

Calculer avec l'axiome du choix et ses variantes

(version de mai 2025 corrigeant une généralisation abusive de la commutation de $\neg\neg$ avec les flux
à la commutation de toute monade forte avec les flux, pages 38-40)

Hugo Herbelin

Journées Francophones des Langages Applicatifs

Roiffé

28 janvier 2025

Outline

I Retrospective: proofs compute

1. Generalities
2. Panorama
3. Computing with classical logic
4. A few words on other axioms: extensionality, Church's thesis

II Computing with the axiom of choice

1. A diversity of formulations
2. Axiom of unique choice
3. Digression on weak König's lemma
4. **DNS** and distributivity of dependent products over a monad

III Computing with principles equivalent to the axiom of choice

1. Panorama of principles equivalent to the axiom of choice
2. Computing with bar induction: bar recursion
3. Computing with update induction: update recursion
4. Other examples

Part I

Retrospective: proofs compute

Brouwer, from 1905: Proofs should be processes

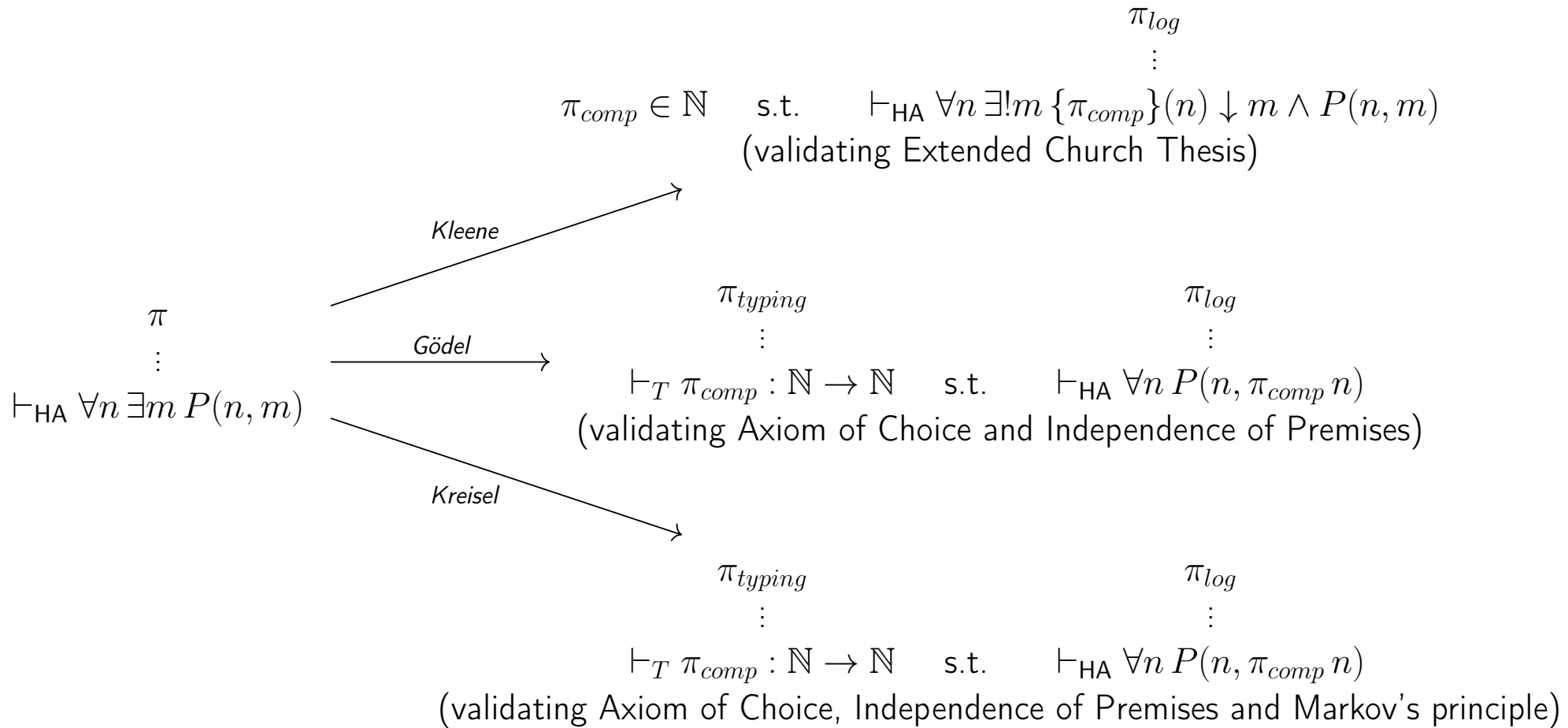
A proof of $\exists n P(n)$ should provide an n_0 such that $P(n_0)$

A proof of $A \vee B$ should be either a proof of A or a proof of B

A proof of $A \rightarrow B$ should build a proof of B from a proof of A

A manifesto for intuitionistic logic: discarding $A \vee \neg A$

Realisability: extracting programs from intuitionistic proofs in HA (Kleene 1945, Gödel 1958, Kreisel 1962)



The transformations split π into a pure computation π_{comp} and a purely logical (proof-irrelevant) specification π_{log} .

Curry 1934, Howard 1968: Intuitionistic proofs *are* typed programs

π $\vdots$ $\vdash_{\text{HA}} \forall n \exists m P(n, m)$	$\pi_{\text{derivation}}$ $\vdots$ $\vdash_{\text{HA}} \pi_{\text{rawterm}} : \forall n \exists m P(n, m)$
proof normalisation	β -reduction on terms + subject reduction on derivations
intrinsic syntax	extrinsic syntax

Originally supposed to hold only between intuitionistic logic and functional programming.

Exploited by Martin-Löf to build his type theories.

However, retrospectively, for type theories, there is still an intrinsic/extrinsic opposition:

categories with families quotient-inductive-inductive types (only derivations)	type theory rawterms + type-checking/conversion rules
--	--

Computing with axioms a panorama

Different kinds of axioms

We shall consider:

- Classical logic
- Extensionality axioms
- “Arithmetisation” axioms (e.g. Church’s thesis)
- Axiom of choice and variants

Classical logic does compute

Gentzen's **cut-elimination** for sequent calculus, 1932: morally a computation, supporting classical logic, as emphasized in e.g., $\bar{\lambda}\mu\tilde{\mu}$ -**calculus and system L** .

Prawitz's **normalisation** for natural deduction, 1965, supporting classical logic.

Double negation translations (Kolmogorov 1925, Gödel-Gentzen 1933, Kuroda, Krivine): **indirectly** provides computational contents to classical logic.

Griffin 1990: Scheme's call-with-current-continuation typable of type:

$$\text{Peirce's law: } ((A \rightarrow \perp) \rightarrow A) \rightarrow A$$

Felleisen's $\mathcal{C}$ and Danvy-Filinski's $\mathcal{S}$ operators typable of type $((A \rightarrow R) \rightarrow R) \rightarrow A$, for R type of toplevel computation; $\mathcal{C}$ happens to compute similarly to Prawitz; e.g.,

$$\text{Felleisen: } \pi_1(\mathcal{C}(\lambda k^{\neg(A \wedge B)}.M)) \rightarrow \lambda k_1^{\neg A}.M[k := \lambda x.k_1(\pi_1 x)]$$

$$\text{Prawitz: } \mathcal{C}(\lambda k^{\neg(A \wedge B)}.M) \rightarrow (\lambda k_1^{\neg A}.M[k := \lambda x.k_1(\pi_1 x)], \lambda k_2^{\neg B}.M[k := \lambda x.k_2(\pi_2 x)])$$

After Griffin: the conviction that all axioms should compute

After Griffin's discovery, it started to become clear that all axioms would compute. In Paris, Krivine started a program to realise those not already realised: classical realisability.

Example: Krivine showed that all axioms of ZF are realisable.

Example: Miquel 2011's abstract machine for forcing uses a memory.

Example: by doing so, programs are given for the ultrafilter theorem or for well-ordering $\mathbb{R}$.

In any case, by abstract reasoning: any consistent axiom computes

Indeed, assume a statement A consistent; this means that there is a model, that is a consistency-preserving syntactic translation $*$ that maps

$$A \vdash M : \perp$$

to

$$\vdash M^* : \perp^*$$

that is, by preservation of consistency, to

$$\vdash M^* : \perp$$

If the target logic has a computational interpretation (e.g., **ZF**, type theory, ...), then, this interpretation can be **pulled back** through $*$ to give a computational content to A .

If the translation is monadic or comonadic, this is a case of Führmann and Munch-Maccagnoni direct style

Führmann 1999: If an axiom is interpretable through some monad T then there is a “think-force” calculus that reflects in **direct style** the behaviour of the axiom through the monad.

Example, the double negation monad $K(A) \triangleq \neg\neg A$ generates a translation:

$$\begin{aligned} X^K &\triangleq X \\ (A \rightarrow B)^K &\triangleq A^K \rightarrow K(B^K) \\ (A \vee B)^K &\triangleq A^K \vee B^K \\ &\dots \end{aligned}$$

such that $\Gamma, \forall A (\neg\neg A \rightarrow A) \vdash_I B$ implies $\Gamma^K \vdash_I K(B^K)$

This in turn defines a “classical” calculus where $\text{DNE}_A : \neg\neg A \rightarrow A$ computes (see, e.g., Hofmann 1994), with rules such as:

$$E[\text{DNE}_A(M)] \rightarrow \text{DNE}_A(\lambda k. (M \lambda x. E[k x]))$$

and where other constructions and rules arrive, e.g., such calculus can be thought as having a **reset** of type $\perp$.

Focus on some intuitionistically compatible classical axioms

Weak classical principles that preserve the disjunction and existence properties

A few standard classical principles are morally intuitionistic in the sense that they preserve the **disjunction** and **existence properties**, that is, for closed proofs:

$$\begin{array}{lll} \vdash A \vee B & \text{implies} & \vdash A \text{ or } \vdash B \\ \vdash \exists x^A P(x) & \text{implies} & \vdash P(t) \text{ for some } t \end{array}$$

This includes:

- Markov's principle
- independence of premises
- double negation shift

We shall survey them.

Markov's principle

One gets Markov's principle by using Coquand-Hofmann translation:

$$\mathbf{MP} : \neg\neg\exists n f(n) = 0 \rightarrow \exists n f(n) = 0$$

(i.e. $\mathbf{DNE}_A$ for $A \in \Sigma_1^0$)

Coquand-Hofmann translation combines a forcing translation (so as to interpret local uses of $\mathbf{MP}$ in the proof) with either an exception monad translation or a double-negation monad over Σ_1^0 formulas (Berger, Pédrot 2020, H.):

$$CH_{B \in \Sigma_1^0}^{exc}(A) \triangleq \forall B' \in \Sigma_1^0, (B \rightarrow B') \rightarrow A \vee B'$$

(then interpreting $\mathbf{MP}$ with exceptions, i.e. with dynamically bound continuations)

$$CH_{B \in \Sigma_1^0}^{callcc}(A) \triangleq \forall B' \in \Sigma_1^0, (B \rightarrow B') \rightarrow (A \rightarrow B') \rightarrow B'$$

(then interpreting $\mathbf{MP}$ with `callcc`, i.e. with statically bound continuations)

This can be generalised in various ways (e.g. hereditarily positive instead of Σ_1^0)

Independence of Premises

Kreisel modified realisability interprets the independence of premises:

$$\text{IP} : (\neg P \rightarrow \exists x^A Q(x)) \rightarrow \exists x^A (\neg P \rightarrow Q(x))$$

by interpreting all propositions, even $\perp$, as inhabited types.

As analysed by Pédrot and Tabareau 2018, this can be rephrased in Curry-Howard style by using **exceptions**, and filtering out the non-exceptional inhabitants afterwards.

Propositional Double Negative Shift

Also called Kuroda principle, propositional Double Negation Shift is the following principle:

$$\text{DNS}_A^{\text{Prop}} : (\forall x^A \neg\neg P(x)) \rightarrow \neg\neg\forall x^A P(x)$$

As studied by Ilik 2011, this can be interpreted in intuitionistic logic by using a **reset** delimiting a local use of classical logic within intuitionistic logic.

We will see later another interpretation when A is $\mathbb{N}$.

Summary: some references about *Curry-Howard* style computing with classical logic

Intuitionistic	“0-to- $\perp$ ”	ex falso quodlibet	Prawitz-Malmnäs
Classical	Intuitionistic-compatible	Markov's rule MP IP $\text{DNS}_A^{\text{Prop}}$ $\text{DNS}_{\mathbb{N}}$ $\text{DNS}_{\mathbb{N}}$	Dragalin, Friedman, Leivant Coquand-Hofmann Pédrot-Tabareau Ilik rephrasing Berger-Oliva (using e.f.q.) rephrasing Escardó-Oliva (coinductive)
	Essentially-classical	DNE EM	Kolmogorov, ...

What about computing with extensionality axioms?

Computing with **extensionality** axioms is mostly about how **equality** and **rewriting** are defined. Here is an excerpt of works describing translations interpreting extensionality axioms:

Function Extensionality	Gandy 1956 / Takeuti 1953 in HOL Altenkirch in type theory
Propositional Extensionality	Gandy 1956 / Takeuti 1953 in HOL Altenkirch 1999 in type theory
Univalence	Cohen-Coquand-Huber-Mörtberg 2014

The case of Church's thesis

Church's thesis reflects within the logic the meta-language countable view at it:

$$\forall f^{\mathbb{N} \rightarrow \mathbb{N}} \exists e^{\mathbb{N}} \forall n^{\mathbb{N}} (\{e\}(n) \downarrow f n)$$

This can be proved using a “quote” operator (see, e.g., Pédrot 2024).

Part II

Computing with the axiom of choice

On the meaning of the existential quantifier

Different possible interpretations of $\exists x^A P(x)$:

- $\Sigma x^A P(x)$, as in type theory
- $\exists x^A P(x)$, as, informally, in first-order logic
- $\neg\neg\exists x^A P(x)$, reasoning classically within intuitionistic logic
- $\oplus_{x^A} P(x)$, as, e.g., in linear logic
- $\cup_{x^A} P(x)$, as, e.g., in Curry-style realisability

We can reasonably set:

$$\exists x^A P(x) \triangleq ||\Sigma x^A P(x)||$$

relying on **Prop**-truncation, also known as (-1) -truncation, that is, in Coq syntax:

```
Inductive inhabited (A : Type) : Prop := inhabits (a : A) : inhabited A.
```

```
Notation "|| A ||" := (inhabited A).
```

```
Notation "| a |" := (inhabits _ a).
```

```
Notation "let | a | := t in u" := (match t with inhabits _ a => u end).
```

Remark: In the presence of classical logic, $||\Sigma x^T P(x)||$ is equivalent to $\neg\neg\Sigma x^T P(x)$ (see HoTT Book).

How the definition of logical concepts impacts the constructivity of choice

One standard formulation (for $R : A \times B \rightarrow \mathbf{Prop}$):

$$\mathbf{AC}_{ABR} : \forall x^A \exists y^B R(x, y) \rightarrow \exists f^{A \rightarrow B} \forall x^A R(x, f(x))$$

Several possible interpretations...

$$\begin{aligned} \mathbf{AC}_{ABR}^{int} & : \forall x^A \Sigma y^B R(x, y) \rightarrow \Sigma f^{A \rightarrow B} \forall x^A R(x, f(x)) \\ \mathbf{AC}_{ABR}^{neg} & : \forall x^A \neg \neg \Sigma y^B R(x, y) \rightarrow \neg \neg \Sigma f^{A \rightarrow B} \forall x^A R(x, f(x)) \\ \mathbf{AC}_{ABR}^{inh} & : \forall x^A || \Sigma y^B R(x, y) || \rightarrow || \Sigma f^{A \rightarrow B} \forall x^A R(x, f(x)) || \\ \mathbf{AC}_{ABR}^{inh, rel} & : \forall x^A || \Sigma y^B R(x, y) || \rightarrow || \Sigma S^{A \times B \rightarrow \mathbf{Prop}} (S \subset R \wedge \forall x^A \exists ! y^B S(x, y)) || \\ \mathbf{AC}_{ABR}^{lin} & : \&_{x^A} \oplus_{y^B} R(x, y) \rightarrow \oplus_{f^{A \rightarrow B}} \&_{x^A} R(x, f(x)) \\ \mathbf{AC}_{ABR}^{\cup} & : \cap_x (x \in A \rightarrow \cup_y (y \in B \otimes R(x, y))) \rightarrow \\ & \quad \cup_f (\cap_x (x \in A \rightarrow f(x) \in B)) \otimes \cap_x (x \in A \rightarrow R(x, f(x))) \end{aligned}$$

...

... resulting in different strengths

Remark: additionally, in the presence of quotients in the ambient logic, $\mathbf{AC}^{inh}$ implies back **EM** (Diaconescu 1975), as well as weak functional extensionality (Carlström 2004).

Martin-Löf's (type-theoretic) constructive (intensional) choice

$$\mathbf{AC}_{AB}^{int} : \forall x^A \Sigma y^B P(x, y) \rightarrow \Sigma f^{A \rightarrow B} \forall x^A P(x, f(x))$$

This formulation has a simple computational contents, by distributivity of $\forall$ over Σ :

$$\mathbf{AC}_{AB}^{int} \triangleq \lambda H. (\lambda x^A. \pi_1 (H x), \lambda x^A. \pi_2 (H x))$$

This implies constructive type-theoretic variants of choice, such as dependent choice:

$$\begin{aligned} \mathbf{DC}_A^{int} : \forall x^A \Sigma y^A P(x, y) &\rightarrow \forall x_0 \Sigma f^{\mathbb{N} \rightarrow A} [f(0) = x_0 \wedge \forall n P(f(n), f(n+1))] \\ &\triangleq \lambda H. \lambda x_0. \left(\lambda n. \text{rec } n \text{ of } \left[\begin{array}{l} 0 \rightarrow x_0 \\ _ + 1 \rightarrow_x \pi_1(H x) \end{array} \right], \right. \\ &\quad \left. (\text{refl}, \lambda n. \text{rec } n \text{ of } \left[\begin{array}{l} 0 \rightarrow \pi_2(H x_0) \\ m + 1 \rightarrow \pi_2(H (\text{rec } n \text{ of } \left[\begin{array}{l} 0 \rightarrow x_0 \\ _ + 1 \rightarrow_x \pi_1(H x) \end{array} \right])) \end{array} \right]) \right] \right) \end{aligned}$$

Convention for the rest of the talk

To reason uniformly, we now assume:

$$\exists x^A P(x) \triangleq M(\Sigma x^A P(x))$$

for $M(A)$ being either A (the Martin-Löf case), $\|A\|$ or $\neg\neg A$ (in all cases a monad).

We write AC_{ABR}^M and $\text{AC}_{ABR}^{M,rel}$ for the corresponding versions of choice.

Digression about the axiom of unique choice

Proper functions vs (total) functional relations, and the role of unique choice

Axiom of unique choice (for $\exists \triangleq M\Sigma$):

$$\text{AC!}_{ABR}^M : \forall x^A \exists! y^B. R(x, y) \rightarrow \exists f^{A \rightarrow B}. \forall x^A. R(x, f(x))$$

(aka known as axiom of no-choice, or functional relation reification)

Unique choice is rather strong: it implies that the logic represents as functions all the functional relations that it can prove total, meaning that the computational level reflects the logical strength.

We have

$$\text{AC}_{ABR}^M = \text{AC}_{ABR}^{M,rel} + \text{AC!}_{ABR}^M$$

Unique choice and classical logic

$$\text{AC!}_{ABR}^M : \forall x^A \exists! y^B R(x, y) \rightarrow \exists f^{A \rightarrow B} \forall x^A R(x, f(x))$$

In the presence of classical logic, that is if $M(A) \triangleq \neg\neg A$, then $\exists$ has the meaning of $\neg\neg\Sigma$ and **AC!** becomes even stronger: *it allows to refer to the characteristic function of any predicate.*

Indeed, by classical reasoning, we have, for any predicate $P(x)$ over A :

$$\forall x^A \exists! b^{\mathbb{B}\text{ool}} (b = \mathbf{true} \leftrightarrow P(x))$$

And, by **AC!**, we get the (classical) existence of $f : A \rightarrow \mathbb{B}\text{ool}$ such that:

$$\forall x^A (f(x) = \mathbf{true} \leftrightarrow P(x))$$

Thus, classical logic with **AC!** makes **Prop** impredicative.

Convention for the rest of the talk

In set theory, $AC^{M,rel}$ does not increase the logical strength.

The non-choosing $AC!$ part of AC is the part which **increases** the logical strength of AC in a classical setting (if not already in an impredicative setting).

We will nevertheless reason with AC_{ABR}^{tt} whose formulation is simpler.

Another digression, about weak König's lemma

Impact of the difference between predicates and characteristic functions: on the intuitionistic provability of weak König's lemma

In the absence of classical logic, it is easier to inhabit $A \rightarrow \mathbf{Prop}$ than $A \rightarrow \mathbf{Bool}$. This has an impact on the intuitionistic provability of some weak choice principles, such as weak König's lemma.

Let $T : \mathbf{Bool}^* \rightarrow \mathbf{Prop}$ be a binary tree represented as a subset of finite sequence of Boolean values. We set:

$$\mathbf{WKL} : (\forall n^{\mathbb{N}} \exists u^{\mathbf{Bool}^*} T(u)) \rightarrow \exists f \forall n T(f|_n)$$

The intuitionistic provability of $\mathbf{WKL}$ depends on how the infinite branch f is characterised:

- if $f : \mathbb{N} \rightarrow \mathbf{Bool}$, it needs a bit of classical reasoning (**LLPO**) and thus a bit of *call/cc*
- if $f : \mathbb{N} \rightarrow \mathbf{Prop}$, it becomes intuitionistically provable (see Coq proof in `Logic/WKL.v`)

Let's come back to the axiom of choice

Choice axioms as existence of ideal objects

- The axiom of choice and their variants assert the existence of **ideal** objects from intensional properties of these objects
- Some advocates that such axioms can be avoided: see e.g., Coquand's program of reformulating standard mathematical statements using equivalent **inductive** properties to avoid the axiom of choice
- Computationally, the existence of **ideal** objects reduces to being able to produce arbitrarily precise approximations of the object

Reducing the axiom of choice to (relevant) double negation shift

We have

$$\text{AC}_{ABR}^M = \text{AC}_{ABR}^{int} + \text{DNS}_A^{M, \text{Type}}$$

where, for $B : A \rightarrow \text{Type}$:

$$\text{DNS}_A^{M, \text{Type}} \triangleq (\forall x^A M(B(x))) \rightarrow M(\forall x^A B(x))$$

(the actual formulation of AC in the HoTT Book)

generalises the following subclassical principle, for $B : A \rightarrow \text{Prop}$:

$$\text{DNS}_A^{\neg\neg, \text{Prop}} \triangleq (\forall x^A \neg\neg B(x)) \rightarrow \neg\neg \forall x^A B(x)$$

$\text{DNS}_{\mathbb{N}}^{\neg\neg, \text{Type}}$ has a simple computational content

Computational content of countable relevant Double Negation Shift

Called *modified bar recursion* in the context of realisability (Berger and Oliva, 2001, up to details), the following is a proof of relevant Double Negation Shift in the countable case.

Assume $H : \forall n^{\mathbb{N}} \neg \neg P(n)$ and $K : \neg \forall n^{\mathbb{N}} P(n)$. We define by recursion:

$$\text{DNS}_{H,K}(l) \triangleq K(\lambda n. \text{if } n < |l| \text{ then } l(n) \text{ else } \text{efq } (H(|l| + 1)(\lambda x^{P(n)}. \text{DNS}_{H,K}(l \star x)))$$

where $\text{efq} : \perp \rightarrow C$ for any C .

The following is then a proof of $\text{DNS}_{\mathbb{N}}^{\neg\neg, \text{Type}}$:

$$\begin{aligned} \text{DNS}_{\mathbb{N}}^{\neg\neg, \text{Type}} &: \forall n^{\mathbb{N}} \neg \neg P(n) \rightarrow \neg \neg \forall n^{\mathbb{N}} P(n) \\ &\triangleq \lambda H. \lambda K. \text{DNS}_{H,K}(\langle \rangle) \end{aligned}$$

which yields a computational content to $\text{AC}_{\mathbb{N}BR}^{\neg\neg}$.

A smarter direct computational content of Double Negation Shift

Escardó and Oliva (2009) define an even smarter direct computational content of DNS (told to be suggested by Berger). The trick is to interpret $\forall n^{\mathbb{N}} \neg\neg P(n)$ as a **stream**:

$$\neg\neg P(0) \wedge (\neg\neg P(1) \wedge (\neg\neg P(2) \wedge \dots))$$

and to reason **corecursively** (under $\neg\neg$) to build a proof of:

$$\neg\neg(P(0) \wedge (P(1) \wedge (P(2) \wedge \dots)))$$

First, let us consider an **efq**-based symmetrical proof of

$$_ \otimes _ : \neg\neg P \wedge \neg\neg Q \rightarrow \neg\neg(P \wedge Q)$$

We also write:

$$\begin{aligned} \phi & : P(m) \wedge (\forall n \geq m+1 P(n)) \rightarrow (\forall n \geq m P(n)) \\ \neg\neg\phi & : \neg\neg(P(m) \wedge (\forall n \geq m+1 P(n))) \rightarrow \neg\neg(\forall n \geq m P(n)) \end{aligned}$$

Then, assuming $H : \forall n^{\mathbb{N}} \neg\neg P(n)$, we define a proof of $\neg\neg \forall n \geq m P(n)$ by:

$$\text{DNS}_H(m) \triangleq \neg\neg\phi(H(m) \otimes \text{DNS}_H(m+1))$$

The following is then a proof of $\text{DNS}_{\mathbb{N}}^{\neg\neg, \text{Type}}$:

$$\text{DNS}_{\mathbb{N}}^{\neg\neg, \text{Type}, \text{cofix}} \triangleq \lambda H. \text{DNS}_H(0)$$

Generalisation to other monads

Escardó and Oliva described their algorithm for the *Peirce monad*, also known as **selection monad**:

$$\text{Peirce}(P) \triangleq \neg P \rightarrow P$$

for which an appropriate e.f.q.-free symmetrical proof of

$$_ \otimes _ : (\neg P \rightarrow P) \wedge (\neg Q \rightarrow Q) \rightarrow \neg(P \wedge Q) \rightarrow P \wedge Q$$

is available, which is able to produce a pair of proofs without needing to first evaluate both sides of the conjunction:

$$p \otimes q \triangleq \lambda k. (p \lambda a. k(a, q \lambda b. k(a, b)), q \lambda b. k(p \lambda a. k(a, b), b))$$

For such appropriate monad M , the proof of $\text{DNS}_{\mathbb{N}}^{\neg\neg}$ generalises into:

$$\begin{aligned} \text{DNS}_H(m) & : M(\forall n \geq m P(n)) \\ & \triangleq (M\phi)(H(m) \otimes_M \text{DNS}_{M,H}(m+1)) \end{aligned}$$

$$\begin{aligned} \text{DNS}_{\mathbb{N}}^{M, \text{Type}, \text{cofix}} & : \forall n^{\mathbb{N}} M(P(n)) \rightarrow M(\forall n^{\mathbb{N}} P(n)) \\ & \triangleq \lambda H. \text{DNS}_H(0) \end{aligned}$$

Other interesting monads

Another interesting monad is the **exception monad**, $Exc_Q(P) \triangleq P \vee Q$ in which case we would get the **constant domain axiom**, CD, also known as DIS:

$$\text{CD} : \forall x^A (P(x) \vee Q) \rightarrow (\forall x^A P(x)) \vee Q$$

The truncation is another interesting monad, what would give a proof of:

$$\text{AC}_{\mathbb{N}BR}^{inh} : \forall n^N ||P(n)|| \rightarrow ||\forall n^N P(n)||$$

In these two cases, it is however unclear how to get a proof of

$$_ \otimes _ : M(P) \wedge M(Q) \rightarrow M(P \wedge Q)$$

able to produce a pair of proofs without needing to first evaluate both sides of the conjunction, as it is crucially the case for $\neg\neg$ and *Peirce*.

Variant for arbitrary strong monads

An alternative to the absence of an appropriate $_ \otimes _$ able to produce a result without having to first explore the two sides is, using the strength of the monad, to build from

$$M(P(0)) \wedge (M(P(1)) \wedge (M(P(2)) \wedge \dots))$$

only a proof of

$$M(P(0) \wedge M(P(1) \wedge M(P(2) \wedge \dots)))$$

This is what is done implicitly in H. 2012 (direct style) and Miquey 2017 (classical realisability interpretation). An extension to dependent streams is then possible, eventually giving a proof of the axiom of dependent choice.

How to scale to DNS on an arbitrary domain?

The above proofs crucially rely on the countability of the domain.

To adapt the proofs to proof of AC_{ABR}^M on any A , one would typically need to replace the test of equality on the domain by a call to excluded-middle, which in turn would require to invoke *call/cc*.

Part III

Principles equivalent to the axiom of choice

Standard equivalent forms of the general axiom of choice in set theory

Three well-known equivalent presentations in set theory:

- **axiom of choice** (AC): any family of non-empty sets has a choice function
- **Zorn's lemma** (ZL): if all chains of a non-empty partially ordered set are bounded upwards, the set has a maximal element
- **the well-ordering principle**: every set can be well-ordered (or well-founded)

Several pairs of principles equivalent to the axiom of choice

A dual family of choice and bar-induction principles

sometimes strictly weaker:

- axiom of **dependent** choice (**DC**), axiom of **countable** choice (**AC**_ω), Boolean prime ideal theorem (**BPI**), ultrafilter lemma (**UF**), Heine-Borel theorem, Hahn-Banach theorem, Gödel's completeness, ...

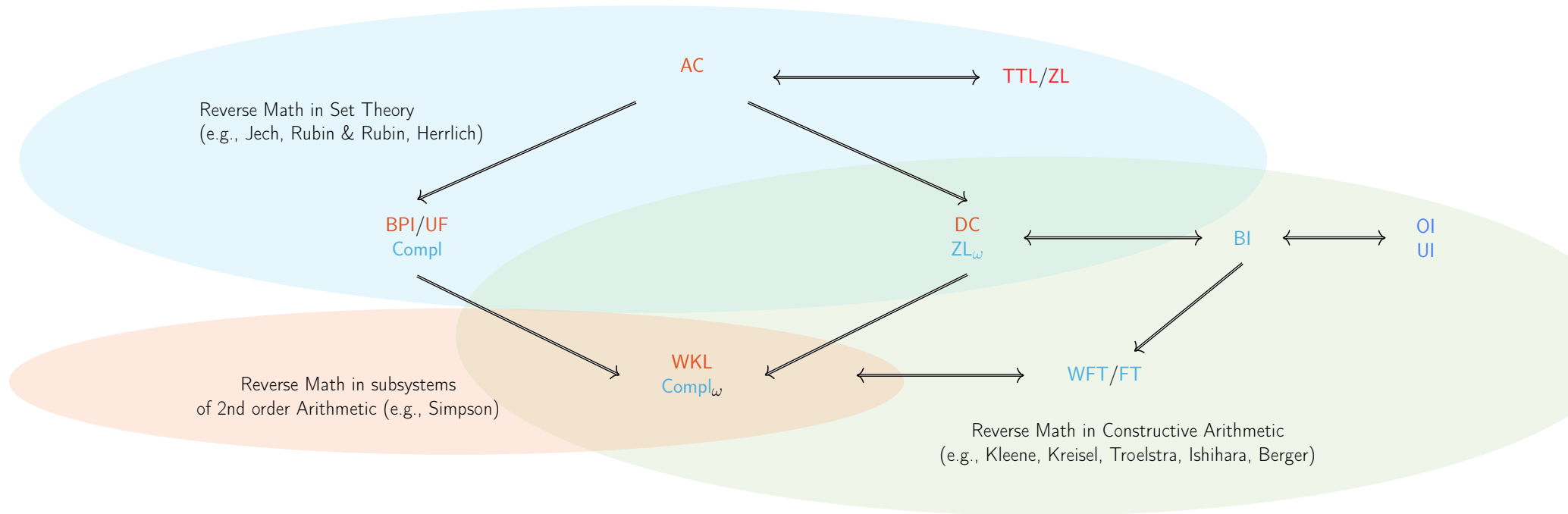
as well as variants in constructive mathematics, classically equivalent to choice or maximality principles:

- **bar induction**, its finitely-branching version **fan theorem**, ...

A dual family of maximality and well-foundedness axioms

- e.g., Teichmüller-Tukey lemma
- e.g., **Update induction** (Berger 2004)

Some standard results about choice, bar induction, maximality and well-foundedness principles



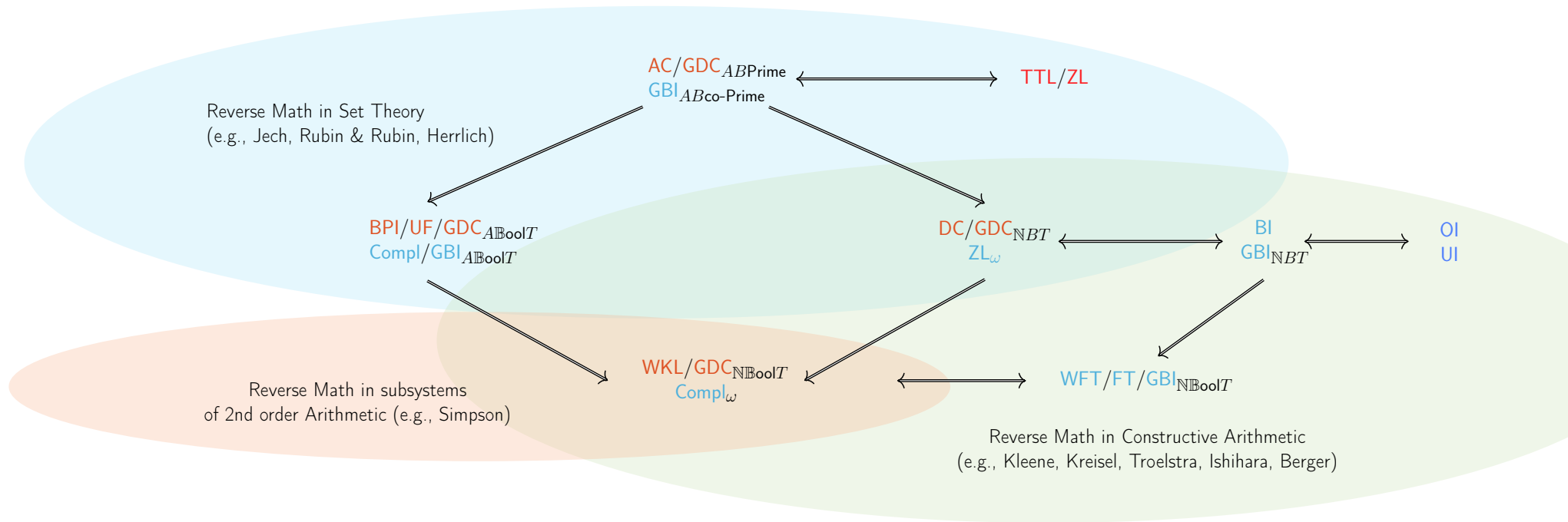
Choice axioms	
BPI	= Boolean Prime Ideal Theorem
UF	= Ultrafilter Theorem
AC	= Axiom of Choice
DC	= Axiom of Dependent Choice
WKL	= Weak König's Lemma

Maximality axioms	
TTL	= Teichmüller-Tukey Lemma
ZL	= Zorn's Lemma

Bar induction principles	
ZL _ω	= Countable Zorn's Lemma
BI	= Bar Induction
(W)FT	= (Weak) Fan Theorem
Compl	= Gödel's Completeness Theorem

Well-foundedness principles	
UI	= Update Induction
OI	= Open Induction

Some additional results about choice, bar induction, maximality and well-foundedness principles (with N. Brede, 2021)



Choice axioms

BPI = Boolean Prime Ideal Theorem
 UF = Ultrafilter Theorem
 AC = Axiom of Choice
 DC = Axiom of Dependent Choice
 WKL = Weak König's Lemma
 GDC = Generalised Dependent Choice

Bar induction principles

ZL_ω = Countable Zorn's Lemma
 BI = Bar Induction
 (W)FT = (Weak) Fan Theorem
 Compl = Gödel's Completeness Theorem
 GBI = Generalised Bar Induction

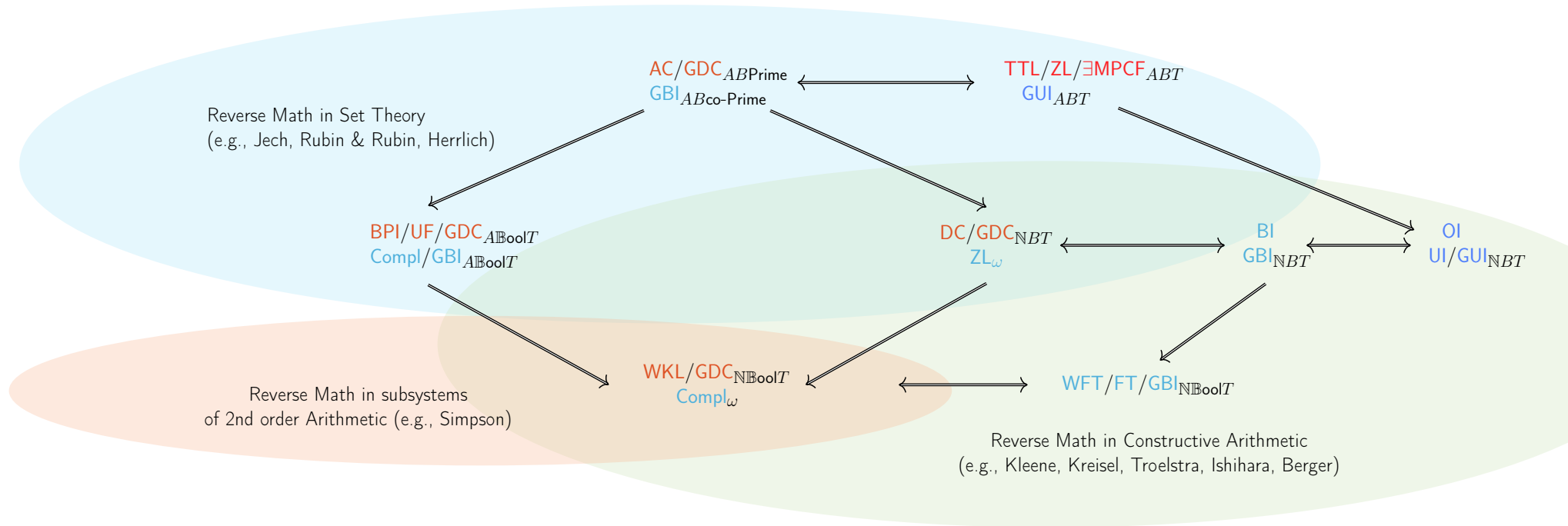
Maximality axioms

TTL = Teichmüller-Tukey Lemma
 ZL = Zorn's Lemma

Well-foundedness principles

UI = Update Induction
 OI = Open Induction

Some additional results about choice, bar induction, maximality and well-foundedness principles (with J. Koleilat, 2024)



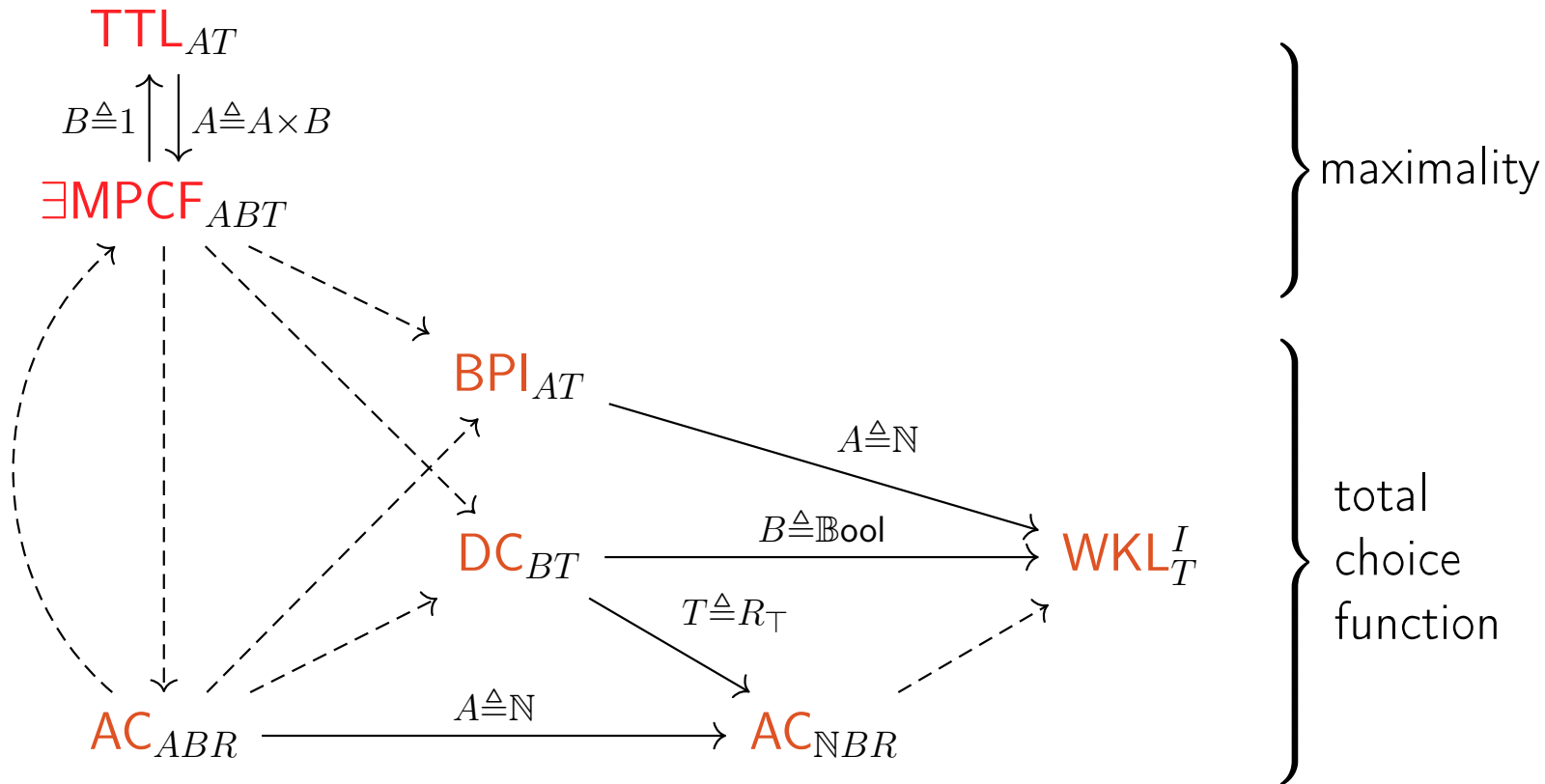
Choice axioms	
BPI	= Boolean Prime Ideal Theorem
UF	= Ultrafilter Theorem
AC	= Axiom of Choice
DC	= Axiom of Dependent Choice
WKL	= Weak König's Lemma
GDC	= Generalised Dependent Choice

Maximality axioms	
TTL	= Teichmüller-Tukey Lemma
ZL	= Zorn's Lemma
∃MPCF	= ∃ Max. Partial Choice Function

Bar induction principles	
ZL _ω	= Countable Zorn's Lemma
BI	= Bar Induction
(W)FT	= (Weak) Fan Theorem
Compl	= Gödel's Completeness Theorem
GBI	= Generalised Bar Induction

Well-foundedness principles	
UI	= Update Induction
OI	= Open Induction
GUI	= Generalised Update Induction

A hierarchy of choice and maximality axioms

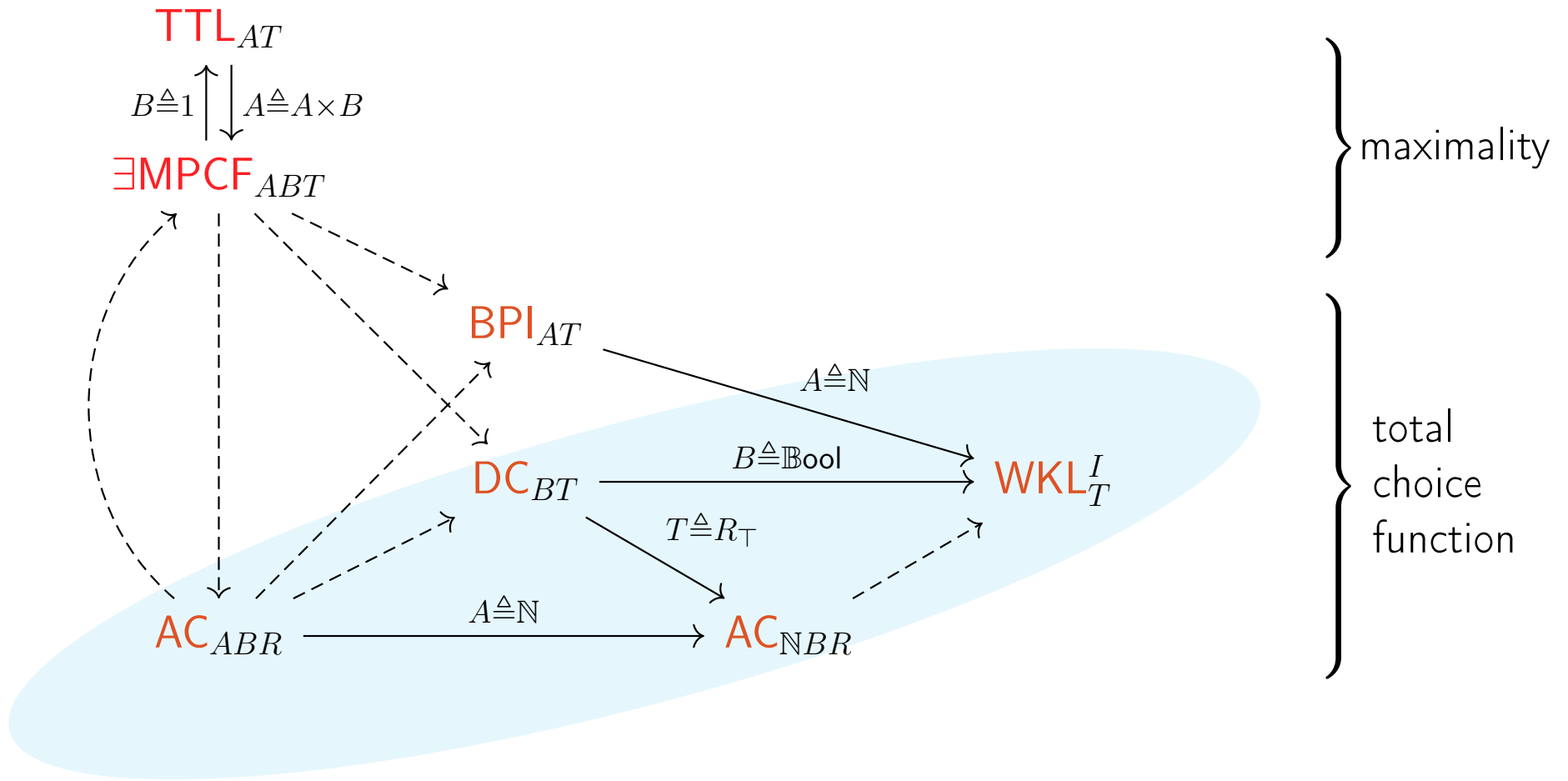


plain arrow: instantiation

dashed arrow: more complex proof (sometimes using LEM and extensionality)

Note: WKL_T^I is here an appropriate LLPO-free variant of WKL_T

Choice principles intuitionistically provable if $\exists$ is Σ

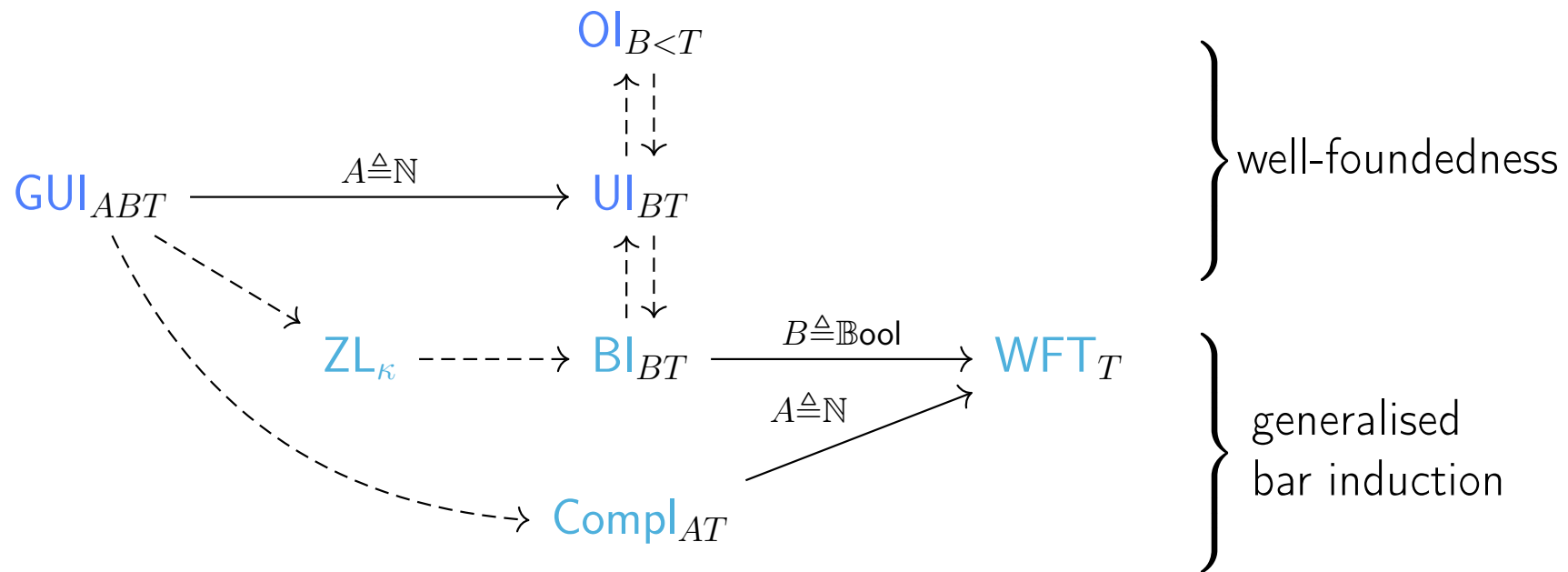


Computing with **AC** if $\exists$ is Σ : Martin-Löf

Computing with $DNS^{M, Type}$ if $\exists$ is $M(\Sigma)$ on a countable domain: Escardó-Oliva, H.

Otherwise, reason classically and try to give a computational content to the dual principles

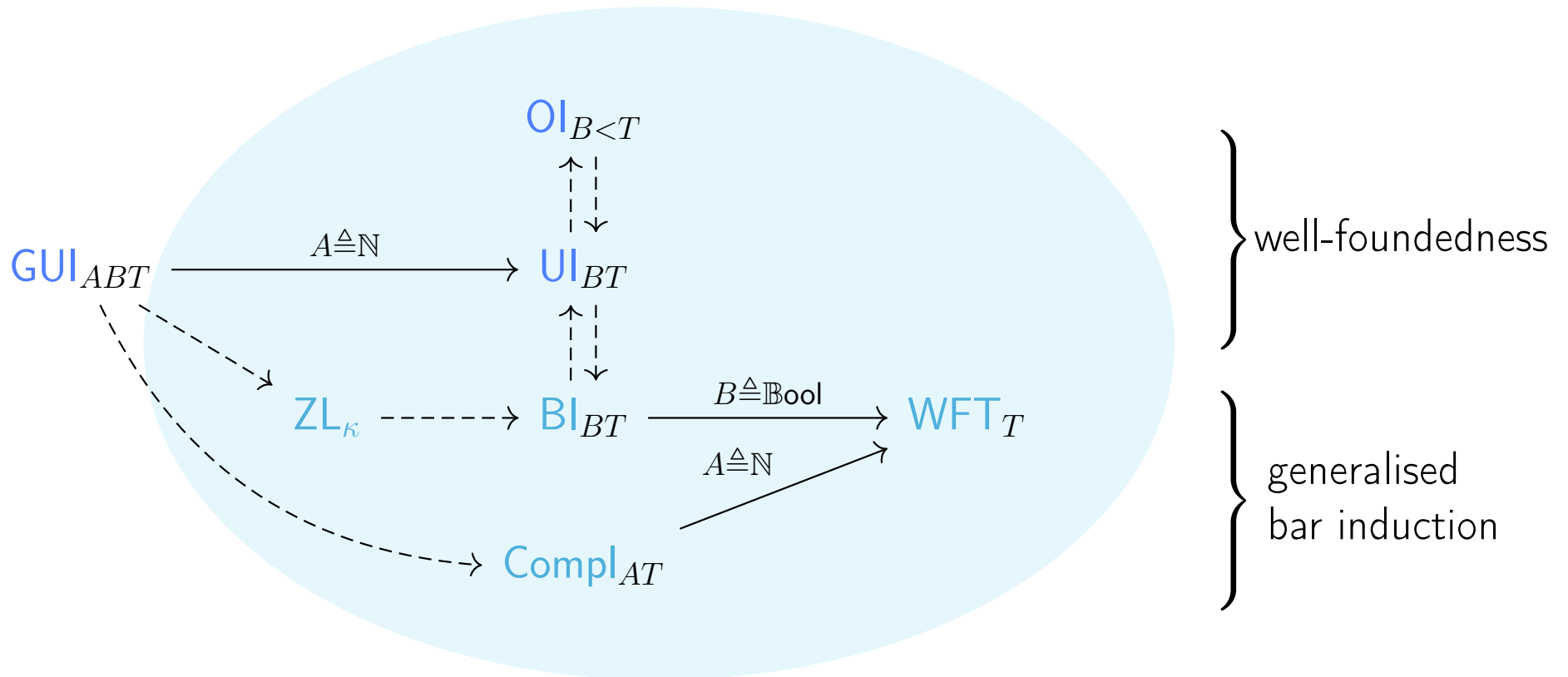
A hierarchy of bar induction and well-foundedness axioms



plain arrow: instantiation

dashed arrow: more complex proof (possibly using LEM and extensionality)

Principles for which a computational interpretation is known and described



Computing with **BI**: Spector 1962, Schwichtenberg, Bezem, Kohlenbach, U. Berger, Berardi-Bezem-Coquand, ...

Computing with **OI**: Coquand 1997, Mahboubi 2004, U. Berger 2004

Computing with **UI**: informally Berardi-Bezem-Coquand 1998, U. Berger 2004

Computing with **ZL_κ**: Fontanella-Geoffroy 2020

Computing with **Compl** on uncountable language: Krivine 2011

What is bar induction?

Let's consider first different ways to define well-foundedness

Trees (and their negative) as predicates

Let B be a domain and u ranges over the set B^* of finite sequences of elements of B . We write $\langle \rangle$ for the empty sequence and $u \star b$ for the extension of u with one element b . For T a **predicate** on B^* , we define:

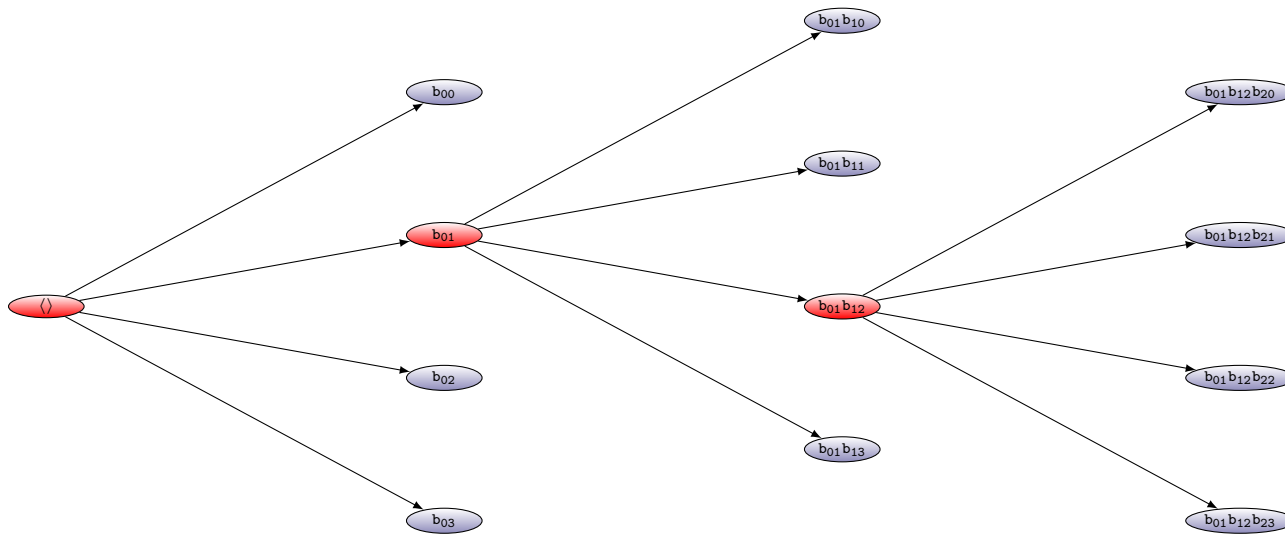
T is a tree (closure under restriction) $\forall u \forall a (u \star a \in T \rightarrow u \in T)$	T is monotone (closure under extension) $\forall u \forall a (u \in T \rightarrow u \star a \in T)$
---	---

Inductive characterisation of a well-founded tree-as-predicate

T *inductively well-founded* is short for *inductively well-founded at* $\langle \rangle \in A^*$

T *inductively well-founded at* u holds when:

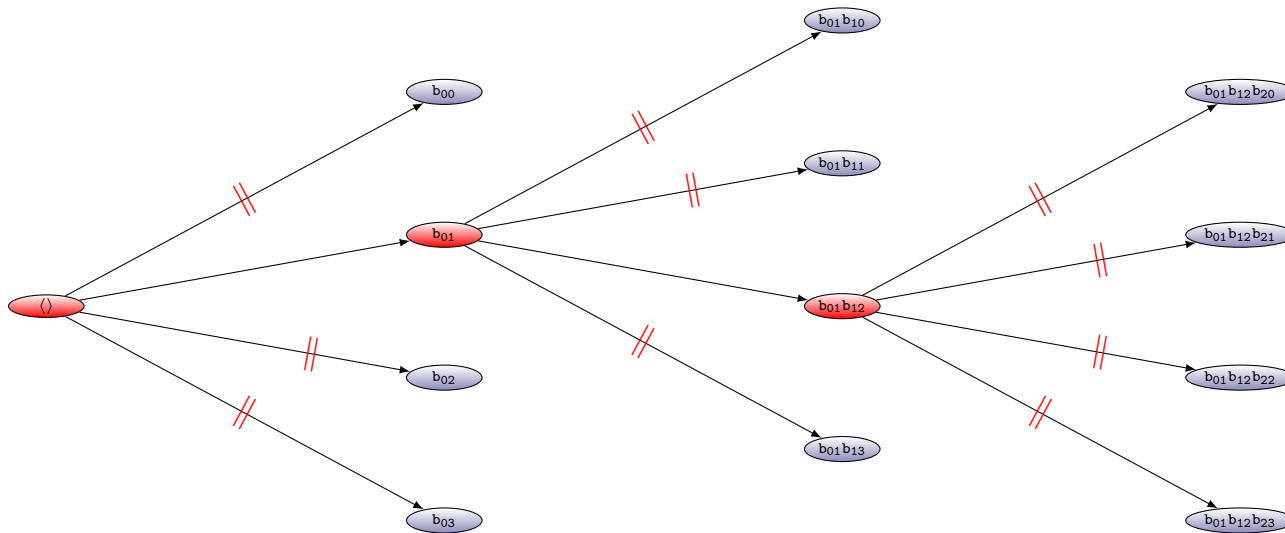
- $u \notin T$
- or, recursively, for all a , T is *inductively well-founded at* $u \star a$



Observational characterisation of a well-founded tree-as-predicate

T observationally well-founded

$$\forall \beta \in \mathbb{N} \rightarrow B. \exists n \in \mathbb{N}. \neg T(\beta|_n)$$



Two characterisations of a well-founded tree-as-predicate an observational vs intensional perspective

- From the “**effective**” representation of a **well-founded** tree we can always construct a predicate that is an “**observational**” representation of the tree
- To conversely obtain an effective representation of a tree T from its observational representation requires an axiom:

$$T \text{ *observationally well-founded* } \implies T \text{ *inductively well-founded* }$$

Bar Induction

If instead we build the **negative** of a tree-as-predicate and restate well-foundedness on the negative tree, one obtains bar induction:

- $\neg T$ *inductively well-founded* is the same as T *inductively barred*
- $\neg T$ *observationally well-founded* is the same as T *barred*
- **Bar Induction** says that for a type B and a tree T ,

$$\underbrace{T \text{ barred}}_{\text{observational}} \implies \underbrace{T \text{ inductively barred}}_{\text{effective}}$$

Dually: ill-foundedness

Dually, **ill-foundedness** of a tree T can be defined in different ways.

Let us concentrate on the finite-branching case. We have:

Effective view

$$T \text{ is staged infinite} \triangleq \forall n \exists u |u| = n \wedge u \in T$$

Observational view

$$T \text{ has an infinite branch} \triangleq \exists \alpha \forall u \leq \alpha T(u)$$

Kőnig's Lemma is a lemma that connects the two views when B is finite:

$$\mathbf{KL}_T \triangleq T \text{ is staged infinite} \rightarrow T \text{ has an infinite branch}$$

III-foundedness, coinductively

Alternatively, by dualising the notion of **inductively barred** we get another **coinductive** definition of ill-foundedness, which we call **productive**. In full:

T **productive** is short for **productive from** $\langle \rangle \in B^*$

T **productive from** $u \in B^*$ holds when:

- u is in T
- *and*, recursively, there is $b \in B$ such that T is **productive from** $u \star b$

Relying on the notion of *inductively barred* and its dual,
we obtain the following dual pair of choice and bar induction principles

Bar induction ($\mathbf{BI}_{BT}$)

T barred $\rightarrow T$ inductively barred

Tree-Based Dependent Choice ($\mathbf{DC}_{BT}^{prod}$)

T productive $\rightarrow T$ has an infinite branch

Recovering standard principles

$\text{WKL}_T \iff \text{DC}_{\mathbb{B}\text{ool}T}^{\text{prod}}$ up to classical (actually co-intuitionistic) reasoning

$\text{WFT}_T \iff \text{BI}_{\mathbb{B}\text{ool}T}$ up to intuitionistic reasoning

$\text{DC}_{BRb_0}^{\text{serial}} \iff \text{DC}_{BR^\triangleright(b_0)}^{\text{prod}}$

where

$$u \in R^\triangleright(b_0) \triangleq \text{case } u \text{ of } \left[\begin{array}{ll} \langle \rangle & \mapsto \top \\ b & \mapsto R(b_0, b) \\ u' \star b \star b' & \mapsto R(b, b') \end{array} \right]$$

$$\text{DC}_{BRb_0}^{\text{serial}} \triangleq \forall b \exists b' R(b, b') \rightarrow \exists \alpha (\alpha(0) = b_0 \wedge \forall n R(\alpha(n), \alpha(n+1)))$$

(one of the most standard statements of dependent choice)

Bar recursion

Bar recursion

Bar recursion is a recursive program that “proves” bar induction. Historically, it was thought as a stand-alone program in some computational model. We present it here as a typed proof term.

Bar recursion on arbitrary bars would require classical logic, but there are several constructive alternatives:

- Bar recursion on monotone bars
- Bar recursion on decidable bars
- Bar recursion on thin bars

Reminder: inductively barred and observationally barred

We say that $P : A^* \rightarrow \mathbf{Prop}$ is *observationally barred* (or simply *barred*) when

$$\forall f^{\mathbb{N} \rightarrow A} \exists n P(f|_n)$$

We say that P is *inductively barred* when it is *inductively barred at* $\langle \rangle$ where:

$$\frac{P(u)}{P \text{ inductively barred at } u} \quad \text{Now} \quad \frac{\forall x (P \text{ inductively barred at } (u \star x))}{P \text{ inductively barred at } u} \quad \text{Later}$$

Naive bar recursion

Let P be a predicate on A^* .

Assume A non empty and a_0 a distinguished inhabitant. For u a sequence, we write $\hat{u}$ for the extension of u into an infinite sequence:

$$\hat{u} \triangleq \lambda n. \text{if } n < |u| \text{ then } u(n) \text{ else } a_0$$

Then, for $b : P$ *barred* (i.e. $b : \forall f^{\mathbb{N} \rightarrow A} \exists n P(f|_n)$), we recursively define a proof $BR_b(u)$ of P *inductively barred at* u by:

match $b(\hat{u})$ with

$$\begin{aligned} BR_b^{naive}(u) &\triangleq \mid (n, a) \Rightarrow \text{if } n < |u| \text{ then Now}(\textcolor{red}{u}) \text{ else Later}(\lambda x. BR_b^{naive}(u \star x)) \\ &\text{end} \end{aligned}$$

... except that we only know $P(u|_n)$, not $P(u)$, so this is wrong.

Bar recursion on monotone bars

Assume $m : P$ monotone, where:

$$P \text{ monotone} \triangleq \forall u^{A^*} \forall u'^{A^*} (u \leq u' \wedge P(u) \rightarrow P(u'))$$

Then, we can define:

match $b(\hat{u})$ with

$\text{BR}_{bm}^{mon}(u) \triangleq \mid (n, a) \Rightarrow \text{if } n < |u| \text{ then Now}(m(u|_n, u, a)) \text{ else Later}(\lambda x. \text{BR}_{bm}^{mon}(u \star x))$
end

Monotone bar recursion is then a proof of monotone bar induction BI_{mon} defined by

$$\text{BR}^{mon} \triangleq \lambda b. \lambda m. \text{BR}_{bm}^{mon}(\langle \rangle)$$

Note: the termination relies on Bar Induction at the meta-level, taking “ $\text{BR}(u)$ terminates” as bar.

Bar recursion on decidable bars

Assume P *decidable* in the sense that there is $f : A^* \rightarrow \mathbb{Bool}$ such that:

$$d : \forall u (f(u) = \mathbf{true} \leftrightarrow P(u))$$

Then, we can define:

$$\mathbf{BR}_{bfd}^{dec}(u) \triangleq \text{if } f(u) \text{ then Now}(d(u)) \text{ else Later}(\lambda x. \mathbf{BR}_{bfd}^{dec}(u \star x))$$

Decidable bar recursion is then a proof of decidable bar induction $\mathbf{BI}_{dec}$ defined by

$$\mathbf{BR}^{dec} \triangleq \lambda b. \lambda f. \lambda d. \mathbf{BR}_{bfd}^{dec}(\langle \rangle)$$

which, again, is terminating by bar induction at the meta-level.

Bar recursion on thin bars

Assume $d : P$ thin where

$$P \text{ thin} \triangleq \forall u (P(u) \rightarrow \forall u' < u \neg P(u'))$$

Then, for $H : \forall n < |u| \neg P(\hat{u}_{|n})$, we can define:

$$\begin{aligned} & \text{match } b(\hat{u}) \text{ with} \\ & \quad \text{if } n < |u| \text{ then efq } (H \ n \ a) \text{ else} \\ \text{BR}_{bd}^{thin}(u, H) & \triangleq \mid (n, a) \Rightarrow \text{if } n = |u| \text{ then Now}(a) \text{ else} \\ & \quad \text{Later}(\lambda x. \text{BR}_{bd}^{thin}(u \star x, g(H, d, a))) \\ & \text{end} \end{aligned}$$

where $g(H, d, a)$ extends H to $u \star x$, using d .

Thin bar recursion is then a proof of thin bar induction BI_{thin} defined by

$$\text{BR}^{thin} \triangleq \lambda b. \lambda d. \text{BR}_{bd}^{thin}(\langle \rangle)$$

which, again, is terminating by bar induction at the meta-level.

Literature says that $\text{BI}_{mon} \rightarrow \text{BI}_{dec}$ and that $\text{BI}_{dec} \leftrightarrow \text{BI}_{thin}$.

Historical notes: on the relation between DNS and BI

Bar recursion was introduced by Spector 1962 to realise the “Dialectica” functional interpretation of DNS but, **counter-intuitively**, its structure is the one of a program of type BI.

Conversely, what was called modified bar recursion has, **counter-intuitively**, a type of the form of DNS (up to details).

Maximality and well-foundedness principles

Generalised update induction

Update Induction (GUI_{ABT})

(if the upwards monotone closure of T is $\prec$ -inductive, it contains all partial functions from A to B)

Or, fully formally:

$$(\forall \alpha^{A \rightarrow B} (\forall \beta \prec \alpha (\mathcal{G}(\beta) \in \langle T \rangle^\circ)) \rightarrow (\mathcal{G}(\alpha) \in \langle T \rangle^\circ)) \rightarrow \forall \alpha (\mathcal{G}(\alpha) \in \langle T \rangle^\circ)$$

where the upwards monotone closure of T is:

$$\langle T \rangle^\circ \triangleq \lambda \alpha^{A \times B \rightarrow \text{Prop}}. \exists u^{(A \times B)^*} (u \subset \alpha \wedge u \in T)$$

the type of partial function is:

$$A \multimap B \triangleq A \rightarrow B_\perp \quad \text{where } B_\perp \text{ is } B \text{ lifted with an extra element } \perp$$

the refinement of partial functions is:

$$\beta \prec \alpha \triangleq \exists a^A (\alpha(a) = \perp \wedge \beta(a) \neq \perp) \wedge \forall a (\alpha(a) \neq \perp \rightarrow \beta(a) = \alpha(a))$$

the graph of a partial function is:

$$\mathcal{G}(\alpha) \triangleq \lambda(a, b). \alpha(a) = b$$

Berger's update induction

Berger's update induction, 2004, is morally GUI_{NBT} .

It computes with a fixpoint:

$$\begin{aligned} \text{UR} &: (\forall \alpha^{\mathbb{N} \rightarrow B} (\forall \beta \prec \alpha (\mathcal{G}(\beta) \in \langle T \rangle^\circ)) \rightarrow (\mathcal{G}(\alpha) \in \langle T \rangle^\circ)) \rightarrow \forall \alpha (\mathcal{G}(\alpha) \in \langle T \rangle^\circ) \\ &\triangleq \lambda H. \lambda \alpha. (H \alpha \lambda \beta. \lambda H'. (\text{UR } H \beta)) \end{aligned}$$

That is: even if the approximation order is not well-founded in general, it is well-founded over predicates which can observe only a finite part of a function, what being of the form $\langle T \rangle^\circ$ means.

Computing with open induction: open recursion

See Berger 2004 or Powell 2014

An alternative computational content for dependent choice on predicates using quote (Krivine 2003)

Here is a rephrasing as a proof of Krivine's classical "quote"-based program for choice of predicates. It relies on the assumption in the language for any $P : \mathbb{N} \rightarrow \mathbf{Prop}$ of a symbol of function ϕ_P such that $\exists X P(X) \rightarrow \exists n P(\phi_P(n))$.

$$\begin{aligned} \mathbf{AC}_{\mathbb{N}} &\triangleq \lambda a. (U_R, \lambda x. \mathbf{match} \chi(a x) \mathbf{with} (n, b) \Rightarrow \mathbf{catch}_{\alpha} \mathbf{wf}_x (\lambda n'. \lambda f. \lambda b'. \mathbf{throw}_{\alpha} (\uparrow_{R(x, Y)}^{n' f b'} b')) n b \mathbf{end}) \\ &: \forall x^{\mathbb{N}} \exists Y^{\mathbb{N} \rightarrow \star} R(x, Y) \rightarrow \exists U^{\mathbb{N} \rightarrow \mathbb{N} \rightarrow \star} \forall x^{\mathbb{N}} R(x, U(x)) \end{aligned}$$

where:

$$\begin{aligned} V(x, n) &\triangleq \neg P(x, \Phi_P(x, n)) \\ Z(x, n) &\triangleq \forall m < n V(x, m) \rightarrow V(x, n) \\ U_P(x) &\triangleq \forall n (\neg Z(x, n) \rightarrow \Phi_P(x, n)) \end{aligned}$$

and

$$\mathbf{wf}_x : \forall n Z(x, n) \rightarrow \forall n V(x, n) \quad \text{is some fixpoint operator}$$

An alternative computational content for dependent choice on predicates using quote (Krivine 2003), continued

We also need, for $f : \forall m < n V(x, m)$ and $b : P(x, \Phi_P(x, n))$, a proof to reason up to propositional equivalence:

$$\uparrow_A^{nfb} : A(\Phi_P(x, n)) \rightarrow A(U_P(x))$$

$$\uparrow_{Y(t)}^{nfb} c \triangleq \lambda n'. \lambda k. \text{if } n = n' \text{ then } c \text{ else } k \lambda f'. \lambda b'. \text{if } n' < n \text{ then } f n' b' \text{ else } f' n b$$

$$\uparrow_{A \wedge B}^{nfb} c \triangleq (\uparrow_A^{nfb} (\pi_1 c), \uparrow_B^{nfb} (\pi_2 c))$$

$$\uparrow_{A \rightarrow B}^{nfb} c \triangleq \lambda a. (\uparrow_B^{nfb} (c (\downarrow_A^{nfb} a)))$$

...

$$\downarrow_A^{nfb} : A(U_P(x)) \rightarrow A(\Phi_P(x, n))$$

$$\downarrow_{Y(t)}^{nfb} c \triangleq c n \lambda k. k f b$$

$$\downarrow_{A \wedge B}^{nfb} c \triangleq (\downarrow_A^{nfb} (\pi_1 c), \downarrow_B^{nfb} (\pi_2 c))$$

$$\downarrow_{A \rightarrow B}^{nfb} c \triangleq \lambda a. (\downarrow_B^{nfb} (c (\uparrow_A^{nfb} a)))$$

...

Computing with the ultrafilter theorem and the wellordering of $2^{\mathbb{N}}$

See a program (apparently) combining forcing and realisability in Krivine 2011.

Computing with the full axiom of choice

Gödel's constructible sets model in **ZF** restricts the quantification on sets to sets definable by a formula of **ZF**. Impredicativity is interpreted by well-founded induction.

See also an approach by Krivine 2021.

Continuity and bar induction

After Bauer, Escardó and others, Baillon, Forster, Mahboubi and Pédrot investigated the close connections between bar induction and continuity.

Partial summary of realisers for choice, bar induction, maximality, well-foundedness axioms

Axiom	Logic		
AC^{int}	type theory	Martin-Löf	
AC^{neg}	ZF	Gödel's constructible sets (abstract) and probably many other models	
AC^{neg}	ZF	Krivine 2021 (complex)	
AC^{neg}	ZF_ϵ	Krivine 2003	classical minimisation through quote
DC_κ/ZL_κ	ZF	Fontanella-Geoffroy 2020	classical minimisation through quote
BPI/UF	PA_2	Krivine 2010	forcing
$WO_{2^{\mathbb{N}}}$	PA_2	Krivine 2010	forcing
BI	HA	Spector 1962, ...	bar recursion, Π -monad distributivity
OI	HA	Mahboubi, Berger 2004	open recursion
UI	HA	Berger 2004	well-founded fixpoint

Not addressed here: Hilbert's ϵ , Church's ι , constructive ϵ as dual of DNS^{Type} , ...

Bibliography

Ulrich Berger, A Computational Interpretation of Open Induction, LICS 2004

Ulrich Berger and Paulo Oliva, Modified bar recursion, MSCS 2006

Martín Escardó and Paulo Oliva, Bar recursion and products of selection functions, 2010

Laura Fontanella and Guillaume Geoffroy, Preserving cardinals and weak forms of Zorn's lemma in realizability models, 2020

Hugo Herbelin, A Constructive Proof of Dependent Choice, Compatible with Classical Logic, LICS 2012

Jean-Louis Krivine, A program for the full axiom of choice, LMCS, 2021

Jean-Louis Krivine, Dependent choice, 'quote' and the clock, TCS, 2003

Jean-Louis Krivine, Realizability algebras: a program to well order $\mathbb{R}$, 2010

Assia Mahboubi, An Induction Principle over Real Numbers, 2016

Thomas Powell, On the computational content of Zorn's lemma, LICS 2020

Thomas Powell, The equivalence of bar recursion and open recursion, 2014

Thomas Powell, Peter Schuster and Franziskus Wiesnet, A universal algorithm for Krull's theorem, Information and Computation, 2022

...

Appendix

Inference rules for $\exists$ in first-order logic

Intrinsic style:

$$\frac{\Gamma \vdash_I P[x \leftarrow t]}{\Gamma \vdash_I \exists x^A P} \quad \exists_I \qquad \frac{\Gamma \vdash_I \exists x^A P \quad \Gamma, x : A, P \vdash_I R}{\Gamma \vdash_I R} \quad \exists_E$$

With proof terms:

$$\frac{\Gamma \vdash_I p : P[x \leftarrow t]}{\Gamma \vdash_I (t, p) : \exists x^A P} \quad \exists_I \qquad \frac{\Gamma \vdash_I p : \exists x^A P \quad \Gamma, x : A, a : P \vdash_I q : R}{\Gamma \vdash_I \text{match } p \text{ with } (x, a) \Rightarrow q \text{ end} : R} \quad \exists_E$$

In particular, no extraction of the witness is made possible.

Inference rules for Σ -types (aka strong $\exists$)

$$\frac{\Gamma \vdash_I p : P[x \leftarrow t]}{\Gamma \vdash_I (t, p) : \Sigma x^A P} \quad \Sigma_I$$

$$\frac{\Gamma \vdash_I p : \Sigma x^A P}{\Gamma \vdash_I \pi_1 p : A} \quad \Sigma_E^1 \qquad \frac{\Gamma \vdash_I p : \Sigma x^A P}{\Gamma \vdash_I \pi_2 p : P[x \leftarrow \pi_1, p]} \quad \Sigma_E^2$$

Note: In an intuitionistic setting with η -rules, it does not matter whether Σ -types are defined positively (as in Coq, from dependent elimination) or negatively (from projections).

Existential quantifier from Σ -types by propositional truncation

In type theory, propositional truncation restricts back elimination to elimination in propositions:

Truncation is a type:

$$P ::= \dots \mid ||P||$$

It adds proofs:

$$p ::= \dots \mid |p| \mid \mathbf{let} \ |a| := p \ \mathbf{in} \ q$$

corresponding to the following inference rules:

$$\frac{\Gamma \vdash p : P}{\Gamma \vdash |p| : ||P||} \qquad \frac{\Gamma \vdash p : ||P|| \quad \Gamma, a : P \vdash q : Q(|a|)}{\Gamma \vdash \mathbf{let} \ |a| := p \ \mathbf{in} \ q : Q(p)}$$

Weak existential quantification can be recovered from strong existential and truncation:

$$\exists x^T P(x) \triangleq ||\Sigma x^T P(x)||$$

Remark: In the presence of classical logic, $||\Sigma x^T P(x)||$ is equivalent to $\neg\neg\Sigma x^T P(x)$ (see HoTT Book).