PROOF MINING

Lecture 2 - The functional interpretation of intuitionistic arithmetic

Thomas Powell
University of Bath

Nordic Logic Summer School 2022

University of Bergen
14 June 2022

These slides are available at `https://t-powell.github.io/`.

Outline

Recap

In Lecture 1 we described, informally, what a proof interpretation was:

$$\mathcal{P} \mapsto \mathcal{N}$$

Proof interpretations were

- Originally developed, in response to Hilbert's program, to establish relative consistency proofs i.e.
$$\mathrm{Con}(\mathcal{N}) \Rightarrow \mathrm{Con}(\mathcal{P});$$
- Later used as a technique for extracting computational content from proofs, as captured by Kreisel's famous quote
  > *"What more do we know if we have proved a theorem by restricted means than if we merely know that it is true?"*

In this lecture, we describe in more detail a particular proof interpretation, namely Gödel's functional interpretation:

$$\mathrm{HA}^{\omega} \mapsto \mathsf{T}$$

Where HA is the theory of intuitionistic (or 'Heyting') arithmetic in all finite types, and T is System T, introduced in Lecture 2.

Plan of the lecture

1. What does it means to extract a program from a proof?
2. A brief account of intuitionistic arithmetic.
3. A description of how the functional interpretation acts on formulas of HA.
4. Gödel's soundness theorem and sketch of the proof.
5. A worked example: Reversing a list.

This is the most technical lecture! Three things to bear in mind:

- The messy details are not important: We just want to convey the main idea.
- There will be lots of examples!
- We will see the payoff in lectures 3 and 4.

Key references

- Avigad, J. and Feferman, S. (1998). Gödel's functional ("Dialectica") interpretation.
  In Buss, S. R., editor, *Handbook of Proof Theory*, volume 137, pages 337–405. Elsevier
- Kohlenbach, U. (2008). *Applied Proof Theory - Proof Interpretations and their Use in Mathematics*.
  Springer Monographs in Mathematics. Springer

Also recommended:

- Gödel, K. (1958). Über eine bisher noch nicht benützte Erweiterung des finiten Standpunktes.
  Dialectica, 12:280–287

UNIVERSITY OF BATH

Outline

Back to prime numbers

Note. For a more detailed account of the content of this section, see [Kohlenbach, 2008, Chapter 2].

Theorem

There are infinitely many prime numbers.

Theorem (Formal version)

For any $m \in \mathbb{N}$ there exists some $p > m$ such that p is prime.

Can I produce a computable function (i.e. a program) $f : \mathbb{N} \to \mathbb{N}$ which produces a prime number of arbitrary size i.e. $f(m) > m$ and $f(m)$ prime?

Define

$$f(m) := p \text{ where } p \text{ is the least prime number greater than } m$$

i.e. a blind search for the next prime number.

This function is certainly computable, but it doesn't tell us anything more. In particular, I have no idea how long it takes to find the next prime.

But maybe there is computational content in the proof...

Euclid's elementary proof

Let's first consider the ancient Greek proof, which we mentioned in Lecture 1.

Proof 1 (Euclid).

Fix $m \in \mathbb{N}$ and consider the number

$$N := 1 + p_1 \cdots p_k$$

where $p_1, \ldots, p_k$ are all the prime numbers $\leq m$. Then N cannot be divisible by any prime number $\leq m$. But N contains at least one prime factor p, which must therefore be greater than m. □

We now have something better than blind search! We have a *bound* on how far we need to look. Define

$$f(m) := \text{least } p \leq 1 + p_1 \cdots p_k \text{ such that } p \text{ prime.}$$

We can even use this to bound the size of the mth prime number p_m. We have

$$p_m \leq 1 + p_1 \ldots p_{m-1}$$

and therefore (by induction)

$$p_m < 2^{2^m}$$

UNIVERSITY OF BATH

A numerical result

Theorem (Original)

For any m there exists some $p > m$ such that p is prime.

Theorem (Stronger)

For any m there exists some $m < p \leq 1 + p_1 \cdots p_k$ such that p is prime, where $p_1, \ldots, p_k$ are the prime numbers $\leq m$.

We didn't have to do anything *new* to product this stronger theorem: The numerical information was already 'hidden' in the proof of the original theorem.

> *"What more do we know if we have proved a theorem by restricted means than if we merely know that it is true?"*

So what happens if we look at a different proof of the same result?

UNIVERSITY OF BATH

Euler's analytic proof I

Proof 2 (Euler).

Suppose there are only finitely many prime numbers $p_1, \ldots, p_m$. We have (by simple combinatorics)

$$\sum_{0 \leq k_1, \ldots, k_m \leq n} \frac{1}{p_1^{k_1} \cdots p_m^{k_m}} = \left(\sum_{i=0}^{n} \frac{1}{p_1^i} \right) \cdots \left(\sum_{i=0}^{n} \frac{1}{p_m^i} \right)$$

But using

$$\sum_{i=0}^{n} \frac{1}{p^i} < \sum_{i=0}^{\infty} \frac{1}{p^i} = \frac{p}{p-1}$$

we have

$$\begin{aligned} \sum_{0 \leq k_1, \ldots, k_m \leq n} \frac{1}{p_1^{k_1} \cdots p_m^{k_m}} &< \frac{p_1}{p_1 - 1} \cdot \cdots \cdot \frac{p_m}{p_m - 1} \\ &\leq \frac{2}{1} \cdot \frac{3}{2} \cdot \cdots \cdot \frac{p_m}{p_m - 1} \\ &= p_m \end{aligned}$$

UNIVERSITY OF BATH

Euler's analytic proof II

Proof cont...

We have shown that

$$\sum_{0 \leq k_1, \ldots, k_m \leq n} \frac{1}{p_1^{k_1} \cdots p_m^{k_m}} < p_m$$

for any n. Now using the prime factorisation theorem, it follows that

$$\sum_{i=1}^{n} \frac{1}{i} \leq p_m$$

for all n, contradicting the fact that

$$\sum_{i=1}^{\infty} \frac{1}{i} = \infty$$

Therefore there are infinitely many primes! □

An even stronger numerical result

An analysis of the proof, using the fact that for all $m \in \mathbb{N}$ we have

$$\sum_{i=1}^{n_m} \frac{1}{i} > m$$

for $n_m := \lceil e^{m-\gamma} \rceil$, where $\gamma \approx 0.5772$ is the so-called Euler-Mascheroni constant yields the following:

Theorem (Stronger)

For any m there exists some $m < p \leq \lceil e^{m-\gamma} \rceil$ such that p is prime.

Again, the numerical information was already hidden in the proof. However, this time we had to provide computational information for the assumptions that were used: We assumed that $\sum_{i=1}^{\infty} \frac{1}{i}$ diverged, and so we needed to know how fast.

More generally, our proof gives us the following procedure:

$$\text{Rate of divergence of} \sum_{i=1}^{\infty} \frac{1}{i} \mapsto \text{Bound on the } m\text{th prime}$$

UNIVERSITY OF BATH

A general pattern

Can we apply this idea to arbitrary proofs? I.e. devise a formal map

$$\text{Proof of } A \mapsto \text{ program which gives a computational interpretation to } A$$

There are already several ambiguities here.

1. How do I treat a proof as a formal object?
2. What do I mean by a 'computational interpretation' of A? If A is of the form

   $$\forall n \in \mathbb{N} \exists m \in \mathbb{N} P(n, m)$$

   it is reasonable to ask for a function $f : \mathbb{N} \to \mathbb{N}$ satisfying

   $$\forall n P(n, f(n)).$$

   But what about formulas of arbitrary logical complexity?
3. How do I guarantee that I can extract a program from any proof?

We deal with each of these in turn.

Outline

Formal theories of arithmetic

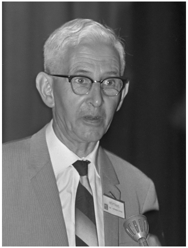
We have already mentioned Peano arithmetic PA many times. This is an axiomatic system for reasoning about the natural numbers, based on a collection of axioms postulated by Giuseppe Peano.

It contains the usual axioms of classical predicate logic, plus some special non-logical axioms like induction.

Intuitionistic, or *Heyting* arithmetic HA (named after Arend Heyting, who was important to the development of intuitionistic logic), is just Peano arithmetic, but based on intuitionistic predicate logic: Simply put, predicate logic without the law of excluded-middle

$$P \vee \neg P$$

We actually work in the more general setting of Heyting arithmetic in all finite types, which we label HA^ω.



UNIVERSITY OF BATH

Heyting arithmetic (the language)

Simply put,

$$\text{Heyting arithmetic } \mathrm{HA}^\omega = \text{System } \mathsf{T} + \text{Intuitionistic predicate logic}$$

Terms of HA^ω include all of the terms of T, i.e. are built up from

- variables x^ρ, y^ρ, z^ρ for each type ρ;
- constants $\mathsf{0}$, s, $\lambda x.t$ and R_ρ.

Formulas of HA^ω are build up as follows.

- $s = t$ is a formula, where s, t are terms of type $\mathbb{N}$.
- If A, B are formulas, so are $A \wedge B, A \vee B, A \to B$.
- If $A(x^\rho)$ is a formula, so is $\exists x^\rho A(x)$ and $\forall x^\rho A(x)$.

UNIVERSITY OF BATH

Heyting arithmetic HA^ω (the axioms and rules)

The axioms (and rules) of Heyting arithmetic are essentially

$$\text{Axioms of higher order predicate logic} + \text{System } \mathsf{T} \text{ axioms}$$
$$+\text{Equality axioms} + \text{Induction on } \textit{arbitrary} \text{ formulas}$$

Axioms and rules of predicate logic include e.g.

- structural rules: $A \to A \wedge A$;
- quantifier axioms: for example $A(t) \to \exists x A(x)$;
- modus ponens: From A and $A \to B$ infer B.

The most important non-logical rule of HA^ω is induction, which is given by

- From $A(0)$ and $\forall n(A(n) \to A(n+1))$ infer $\forall n A(n)$.

Note. For full details, see [Avigad and Feferman, 1998] or [Kohlenbach, 2008, Chapter 3].

Formal proofs in Heyting arithmetic

A formal proof in HA^ω is a derivation using the axioms and rules. We write $\mathrm{HA}^\omega \vdash A$ for 'we can prove A in HA^ω'. Formal proofs are much longer that their 'informal' textbook counterparts!

Theorem

$\mathrm{HA}^\omega \vdash \forall n(n = 0 \lor n \neq 0)$.

Formal proof (sketch!)

Let $A(n) :\equiv n = 0 \lor n \neq 0$.

1. $0 = 0$ and $0 = 0 \to 0 = 0 \lor 0 \neq 0$ therefore $A(0)$.
2. $n + 1 \neq 0$ and $n + 1 \neq 0 \to n + 1 = 0 \lor n + 1 \neq 0$ therefore $A(n + 1)$.
3. $A(n + 1) \land A(n) \to A(n + 1)$.
4. $A(n + 1) \to (A(n) \to A(n + 1))$.
5. Therefore $A(n) \to A(n + 1)$.
6. By quantifier-rules $\forall n(A(n) \to A(n + 1))$.
7. By the rule of induction $\forall n A(n)$.

□

Remark. In Peano arithmetic, $n = 0 \lor n \neq 0$ follows from excluded-middle.

Outline

UNIVERSITY OF BATH

Functional interpretation: The basic idea

The functional interpretation (also known as the Dialectica interpretation) is a translation of the following form:

$$A \mapsto \exists x^{\vec{\rho}} \forall y^{\vec{\sigma}} A_D(x, y)$$

where

- A is a formula of Heyting arithmetic HA^ω;
- $A_D(x, y)$ is a formula of System T (also a quantifier-free formula of HA^ω);
- $x^{\vec{\rho}}$ and $y^{\vec{\sigma}}$ are (potentially empty) tuples of variables.

The idea is that $\exists x \forall y A_D(x, y)$ is obtained from A by

'pulling all its quantifiers to the front'.

In particular,

$$A \leftrightarrow \exists x \forall y A_D(x, y)$$

over some suitable theory.

Functional interpretation: The simple cases

We define the functional interpretation formally using induction over the logical structure of A.

For the base case:

- If A is atomic then $A \mapsto A$ i.e. x, y are empty and $A_D := A$.

Suppose that $A \mapsto \exists x \forall y A_D(x, y)$ and $B \mapsto \exists u \forall v B_D(u, v)$. Then

- $A \wedge B \mapsto \exists x, u \forall y, v (A_D(x, y) \wedge B_D(u, v))$
- $A \vee B \mapsto \exists b^0, x, u \forall y, v ((b = 0 \to A_D(x, y)) \wedge (b \neq 0 \to B_D(u, v)))$
- $\exists z A(z) \mapsto \exists z, x \forall y A_D(x, y, z)$
- $\forall z A(z) \mapsto \exists X \forall z, y A_D(X(z), y, z)$

Note that implication is still missing... This is much more subtle and will come later.

UNIVERSITY OF BATH

Prime numbers again

Theorem

There are infinitely many primes.

Theorem (Formal)

$\forall n \exists x (x \geq n \wedge x \text{ is prime})$.

Proof.

Euclid or Euler. □

Theorem (Functional interpretation)

$\exists X \forall n (X(n) \geq n \wedge X(n) \text{ is prime})$.

Canditates for X include:

- $X(n) :=$ search up to $1 + p_1 \ldots p_k$ (corresponds to Euclid).
- $X(n) :=$ search up to $\lceil e^{n-\gamma} \rceil$ (corresponds to Euler).

UNIVERSITY OF BATH

Another simple example

Theorem

For any number n there exists some m such that $n = 2m$ *or* $n = 2m + 1$.

Theorem (Formal)

$\forall n \exists m (n = 2m \vee n = 2m + 1)$.

Proof.

Induction using case distinctions. □

Theorem (Functional interpretation)

$\exists M, B \forall n ((B(n) = 0 \rightarrow n = 2M(n)) \wedge (B(n) = 1 \rightarrow n = 2M(n) + 1))$.

Candidate realizer:

$$B(n) = \begin{cases} 0 & \text{if } n \text{ even} \\ 1 & \text{if } n \text{ odd} \end{cases}$$

$$M(n) = \lfloor n/2 \rfloor$$

Outline

Completing the definition

We have one more logical connective to deal with, namely implication $A \to B$.

Suppose that $A \mapsto \exists x \forall y A_D(x, y)$ and $B \mapsto \exists u \forall v B_D(u, v)$, and consider the implication

$$\exists x \forall y A_D(x, y) \to \exists u \forall v B_D(u, v).$$

We want to bring the quantifiers to the front in the least non-constructive way possible.

Note. For a detailed discussion of this, and the various possibilities, see [Kohlenbach, 2008, Chapter 8].

We describe the functional interpretation of implication using the language of game semantics. The idea here is to visualise the quantifiers as representing a game between two players:

- Eloise(existential quantifier) wants to find evidence that the statement is true;
- Abelard(universal quantifier) tries to confound *Eloise* by claiming that the statement is false.

The functional interpretation of implication I

$$\exists x \forall y A_D(x, y) \to \exists u \forall v B_D(u, v)$$

Let's imagine this as a game between Eloise and Abelard, who are trying to respectively prove and disprove the implication.

- Abelard: I claim that there is a realizer x for the premise, and challenge you to find a realizer for the conclusion.
- Eloise: I accept the challenge, and give you a witness u for the conclusion.
- Abelard: I claim that there is a counterexample v to your witness u.
- Eloise: In which case, I give you a counterexample y to your original witness x.

The formula is true if Eloisehas a winning strategy against any choices from Abelard.

UNIVERSITY OF BATH

The functional interpretation of implication II

For any witness challenge x from Abelard

$$\forall x(\forall y A_D(x,y) \to \exists u \forall v B_D(u,v))$$

there is a witness response u from Eloise

$$\forall x \exists u(\forall y A_D(x,y) \to \forall v B_D(u,v))$$

such that for any counterexample challenge v from Abelard

$$\forall x \exists u \forall v(\forall y A_D(x,y) \to B_D(u,v))$$

there is a counterexample response y from Eloise

$$\forall x \exists u \forall v \exists y(A_D(x,y) \to B_D(u,v))$$

Now we convert these to functions:

$$\exists U, Y \forall x, v \underbrace{(A_D(x, Y(x,v)) \to B_D(U(x), v))}_{(A \to B)_D(U,Y,x,v)}$$

Euler's proof revisited I

Euler's proof of the infinitude of primes used the assumption that $\sum_{i=1}^{\infty} \frac{1}{i}$ diverges i.e.

$$\sum_{i=1}^{\infty} \frac{1}{i} = \infty \to \forall n \exists x (x > n \wedge x \text{ is prime}).$$

Written out with quantifiers this becomes

$$\forall m \exists k \left(\sum_{i=1}^{k} \frac{1}{i} > m \right) \to \forall n \exists x (x > n \wedge x \text{ is prime}).$$

Applying the functional interpretation to premise and conclusion:

$$\exists g \forall m \left(\sum_{i=1}^{g(m)} \frac{1}{i} > m \right) \to \exists X \forall n (X(n) > n \wedge X(n) \text{ is prime}).$$

Now interpreting implication:

$$\exists X, M \forall g, n \left(\sum_{i=1}^{g(M(g,n))} \frac{1}{i} > M(g,n) \to X(g,n) > n \wedge X(g,n) \text{ is prime} \right)$$

UNIVERSITY OF BATH

Euler's proof revisited II

A quantitative analysis of Euler's proof actually produces functionals X and M satisfying

$$\exists X, M \forall g, n \left(\sum_{i=1}^{g(M(g,n))} > M(g,n) \to X(g,n) < n \wedge X(g,n) \text{ is prime} \right)$$

In particular, we have a map

$$\text{rate of divergence } g \mapsto \text{function } X(g) \text{ for finding the next prime.}$$

In our case we took a known rate of divergence, namely

$$\sum_{i=1}^{\lceil e^{m-\gamma} \rceil} \frac{1}{i} > m$$

where γ is the Euler-Mascheroni constant, and used it to produce an upper bound on the next prime.

Outline

UNIVERSITY OF BATH

Gödel's main theorem

So far we have a translation

$$A \mapsto \exists x \forall y A_D(x, y).$$

which maps formulas A of HA^ω to formulas $A_D(x, y)$ of System T.

Gödel's soundness theorem says that we can translate a proof of A to a program witnessing $\exists x \forall y A_D(x, y)$.

Theorem (K. Gödel, 1958)

Suppose that

$$\mathrm{HA}^\omega \vdash A$$

Then there exists a term t of System T such that

$$\mathrm{HA}^\omega \vdash \forall y A_D(t, y)$$

and moreover, we can formally extract t from the proof of A.

Proof.

Induction over formal proofs of HA^ω. □

A quick aside: Relative consistency

Actually, Gödel established the conclusion of the soundness theorem within System T itself i.e. he showed that if $\mathrm{HA}^\omega \vdash A$ then

$$\text{System } \mathsf{T} \vdash A_D(t, y).$$

Therefore, if HA^ω is inconsistent i.e. $\mathrm{HA}^\omega \vdash 0 = 1$, then System T is inconsistent i.e. System $\mathsf{T} \vdash 0 = 1$.

This follows from the soundness of the functional intepretation, plus the fact that $0 = 1$ gets mapped to itself.

Another way of saying this is

$$\mathrm{Con}(\mathsf{T}) \Rightarrow \mathrm{Con}(\mathrm{HA}^\omega).$$

This is the last time we mention relative consistency proofs and Hilbert's program! From now on, our interest lies primarily in Kreisel's shift of emphasis towards the extraction of programs from proofs.

In particular, it is more practical to reason about terms of System T in HA^ω, since we have access to quantifiers.

UNIVERSITY OF BATH

The proof: Modus ponens

Modus ponens. If A and $A \to B$ then we can infer B:

Soundness of modus ponens. If we have a witness for the f.i. of A and $A \to B$ then we can produce a witness for the f.i. of B.

Suppose that $A \mapsto \exists x \forall y A_D(x, y)$ and $B \mapsto \exists u \forall v B_D(u, v)$. We are given

- A term r such that $\forall y A_D(r, y)$;
- Terms s_1 and s_2 such that $\forall x, v(A_D(x, s_2 xv) \to B_D(s_1 x, v))$.

We want to produce:

- A term t such that $\forall v B_D(t, v)$.

For any v, instantiating $y := s_2 rv$ and $x := r$ yields

$$A_D(r, s_2 rv) \quad \text{and} \quad A_D(r, s_2 rv) \to B_D(s_1 r, v)$$

from which we infer

$$B_D(s_1 r, v).$$

So $t := s_1 r$ works.

The proof: Induction

Induction. From $A(0)$ and $\forall n(A(n) \to A(n+1))$ we can infer $\forall n A(n)$.

Soundness of induction. If we have a witness for the f.i. of $A(0)$ and $\forall n(A(n) \to A(n+1))$ then we can produce a witness for the f.i. of $\forall n A(n)$.

Suppose that $A(n) \mapsto \exists x \forall y A_D(n, x, y)$. We are given

- A term r such that $\forall y A_D(0, r, y)$;
- Terms s_1 and s_2 such that $\forall n, x, y(A_D(n, x, s_2 xy) \to A_D(n+1, s_1 x, y))$.

We want to produce:

- A term t such that $\forall n, y A_D(n, tn, y)$.

Using the recursors, define t by

$$t0 := r \quad \text{and} \quad t(n+1) := s_1(tn).$$

We prove by another induction that this term works. First, note that $\forall y A_D(0, t0, y)$. Now if $\forall y A(n, tn, y)$, then in particular $A_D(n, tn, s_2(tn)y)$ and therefore $A_D(n+1, s_1(tn), y)$, which is just $A_D(n+1, t(n+1), y)$.

UNIVERSITY OF BATH

For the enthusiasts: Contraction

The rest of the soundness proof is fairly straightforward, with the exception of the seemingly innocuous axiom of contraction i.e.

$$A \to A \wedge A.$$

For this we need a pair of function t_1, t_2 and s satisfying

$$\forall x, y, y' (A_D(x, sxyy') \to A_D(t_1 x, y) \wedge A_D(t_2 x, y')).$$

Define $t_1 x = t_2 x := x$ and

$$sxyy' := \begin{cases} y & \text{if } A_D(x, y') \\ y' & \text{if } \neg A_D(x, y') \end{cases}$$

This works, but there are two issues:

1. We need the quantifier-free $A_D(x, y')$ to be decidable.
   Consequence. Extending the functional interpretation to theories where quantifier-free formulas are not decidable (e.g. set theory) is difficult.
2. The interpretation is asymmetric.
   Consequence. Building nice categorical models of the functional interpretation is difficult.

University of BATH

Outline

A proof that all lists can be reversed

Theorem

For all lists of natural numbers $a \in \mathbb{N}^*$ there exists a list $b \in \mathbb{N}^*$ which is the reversal of a.

Theorem (Formal)

$\forall a \exists b \ \mathrm{Rev}(a, b)$.

Note. *We can encode lists as single natural numbers and reason about them in* HA^ω.

Proof.

We use induction on the length of a. Define

$$A(n, a, b) :\equiv (\mathrm{len}(a) = n \rightarrow \mathrm{Rev}(a, b)).$$

First, note that $A(0, a, [])$ and so $\forall a \exists b A(0, a, b)$.

Now fix n and suppose that $\forall a \exists b A(n, a, b)$. Take some a' with $\mathrm{len}(a') = n + 1$. Then $a' = x :: a$ for some a with $\mathrm{len}(a) = n$. By hypothesis there is some b with $\mathrm{Rev}(a, b)$ and hence $\mathrm{Rev}(x :: a, b :: x)$. We have shown

$$\forall a \exists b A(n, a, b) \rightarrow \forall a' \exists b' A(n + 1, a', b')$$

and hence by induction $\forall n, a \exists b A(n, a, b)$, from which the theorem follows. □

An extracted list reversal program

How does the functional interpretation treat this proof? Note that

$$\forall a \exists b A(n, a, b) \mapsto \exists f \forall a A(n, a, fa).$$

where $A(n, a, b) :\equiv (\mathrm{len}(a) = n \to \mathrm{Rev}(a, b))$. We have:

- $\forall a A(0, a, \underbrace{[\,]}_{ra})$
- $\forall n, f, a(A_D(n, \underbrace{\mathrm{tail}(a)}_{s_2 fa}, f(\mathrm{tail}(a))) \to A_D(n+1, a, \underbrace{f(\mathrm{tail}(a)) :: \mathrm{head}(a)}_{s_1 fa}))$

Therefore defining

$$t(0, a) = [\,] \quad \text{and} \quad t(n+1, a) = t(n, \mathrm{tail}(a)) :: \mathrm{head}(a)$$

we have $\forall n A(n, a, t(n, a))$. In other words, defining

$$t'a := t(\mathrm{len}(a), a)$$

we have $\forall a \mathrm{Rev}(a, t'a)$.

UNIVERSITY OF BATH

The extraction process

How do we extract programs from proofs in practice?

Option 1: 'by hand'

- Take a textbook proof and try to understand its general logical structure.
- Use the proof interpretation as a tool to guide you in extracting a program.
- Write down the program using pen and paper.

Option 1: 'by machine'

- Take a textbook proof and formalise it rigorously in a proof assistant.
- Press a button and automatically extract a program.
- Display the program in some implementation of System T.

Both approaches have their advantages and drawbacks, and depending on the context.

UNIVERSITY OF BATH

A look ahead to Lecture 3

Everything in thie lecture applied to intuitionistic arithmetic. All of our proofs were fundamentally constructive in nature - the functional interpretation just allows us to systematically extract the program implicit in the proof.

Classical arithmetic, on the other hand, is able to prove existential theorems whose functional interpretation cannot be realized by any computable functional.

But does this mean that classical proofs don't contain *any* computational information?

Outline

UNIVERSITY OF BATH

References I

Avigad, J. and Feferman, S. (1998).
Gödel's functional ("Dialectica") interpretation.
In Buss, S. R., editor, *Handbook of Proof Theory*, volume 137, pages 337–405. Elsevier.

Gödel, K. (1958).
Über eine bisher noch nicht benützte Erweiterung des finiten Standpunktes.
Dialectica, 12:280–287.

Kohlenbach, U. (2008).
Applied Proof Theory - Proof Interpretations and their Use in Mathematics.
Springer Monographs in Mathematics. Springer.

UNIVERSITY OF BATH