neg}) & (\forall x : X.\ \varphi)^{\neg\neg} & :\equiv & (\forall x : X.\ \varphi^{\neg\neg})
\end{array}
$$

Ex. $(\forall a{:}X.\, \exists b{:}X.\, a = b \vee \ldots)^{\neg\neg} \equiv (\forall a{:}X.\, \neg\neg\exists b{:}X.\, \neg\neg(\neg\neg(a = b) \vee (\ldots)^{\neg\neg}))$.


The double-negation embedding

Def. For formulas over a fixed first-order signature, the **$\neg\neg$-translation** $\varphi \mapsto \varphi^{\neg\neg}$ is defined by the following clauses.

$(\varphi_{\text{atomic}})^{\neg\neg} :\equiv \neg\neg\varphi_{\text{atomic}}$ $(\varphi \Rightarrow \psi)^{\neg\neg} :\equiv (\varphi^{\neg\neg} \Rightarrow \psi^{\neg\neg})$
$\bot^{\neg\neg} :\equiv \neg\neg\bot$ $\top^{\neg\neg} :\equiv \top$
$(\varphi \vee \psi)^{\neg\neg} :\equiv \neg\neg(\varphi^{\neg\neg} \vee \psi^{\neg\neg})$ $(\varphi \wedge \psi)^{\neg\neg} :\equiv (\varphi^{\neg\neg} \wedge \psi^{\neg\neg})$
$(\exists x : X.\ \varphi)^{\neg\neg} :\equiv \neg\neg(\exists x : X.\ \varphi^{\neg\neg})$ $(\forall x : X.\ \varphi)^{\neg\neg} :\equiv (\forall x : X.\ \varphi^{\neg\neg})$

Ex. $(\forall a{:}X.\, \exists b{:}X.\, a = b \vee \ldots)^{\neg\neg} \equiv (\forall a{:}X.\, \neg\neg\exists b{:}X.\, \neg\neg(\neg\neg(a = b) \vee (\ldots)^{\neg\neg}))$.

Minimal logic is intuitionistic logic minus *ex falsum quodlibet* ($\bot \vdash_{\vec{x}} \varphi$).

The double-negation embedding builds on a fundamental observation: While the law of excluded middle is not available in minimal or intuitionistic logic, the double negation of every instance is—for every formula φ, the formula $\neg\neg(\varphi \vee \neg\varphi)$ is an intuitionistic tautology:

> *In order to show $\neg\neg(\varphi \vee \neg\varphi)$, assume $\neg(\varphi \vee \neg\varphi)$ and deduce $\bot$.*
>
> *As a preparatory step, we verify $\neg\varphi$: If φ, then in particular $\varphi \vee \neg\varphi$, hence $\bot$ by assumption.*
>
> *Having established $\neg\varphi$, we notice that also $\varphi \vee \neg\varphi$, hence $\bot$ by assumption.*

Also, a doubly negated statement is not a dead end. Instead, we can continue reasoning, by the following tautology (“Kleisli extension”, “monadic bind”, “nucleus axiom”):

$$(\neg\neg\varphi \wedge (\varphi \Rightarrow \neg\neg\psi)) \Longrightarrow \neg\neg\psi$$

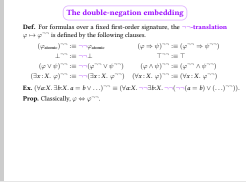
The double-negation embedding

Def. For formulas over a fixed first-order signature, the **¬¬-translation** $\varphi \mapsto \varphi^{\neg\neg}$ is defined by the following clauses.

$$\begin{array}{rclrcl}
(\varphi_{\text{atomic}})^{\neg\neg} & :\equiv & \neg\neg\varphi_{\text{atomic}} & (\varphi \Rightarrow \psi)^{\neg\neg} & :\equiv & (\varphi^{\neg\neg} \Rightarrow \psi^{\neg\neg}) \\
\bot^{\neg\neg} & :\equiv & \neg\neg\bot & \top^{\neg\neg} & :\equiv & \top \\
(\varphi \vee \psi)^{\neg\neg} & :\equiv & \neg\neg(\varphi^{\neg\neg} \vee \psi^{\neg\neg}) & (\varphi \wedge \psi)^{\neg\neg} & :\equiv & (\varphi^{\neg\neg} \wedge \psi^{\neg\neg}) \\
(\exists x : X.\ \varphi)^{\neg\neg} & :\equiv & \neg\neg(\exists x : X.\ \varphi^{\neg\neg}) & (\forall x : X.\ \varphi)^{\neg\neg} & :\equiv & (\forall x : X.\ \varphi^{\neg\neg})
\end{array}$$

Ex. $(\forall a{:}X.\, \exists b{:}X.\, a = b \vee \ldots)^{\neg\neg} \equiv (\forall a{:}X.\, \neg\neg\exists b{:}X.\, \neg\neg(\neg\neg(a = b) \vee (\ldots)^{\neg\neg}))$.

Prop. Classically, $\varphi \Leftrightarrow \varphi^{\neg\neg}$.


The double-negation embedding

Def. For formulas over a fixed first-order signature, the **¬¬-translation** $\varphi \mapsto \varphi^{\neg\neg}$ is defined by the following clauses.

$(\varphi_{\text{atomic}})^{\neg\neg} :\equiv \neg\neg\varphi_{\text{atomic}}$ $(\varphi \Rightarrow \psi)^{\neg\neg} :\equiv (\varphi^{\neg\neg} \Rightarrow \psi^{\neg\neg})$
$\bot^{\neg\neg} :\equiv \neg\neg\bot$ $\top^{\neg\neg} :\equiv \top$
$(\varphi \vee \psi)^{\neg\neg} :\equiv \neg\neg(\varphi^{\neg\neg} \vee \psi^{\neg\neg})$ $(\varphi \wedge \psi)^{\neg\neg} :\equiv (\varphi^{\neg\neg} \wedge \psi^{\neg\neg})$
$(\exists x : X.\ \varphi)^{\neg\neg} :\equiv \neg\neg(\exists x : X.\ \varphi^{\neg\neg})$ $(\forall x : X.\ \varphi)^{\neg\neg} :\equiv (\forall x : X.\ \varphi^{\neg\neg})$

Ex. $(\forall a{:}X.\, \exists b{:}X.\, a = b \vee \ldots)^{\neg\neg} \equiv (\forall a{:}X.\, \neg\neg\exists b{:}X.\, \neg\neg(\neg\neg(a = b) \vee (\ldots)^{\neg\neg}))$.

Prop. Classically, $\varphi \Leftrightarrow \varphi^{\neg\neg}$.

Minimal logic is intuitionistic logic minus *ex falsum quodlibet* ($\bot \nvdash \varphi$).

The double-negation embedding builds on a fundamental observation: While the law of excluded middle is not available in minimal or intuitionistic logic, the double negation of every instance is—for every formula φ, the formula $\neg\neg(\varphi \vee \neg\varphi)$ is an intuitionistic tautology:

> *In order to show* $\neg\neg(\varphi \vee \neg\varphi)$*, assume* $\neg(\varphi \vee \neg\varphi)$ *and deduce* $\bot$*.*
>
> *As a preparatory step, we verify* $\neg\varphi$*: If* φ*, then in particular* $\varphi \vee \neg\varphi$*, hence* $\bot$ *by assumption.*
>
> *Having established* $\neg\varphi$*, we notice that also* $\varphi \vee \neg\varphi$*, hence* $\bot$ *by assumption.*

Also, a doubly negated statement is not a dead end. Instead, we can continue reasoning, by the following tautology (“Kleisli extension”, “monadic bind”, “nucleus axiom”):

$$(\neg\neg\varphi \wedge (\varphi \Rightarrow \neg\neg\psi)) \Longrightarrow \neg\neg\psi$$

The double-negation embedding

Def. For formulas over a fixed first-order signature, the **$\neg\neg$-translation** $\varphi \mapsto \varphi^{\neg\neg}$ is defined by the following clauses.

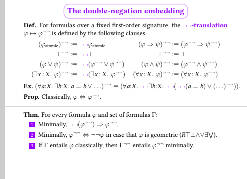
$$\begin{array}{rclcrcl}
(\varphi_{\text{atomic}})^{\neg\neg} & :\equiv & \neg\neg\varphi_{\text{atomic}} & \quad & (\varphi \Rightarrow \psi)^{\neg\neg} & :\equiv & (\varphi^{\neg\neg} \Rightarrow \psi^{\neg\neg}) \\
\bot^{\neg\neg} & :\equiv & \neg\neg\bot & & \top^{\neg\neg} & :\equiv & \top \\
(\varphi \vee \psi)^{\neg\neg} & :\equiv & \neg\neg(\varphi^{\neg\neg} \vee \psi^{\neg\neg}) & & (\varphi \wedge \psi)^{\neg\neg} & :\equiv & (\varphi^{\neg\neg} \wedge \psi^{\neg\neg}) \\
(\exists x\,{:}\,X.\ \varphi)^{\neg\neg} & :\equiv & \neg\neg(\exists x\,{:}\,X.\ \varphi^{\neg\neg}) & & (\forall x\,{:}\,X.\ \varphi)^{\neg\neg} & :\equiv & (\forall x\,{:}\,X.\ \varphi^{\neg\neg})
\end{array}$$

Ex. $(\forall a{:}X.\, \exists b{:}X.\, a = b \vee \ldots)^{\neg\neg} \equiv (\forall a{:}X.\, \neg\neg\exists b{:}X.\, \neg\neg(\neg\neg(a = b) \vee (\ldots)^{\neg\neg}))$.

Prop. Classically, $\varphi \Leftrightarrow \varphi^{\neg\neg}$.

Thm. For every formula φ and set of formulas Γ:

1. Minimally, $\neg\neg(\varphi^{\neg\neg}) \Rightarrow \varphi^{\neg\neg}$.
2. Minimally, $\varphi^{\neg\neg} \Leftrightarrow \neg\neg\varphi$ in case that φ is geometric ($R\top\bot\wedge\vee\exists\bigvee$).
3. If Γ entails φ classically, then $\Gamma^{\neg\neg}$ entails $\varphi^{\neg\neg}$ minimally.



Minimal logic is intuitionistic logic minus *ex falsum quodlibet* ($\bot \nvdash \varphi$).

The double-negation embedding builds on a fundamental observation: While the law of excluded middle is not available in minimal or intuitionistic logic, the double negation of every instance is—for every formula φ, the formula $\neg\neg(\varphi \vee \neg\varphi)$ is an intuitionistic tautology:

> *In order to show $\neg\neg(\varphi \vee \neg\varphi)$, assume $\neg(\varphi \vee \neg\varphi)$ and deduce $\bot$.*
>
> *As a preparatory step, we verify $\neg\varphi$: If φ, then in particular $\varphi \vee \neg\varphi$, hence $\bot$ by assumption.*
>
> *Having established $\neg\varphi$, we notice that also $\varphi \vee \neg\varphi$, hence $\bot$ by assumption.*

Also, a doubly negated statement is not a dead end. Instead, we can continue reasoning, by the following tautology (“Kleisli extension”, “monadic bind”, “nucleus axiom”):

$$(\neg\neg\varphi \wedge (\varphi \Rightarrow \neg\neg\psi)) \Longrightarrow \neg\neg\psi$$

The three parts of the theorem are each proven by induction, on the structure of φ for the first two parts and on the structure of derivations for the third part. Carrying this out is an instructive exercise!

The double-negation embedding

Def. For formulas over a fixed first-order signature, the **$\neg\neg$-translation** $\varphi \mapsto \varphi^{\neg\neg}$ is defined by the following clauses.

$$\begin{array}{rclrcl}
(\varphi_{\text{atomic}})^{\neg\neg} & :\equiv & \neg\neg\varphi_{\text{atomic}} & (\varphi \Rightarrow \psi)^{\neg\neg} & :\equiv & (\varphi^{\neg\neg} \Rightarrow \psi^{\neg\neg}) \\
\bot^{\neg\neg} & :\equiv & \neg\neg\bot & \top^{\neg\neg} & :\equiv & \top \\
(\varphi \vee \psi)^{\neg\neg} & :\equiv & \neg\neg(\varphi^{\neg\neg} \vee \psi^{\neg\neg}) & (\varphi \wedge \psi)^{\neg\neg} & :\equiv & (\varphi^{\neg\neg} \wedge \psi^{\neg\neg}) \\
(\exists x : X.\ \varphi)^{\neg\neg} & :\equiv & \neg\neg(\exists x : X.\ \varphi^{\neg\neg}) & (\forall x : X.\ \varphi)^{\neg\neg} & :\equiv & (\forall x : X.\ \varphi^{\neg\neg})
\end{array}$$

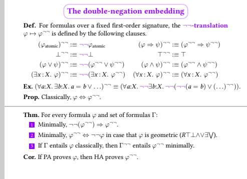
Ex. $(\forall a{:}X.\, \exists b{:}X.\, a = b \vee \ldots)^{\neg\neg} \equiv (\forall a{:}X.\, \neg\neg\exists b{:}X.\, \neg\neg(\neg\neg(a = b) \vee (\ldots)^{\neg\neg}))$.

Prop. Classically, $\varphi \Leftrightarrow \varphi^{\neg\neg}$.

Thm. For every formula φ and set of formulas Γ:

1. Minimally, $\neg\neg(\varphi^{\neg\neg}) \Rightarrow \varphi^{\neg\neg}$.
2. Minimally, $\varphi^{\neg\neg} \Leftrightarrow \neg\neg\varphi$ in case that φ is geometric ($R\top\bot\wedge\vee\exists\bigvee$).
3. If Γ entails φ classically, then $\Gamma^{\neg\neg}$ entails $\varphi^{\neg\neg}$ minimally.

Cor. If PA proves φ, then HA proves $\varphi^{\neg\neg}$.


The double-negation embedding
Def. For formulas over a fixed first-order signature, the ¬¬-translation $\varphi \mapsto \varphi^{\neg\neg}$ is defined by the following clauses.
Ex. $(\forall a{:}X.\, \exists b{:}X.\, a = b \vee \ldots)^{\neg\neg} \equiv (\forall a{:}X.\, \neg\neg\exists b{:}X.\, \neg\neg(\neg\neg(a = b) \vee (\ldots)^{\neg\neg}))$.
Prop. Classically, $\varphi \Leftrightarrow \varphi^{\neg\neg}$.
Thm. For every formula φ and set of formulas Γ:
1 Minimally, $\neg\neg(\varphi^{\neg\neg}) \Rightarrow \varphi^{\neg\neg}$.
2 Minimally, $\varphi^{\neg\neg} \Leftrightarrow \neg\neg\varphi$ in case that φ is geometric ($R\top\bot\wedge\vee\exists\bigvee$).
3 If Γ entails φ classically, then $\Gamma^{\neg\neg}$ entails $\varphi^{\neg\neg}$ minimally.
Cor. If PA proves φ, then HA proves $\varphi^{\neg\neg}$.

The corollary follows from the theorem because HA proves the double-negation translation of every axiom of PA. For instance, the translation of the axiom

$$\forall x : N.\ x + 0 = x$$

is

$$\forall x : N.\ \neg\neg(x + 0 = x),$$

and this weaker statement is provable thanks to the intuitionistic (even minimal) tautology $\varphi \Rightarrow \neg\neg\varphi$.

Similarly, the translation of an instance of the induction scheme

$$\varphi(0) \wedge \big(\forall x : N.\ \varphi(x) \Rightarrow \varphi(S(x))\big) \quad\Longrightarrow\quad \forall x : N.\ \varphi(x)$$

is again an instance of the induction scheme:

$$\varphi(0)^{\neg\neg} \wedge \big(\forall x : N.\ \varphi(x)^{\neg\neg} \Rightarrow \varphi(S(x))^{\neg\neg}\big) \quad\Longrightarrow\quad \forall x : N.\ \varphi(x)^{\neg\neg}$$

The double-negation embedding

Def. For formulas over a fixed first-order signature, the **$\neg\neg$-translation** $\varphi \mapsto \varphi^{\neg\neg}$ is defined by the following clauses.

$$\begin{aligned}
(\varphi_{\text{atomic}})^{\neg\neg} &:\equiv \neg\neg\varphi_{\text{atomic}} & (\varphi \Rightarrow \psi)^{\neg\neg} &:\equiv (\varphi^{\neg\neg} \Rightarrow \psi^{\neg\neg}) \\
\bot^{\neg\neg} &:\equiv \neg\neg\bot & \top^{\neg\neg} &:\equiv \top \\
(\varphi \vee \psi)^{\neg\neg} &:\equiv \neg\neg(\varphi^{\neg\neg} \vee \psi^{\neg\neg}) & (\varphi \wedge \psi)^{\neg\neg} &:\equiv (\varphi^{\neg\neg} \wedge \psi^{\neg\neg}) \\
(\exists x\,{:}\,X.\ \varphi)^{\neg\neg} &:\equiv \neg\neg(\exists x\,{:}\,X.\ \varphi^{\neg\neg}) & (\forall x\,{:}\,X.\ \varphi)^{\neg\neg} &:\equiv (\forall x\,{:}\,X.\ \varphi^{\neg\neg})
\end{aligned}$$

Ex. $(\forall a{:}X.\, \exists b{:}X.\, a = b \vee \ldots)^{\neg\neg} \equiv (\forall a{:}X.\, \neg\neg\exists b{:}X.\, \neg\neg(\neg\neg(a = b) \vee (\ldots)^{\neg\neg}))$.

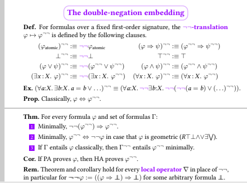
Prop. Classically, $\varphi \Leftrightarrow \varphi^{\neg\neg}$.

Thm. For every formula φ and set of formulas Γ:

1. Minimally, $\neg\neg(\varphi^{\neg\neg}) \Rightarrow \varphi^{\neg\neg}$.
2. Minimally, $\varphi^{\neg\neg} \Leftrightarrow \neg\neg\varphi$ in case that φ is geometric ($R\top\bot\wedge\vee\exists\bigvee$).
3. If Γ entails φ classically, then $\Gamma^{\neg\neg}$ entails $\varphi^{\neg\neg}$ minimally.

Cor. If PA proves φ, then HA proves $\varphi^{\neg\neg}$.

Rem. Theorem and corollary hold for every **local operator** ∇ in place of $\neg\neg$, in particular for $\neg\neg\varphi :\equiv ((\varphi \Rightarrow \perp\!\!\!\perp) \Rightarrow \perp\!\!\!\perp)$ for some arbitrary formula $\perp\!\!\!\perp$.


The double-negation embedding
Def. For formulas over a fixed first-order signature, the $\neg\neg$-translation $\varphi \mapsto \varphi^{\neg\neg}$ is defined by the following clauses.
$(\varphi_{\text{atomic}})^{\neg\neg} :\equiv \neg\neg\varphi_{\text{atomic}}$ $(\varphi \Rightarrow \psi)^{\neg\neg} :\equiv (\varphi^{\neg\neg} \Rightarrow \psi^{\neg\neg})$
$\bot^{\neg\neg} :\equiv \neg\neg\bot$ $\top^{\neg\neg} :\equiv \top$
$(\varphi \vee \psi)^{\neg\neg} :\equiv \neg\neg(\varphi^{\neg\neg} \vee \psi^{\neg\neg})$ $(\varphi \wedge \psi)^{\neg\neg} :\equiv (\varphi^{\neg\neg} \wedge \psi^{\neg\neg})$
$(\exists x{:}X.\ \varphi)^{\neg\neg} :\equiv \neg\neg(\exists x{:}X.\ \varphi^{\neg\neg})$ $(\forall x{:}X.\ \varphi)^{\neg\neg} :\equiv (\forall x{:}X.\ \varphi^{\neg\neg})$
Ex. $(\forall a{:}X.\, \exists b{:}X.\, a = b \vee \ldots)^{\neg\neg} \equiv (\forall a{:}X.\, \neg\neg\exists b{:}X.\, \neg\neg(\neg\neg(a = b) \vee (\ldots)^{\neg\neg}))$.
Prop. Classically, $\varphi \Leftrightarrow \varphi^{\neg\neg}$.
Thm. For every formula φ and set of formulas Γ:
1 Minimally, $\neg\neg(\varphi^{\neg\neg}) \Rightarrow \varphi^{\neg\neg}$.
2 Minimally, $\varphi^{\neg\neg} \Leftrightarrow \neg\neg\varphi$ in case that φ is geometric ($R\top\bot\wedge\vee\exists\bigvee$).
3 If Γ entails φ classically, then $\Gamma^{\neg\neg}$ entails $\varphi^{\neg\neg}$ minimally.
Cor. If PA proves φ, then HA proves $\varphi^{\neg\neg}$.
Rem. Theorem and corollary hold for every local operator ∇ in place of $\neg\neg$, in particular for $\neg\neg\varphi :\equiv ((\varphi \Rightarrow \perp\!\!\!\perp) \Rightarrow \perp\!\!\!\perp)$ for some arbitrary formula $\perp\!\!\!\perp$.

Live demo with Agda code for the theorem on all infinite sequences of natural numbers being good:

1. The first proof there implements the classical proof, postulating the law of excluded middle as an unprovable axiom. Thereby Agda can verify the classical proof to be correct, however trying to run the proof will get stuck on the postulated LEM-oracle.
2. The second proof rewrites the statement of the theorem and its proof by the double-negation translation. The result is a constructive proof which doesn't rely on postulates; however there is still no direct computational content, as the asserted claim is a (doubly) negated statement.
3. In a logical sleight of hand, the third proof imports the second proof but specializes $\perp\!\!\!\perp$, which was an arbitrary constant there, to the asserted claim. The tautology $\neg\neg\perp\!\!\!\perp \Rightarrow \perp\!\!\!\perp$ of minimal logic is then used to escape the double-negation monad and obtain a constructive proof of the full result.

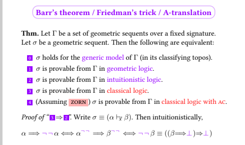
Barr's theorem / Friedman's trick / A-translation

Thm. Let Γ be a set of geometric sequents over a fixed signature. Let σ be a geometric sequent. Then the following are equivalent:

- $\boxed{0}$ σ holds for the generic model of Γ (in its classifying topos).
- $\boxed{1}$ σ is provable from Γ in geometric logic.
- $\boxed{2}$ σ is provable from Γ in intuitionistic logic.
- $\boxed{3}$ σ is provable from Γ in classical logic.
- $\boxed{4}$ (Assuming ZORN) σ is provable from Γ in classical logic with AC.

Proof of "$\boxed{3} \Rightarrow \boxed{2}$". Write $\sigma \equiv (\alpha \vdash_{\vec{x}} \beta)$. Then intuitionistically,

$$\alpha \Longrightarrow \neg\neg\alpha \Longleftrightarrow \alpha^{\neg\neg} \Longrightarrow \beta^{\neg\neg} \Longleftrightarrow \neg\neg\beta \equiv ((\beta \Longrightarrow \bot) \Rightarrow \bot)$$



Barr's theorem / Friedman's trick / A-translation

Thm. Let Γ be a set of geometric sequents over a fixed signature. Let σ be a geometric sequent. Then the following are equivalent:

- $\boxed{0}$ σ holds for the generic model of Γ (in its classifying topos).
- $\boxed{1}$ σ is provable from Γ in geometric logic.
- $\boxed{2}$ σ is provable from Γ in intuitionistic logic.
- $\boxed{3}$ σ is provable from Γ in classical logic.
- $\boxed{4}$ (Assuming ZORN) σ is provable from Γ in classical logic with AC.

Proof of "$\boxed{3} \Rightarrow \boxed{2}$". Write $\sigma \equiv (\alpha \vdash_{\vec{x}} \beta)$. Then intuitionistically,

$\alpha \Longrightarrow \neg\neg\alpha \Longleftrightarrow \alpha^{\neg\neg} \Longrightarrow \beta^{\neg\neg} \Longleftrightarrow \neg\neg\beta \equiv ((\beta \Longrightarrow \bot) \Rightarrow \bot)$

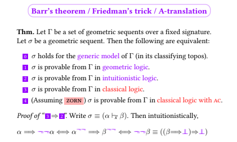
Barr's theorem / Friedman's trick / A-translation

Thm. Let Γ be a set of geometric sequents over a fixed signature. Let σ be a geometric sequent. Then the following are equivalent:

- $\boxed{0}$ σ holds for the generic model of Γ (in its classifying topos).
- $\boxed{1}$ σ is provable from Γ in geometric logic.
- $\boxed{2}$ σ is provable from Γ in intuitionistic logic.
- $\boxed{3}$ σ is provable from Γ in classical logic.
- $\boxed{4}$ (Assuming ZORN) σ is provable from Γ in classical logic with AC.

Proof of "$\boxed{3} \Rightarrow \boxed{2}$". Write $\sigma \equiv (\alpha \vdash_{\vec{x}} \beta)$. Then intuitionistically,

$$\alpha \Longrightarrow \neg\neg\alpha \Longleftrightarrow \alpha^{\neg\neg} \Longrightarrow \beta^{\neg\neg} \Longleftrightarrow \neg\neg\beta \equiv ((\beta \Longrightarrow \bot\!\!\!\bot) \Rightarrow \bot\!\!\!\bot)$$



Barr's theorem / Friedman's trick / A-translation

Thm. Let Γ be a set of geometric sequents over a fixed signature. Let σ be a geometric sequent. Then the following are equivalent:

- $\boxed{0}$ σ holds for the generic model of Γ (in its classifying topos).
- $\boxed{1}$ σ is provable from Γ in geometric logic.
- $\boxed{2}$ σ is provable from Γ in intuitionistic logic.
- $\boxed{3}$ σ is provable from Γ in classical logic.
- $\boxed{4}$ (Assuming ZORN) σ is provable from Γ in classical logic with AC.

Proof of "$\boxed{3} \Rightarrow \boxed{2}$". Write $\sigma \equiv (\alpha \vdash_{\vec{x}} \beta)$. Then intuitionistically,

$\alpha \Longrightarrow \neg\neg\alpha \Longleftrightarrow \alpha^{\neg\neg} \Longrightarrow \beta^{\neg\neg} \Longleftrightarrow \neg\neg\beta \equiv ((\beta \Longrightarrow \bot\!\!\!\bot) \Rightarrow \bot\!\!\!\bot)$

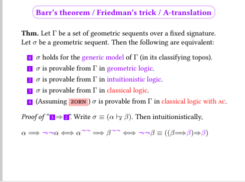
Barr’s theorem / Friedman’s trick / A-translation

Thm. Let Γ be a set of geometric sequents over a fixed signature. Let σ be a geometric sequent. Then the following are equivalent:

- $\boxed{0}$ σ holds for the generic model of Γ (in its classifying topos).
- $\boxed{1}$ σ is provable from Γ in geometric logic.
- $\boxed{2}$ σ is provable from Γ in intuitionistic logic.
- $\boxed{3}$ σ is provable from Γ in classical logic.
- $\boxed{4}$ (Assuming ZORN) σ is provable from Γ in classical logic with AC.

Proof of “$\boxed{3} \Rightarrow \boxed{2}$”. Write $\sigma \equiv (\alpha \vdash_{\vec{x}} \beta)$. Then intuitionistically,

$$\alpha \Longrightarrow \neg\neg\alpha \Longleftrightarrow \alpha^{\neg\neg} \Longrightarrow \beta^{\neg\neg} \Longleftrightarrow \neg\neg\beta \equiv ((\beta\Longrightarrow\beta)\Rightarrow\beta)$$


Barr’s theorem / Friedman’s trick / A-translation

Thm. Let Γ be a set of geometric sequents over a fixed signature. Let σ be a geometric sequent. Then the following are equivalent:

0 σ holds for the generic model of Γ (in its classifying topos).
1 σ is provable from Γ in geometric logic.
2 σ is provable from Γ in intuitionistic logic.
3 σ is provable from Γ in classical logic.
4 (Assuming ZORN) σ is provable from Γ in classical logic with AC.

Proof of “3⇒2”. Write σ ≡ (α ⊢ β). Then intuitionistically,

α ⟹ ¬¬α ⟺ α^¬¬ ⟹ β^¬¬ ⟺ ¬¬β ≡ ((β⟹β)⇒β)

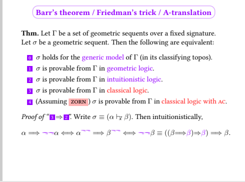
Barr's theorem / Friedman's trick / A-translation

Thm. Let Γ be a set of geometric sequents over a fixed signature. Let σ be a geometric sequent. Then the following are equivalent:

- 0 σ holds for the generic model of Γ (in its classifying topos).
- 1 σ is provable from Γ in geometric logic.
- 2 σ is provable from Γ in intuitionistic logic.
- 3 σ is provable from Γ in classical logic.
- 4 (Assuming ZORN) σ is provable from Γ in classical logic with AC.

Proof of "3 $\Rightarrow$ 2". Write $\sigma \equiv (\alpha \vdash_{\vec{x}} \beta)$. Then intuitionistically,

$$\alpha \Longrightarrow \neg\neg\alpha \Longleftrightarrow \alpha^{\neg\neg} \Longrightarrow \beta^{\neg\neg} \Longleftrightarrow \neg\neg\beta \equiv ((\beta \Longrightarrow \beta) \Rightarrow \beta) \Longrightarrow \beta.$$


Barr's theorem / Friedman's trick / A-translation

Thm. Let Γ be a set of geometric sequents over a fixed signature. Let σ be a geometric sequent. Then the following are equivalent:

0 σ holds for the generic model of Γ (in its classifying topos).
1 σ is provable from Γ in geometric logic.
2 σ is provable from Γ in intuitionistic logic.
3 σ is provable from Γ in classical logic.
4 (Assuming ZORN) σ is provable from Γ in classical logic with AC.

Proof of "3 $\Rightarrow$ 2". Write $\sigma \equiv (\alpha \vdash_{\vec{x}} \beta)$. Then intuitionistically,

$\alpha \Longrightarrow \neg\neg\alpha \Longleftrightarrow \alpha^{\neg\neg} \Longrightarrow \beta^{\neg\neg} \Longleftrightarrow \neg\neg\beta \equiv ((\beta \Longrightarrow \beta) \Rightarrow \beta) \Longrightarrow \beta.$

The proof transformation for "3 $\Rightarrow$ 2" is explicit in nature, feasible in practice and increases the proof length only polynomially. There is also an intriguing alternative approach by Giulio Fellin, Sara Negri and Eugenio Orlandelli employing cut elimination (thereby increasing the proof length substantially, but conceptually beneficial in other aspects). Furthermore, this metatheorem can also be cast in semantic terms:

Every Grothendieck topos admits a cover (a surjective geometric morphism from another Grothendieck topos) by a Grothendieck topos which is boolean. This can be obtained by first adjoining a *generic proposition* χ and then taking sheaves with respect to the modality $((\cdot \Rightarrow \chi) \Rightarrow \chi)$.

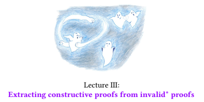
Applied to the classifying topos of Γ, this insight yields a proof of "3 $\Rightarrow$ 0", since pullback along surjective geometric morphisms reflects validity of geometric sequents.

Furthermore, every Grothendieck topos admits a cover by a Grothendieck topos over a complete Boolean algebra. Assuming Zorn's lemma in the metatheory, this topos validates LEM and ZORN, hence AC. This shows "4 $\Rightarrow$ 0".

An introduction to topos-theoretic generic models with a view towards applications in constructive commutative algebra is contained in this survey.

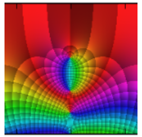
Lecture III:

Extracting constructive proofs from invalid* proofs

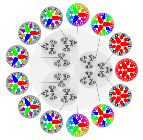


Lecture III:

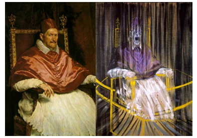
Extracting constructive proofs from invalid* proofs



$\mathbb{C}$



$\mathbb{Q}_p$



$\mathbb{F}_1$

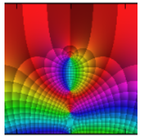
∞

Lecture III:
Extracting constructive proofs from invalid* proofs

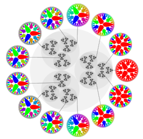
Lecture III:

Extracting constructive proofs from invalid* proofs

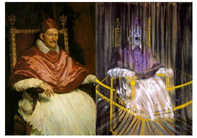
* *higher-order proofs of first-order statements using the assumption that a given (perhaps uncountable) set is countable*



$\mathbb{C}$



$\mathbb{Q}_p$



$\mathbb{F}_1$

∞

Thm. 1. From every proof p of a statement φ such that

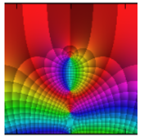
1. the assertion φ is a first-order statement,
2. the proof p is constructive,
3. the proof p is formulated in a certain higher-order system (in particular, the proof may, unlike its end result, freely employ higher-order notions) and
4. for a certain set X appearing in p, the proof assumes that X is countable,

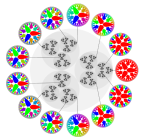
a constructive proof of the same statement φ, set in the same system, but *without requiring the countability assumption* can be extracted.

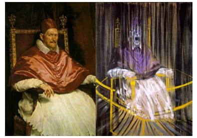
Lecture III:

Extracting constructive proofs from invalid* proofs

* *higher-order proofs of first-order statements using the assumption that a given (perhaps uncountable) set is countable*


$\mathbb{C}$


$\mathbb{Q}_p$


$\mathbb{F}_1$

∞

Thm. 2. From every proof p of a statement φ such that

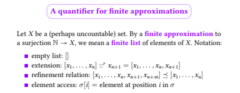
1. the assertion φ is a geometric implication,
2. the proof p is classical (constructive + LEM, but still no ZORN),
3. the proof p is formulated in a certain higher-order system (in particular, the proof may, unlike its end result, freely employ higher-order notions) and
4. for a certain set X appearing in p, the proof assumes that X is countable,

a classical proof of the same statement φ, set in the same system, but *without requiring the countability assumption* can be extracted.

A quantifier for finite approximations

Let X be a (perhaps uncountable) set. By a **finite approximation** to a surjection $\mathbb{N} \twoheadrightarrow X$, we mean a **finite list** of elements of X. Notation:

- empty list: $[]$
- extension: $[x_1, \ldots, x_n] ::^r x_{n+1} = [x_1, \ldots, x_n, x_{n+1}]$
- refinement relation: $[x_1, \ldots, x_n, x_{n+1}, x_{n+m}] \preceq [x_1, \ldots, x_n]$
- element access: $\sigma[i]$ = element at position i in σ


A quantifier for finite approximations

Let X be a (perhaps uncountable) set. By a **finite approximation** to a surjection $\mathbb{N} \twoheadrightarrow X$, we mean a **finite list** of elements of X. Notation:

- empty list: $[]$
- extension: $[x_1, \ldots, x_n] ::^r x_{n+1} = [x_1, \ldots, x_n, x_{n+1}]$
- refinement relation: $[x_1, \ldots, x_n, x_{n+1}, x_{n+m}] \preceq [x_1, \ldots, x_n]$
- element access: $\sigma[i]$ = element at position i in σ

A quantifier for finite approximations

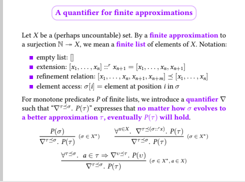
Let X be a (perhaps uncountable) set. By a **finite approximation** to a surjection $\mathbb{N} \twoheadrightarrow X$, we mean a **finite list** of elements of X. Notation:

- empty list: $[]$
- extension: $[x_1, \ldots, x_n] ::^r x_{n+1} = [x_1, \ldots, x_n, x_{n+1}]$
- refinement relation: $[x_1, \ldots, x_n, x_{n+1}, x_{n+m}] \preceq [x_1, \ldots, x_n]$
- element access: $\sigma[i]$ = element at position i in σ

For monotone predicates P of finite lists, we introduce a **quantifier** ∇ such that "$\nabla^{\tau \preceq \sigma}.\ P(\tau)$" expresses that **no matter how σ evolves to a better approximation τ, eventually $P(\tau)$ will hold**.

$$\frac{P(\sigma)}{\nabla^{\tau \preceq \sigma}.\ P(\tau)}\ (\sigma \in X^*) \qquad \frac{\forall^{x \in X}.\ \nabla^{\tau \preceq (\sigma ::^r x)}.\ P(\tau)}{\nabla^{\tau \preceq \sigma}.\ P(\tau)}\ (\sigma \in X^*)$$

$$\frac{\forall^{\tau \preceq \sigma}.\ a \in \tau \Rightarrow \nabla^{\upsilon \preceq \tau}.\ P(\upsilon)}{\nabla^{\tau \preceq \sigma}.\ P(\tau)}\ (\sigma \in X^*, a \in X)$$


A quantifier for finite approximations

Let X be a (perhaps uncountable) set. By a finite approximation to a surjection $\mathbb{N} \twoheadrightarrow X$, we mean a finite list of elements of X. Notation:

- empty list: $[]$
- extension: $[x_1, \ldots, x_n] ::^r x_{n+1} = [x_1, \ldots, x_n, x_{n+1}]$
- refinement relation: $[x_1, \ldots, x_n, x_{n+1}, x_{n+m}] \preceq [x_1, \ldots, x_n]$
- element access: $\sigma[i]$ = element at position i in σ

For monotone predicates P of finite lists, we introduce a quantifier ∇ such that "$\nabla^{\tau \preceq \sigma}.\ P(\tau)$" expresses that no matter how σ evolves to a better approximation τ, eventually $P(\tau)$ will hold.

$$\frac{P(\sigma)}{\nabla^{\tau \preceq \sigma}.\ P(\tau)}\ (\sigma \in X^*) \qquad \frac{\forall^{x \in X}.\ \nabla^{\tau \preceq (\sigma ::^r x)}.\ P(\tau)}{\nabla^{\tau \preceq \sigma}.\ P(\tau)}\ (\sigma \in X^*)$$

$$\frac{\forall^{\tau \preceq \sigma}.\ a \in \tau \Rightarrow \nabla^{\upsilon \preceq \tau}.\ P(\upsilon)}{\nabla^{\tau \preceq \sigma}.\ P(\tau)}\ (\sigma \in X^*, a \in X)$$

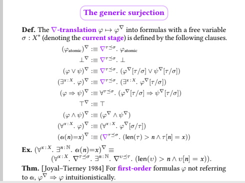
The generic surjection

Def. The ∇-**translation** $\varphi \mapsto \varphi^\nabla$ into formulas with a free variable $\sigma : X^*$ (denoting the **current stage**) is defined by the following clauses.

$$\begin{aligned}
(\varphi_{\text{atomic}})^\nabla &:\equiv \nabla^{\tau \preceq \sigma}.\ \varphi_{\text{atomic}} \\
\bot^\nabla &:\equiv \nabla^{\tau \preceq \sigma}.\ \bot \\
(\varphi \vee \psi)^\nabla &:\equiv \nabla^{\tau \preceq \sigma}.\ (\varphi^\nabla[\tau/\sigma] \vee \psi^\nabla[\tau/\sigma]) \\
(\exists^{x:X}.\ \varphi)^\nabla &:\equiv \nabla^{\tau \preceq \sigma}.\ (\exists^{x:X}.\ \varphi^\nabla[\tau/\sigma]) \\
(\varphi \Rightarrow \psi)^\nabla &:\equiv \forall^{\tau \preceq \sigma}.\ (\varphi^\nabla[\tau/\sigma] \Rightarrow \psi^\nabla[\tau/\sigma]) \\
\top^\nabla &:\equiv \top \\
(\varphi \wedge \psi)^\nabla &:\equiv (\varphi^\nabla \wedge \psi^\nabla) \\
(\forall^{x:X}.\ \varphi)^\nabla &:\equiv (\forall^{x:X}.\ \varphi^\nabla[\sigma/\tau]) \\
(\alpha(n){=}x)^\nabla &:\equiv (\nabla^{\tau \preceq \sigma}.\ (\mathrm{len}(\tau) > n \wedge \tau[n] = x))
\end{aligned}$$

Ex. $(\forall^{x:X}.\ \exists^{n:\mathbb{N}}.\ \alpha(n){=}x)^\nabla \equiv$
$$(\forall^{x:X}.\ \nabla^{\tau \preceq \sigma}.\ \exists^{n:\mathbb{N}}.\ \nabla^{\upsilon \preceq \tau}.\ (\mathrm{len}(\upsilon) > n \wedge \upsilon[n] = x)).$$

Thm. [Joyal–Tierney 1984] For **first-order** formulas φ not referring to α, $\varphi^\nabla \Rightarrow \varphi$ intuitionistically.



The generic surjection

Def. The ∇-**translation** $\varphi \mapsto \varphi^\nabla$ into formulas with a free variable $\sigma : X^*$ (denoting the **current stage**) is defined by the following clauses.

$$\begin{aligned}
(\varphi_{\text{atomic}})^\nabla &:\equiv \nabla^{\tau \preceq \sigma}.\ \varphi_{\text{atomic}} \\
\bot^\nabla &:\equiv \nabla^{\tau \preceq \sigma}.\ \bot \\
(\varphi \vee \psi)^\nabla &:\equiv \nabla^{\tau \preceq \sigma}.\ (\varphi^\nabla[\tau/\sigma] \vee \psi^\nabla[\tau/\sigma]) \\
(\exists^{x:X}.\ \varphi)^\nabla &:\equiv \nabla^{\tau \preceq \sigma}.\ (\exists^{x:X}.\ \varphi^\nabla[\tau/\sigma]) \\
(\varphi \Rightarrow \psi)^\nabla &:\equiv \forall^{\tau \preceq \sigma}.\ (\varphi^\nabla[\tau/\sigma] \Rightarrow \psi^\nabla[\tau/\sigma]) \\
\top^\nabla &:\equiv \top \\
(\varphi \wedge \psi)^\nabla &:\equiv (\varphi^\nabla \wedge \psi^\nabla) \\
(\forall^{x:X}.\ \varphi)^\nabla &:\equiv (\forall^{x:X}.\ \varphi^\nabla[\sigma/\tau]) \\
(\alpha(n){=}x)^\nabla &:\equiv (\nabla^{\tau \preceq \sigma}.\ (\mathrm{len}(\tau) > n \wedge \tau[n] = x))
\end{aligned}$$

Ex. $(\forall^{x:X}.\ \exists^{n:\mathbb{N}}.\ \alpha(n){=}x)^\nabla \equiv$
$$(\forall^{x:X}.\ \nabla^{\tau \preceq \sigma}.\ \exists^{n:\mathbb{N}}.\ \nabla^{\upsilon \preceq \tau}.\ (\mathrm{len}(\upsilon) > n \wedge \upsilon[n] = x)).$$

Thm. [Joyal–Tierney 1984] For **first-order** formulas φ not referring to α, $\varphi^\nabla \Rightarrow \varphi$ intuitionistically.