# Oracles Just for Fan

## A Robust Computational Interpretation of the Fan Theorem

**Titouan Leclercq**
Aix Marseille Université, CNRS, I2M, Marseille, France

**Étienne Miquey**
Aix Marseille Université, CNRS, I2M, Marseille, France

**Abstract**

Friedman-Simpson's original program of reverse mathematics, as is also the case for most of standard mathematics, has been developed in classical subsystems of second-order arithmetic. As such, (classical) reverse mathematics presents various limitations from a constructive point of view, since for instance it is unable to distinguish between a statement and its contrapositive (e.g. dependent choice and the bar induction principles). The case of (Weak) Kőnig's Lemma (WKL) and Fan Theorem (FT) is particularly interesting in that regard: while WKL is well-known to imply FT, and if constructivists like Brouwer rejected the former while admitting the latter, the converse implication has not been much studied for years. It is only recently that a growing enthusiasm for constructive reverse mathematics pushed towards a finer-grained analysis of the connection between such principles.

In addition to intuitionistic reverse mathematics, the realizability approach to logical principles adds a computational meaning to purely logical statements. We follow this path to investigate the computational meaning of Brouwer's Fan Theorem: building on recent work by Lubarsky and Rathjen, we first construct a realizability interpretation of higher-order logic validating FT while refuting WKL. This interpretation relies on a $\lambda$-calculus extended with oracles while preserving a notion of continuity for realizers.

We then push this approach a step further to show the robustness of this realizability interpretation by identifying, in the abstract and general setting of evidenced frames, sufficient computational conditions entailing FT.

**2012 ACM Subject Classification** Theory of computation → Constructive mathematics

**Keywords and phrases** Constructive Mathematics, Reverse mathematics, Fan Theorem, Oracles, Realizability, Evidenced frames

**Digital Object Identifier** 10.4230/LIPIcs.LICS.2026.63

**Funding** This work was supported by the ANR-23-CE48-0016 CHoICE project.

## 1 Introduction

The program of reverse mathematics, first initiated by Friedman in the 70s, aims at answering questions of the form: "*what are the minimal axioms necessary to derive the theorem $\varphi$?*". To do so, formulas are classified via their logical equivalence classes over the ambient theory $\mathcal{T}$, which has then to be chosen carefully. Indeed, $\mathcal{T}$ has to be expressive enough for usual mathematical statements to have meaning, while a theory too strong would identify all the provable formulas. Classical reverse mathematics has now identified a well-established hierarchy of subsystems of second-order arithmetic serving this purpose [39], but the literature in constructive reverse mathematics is much more recent and the situation is more complex: several principles compatible with theories used in constructive mathematics lead to logical inconsistencies when added together (*e.g.*, Church's thesis and the law of excluded-middle).


Among the various limitations that classical reverse mathematics presents from a constructive point of view, one is that it is intrinsically bound to classify theorems that are compatible with classical logic (which is not the case for all intuitionistic theorems), and is unable to distinguish between a statement and its contrapositive. More recently, Ishihara advocated for a constructive approach to reverse mathematics to overcome such limitations [27], but with purely logical considerations, without paying attention to the computational counterpart of constructive results. It is only during the last few years that works in that direction have been developed, especially around Gödel's completeness theorem for first-order logic [24, 15]. We advocate for a generalization of this approach, which has shown to be very conducive over the past few years [29, 9, 25, 2].

In a recent paper, Brede & Herbelin put forward a generalized variant of the dependent choice $\mathrm{GDC}_{ABT}$, and its contrapositive bar induction principle $\mathrm{GBI}_{ABT}$, expressed in terms of trees [6]. Different instantiations of the parameters $A$, $B$, $T$, which refer to the objects to which they apply, allow to capture a structured hierarchy of choice principles, ranging from Weak König's Lemma (when $A$ is the type $\mathbb{N}$ of natural numbers and $B$ is the type $\mathbb{B}$ of Booleans) to the Axiom of Choice (when $T$ is a filter induced by the considered relation), together with their contrapositive. Following the recent advances in constructive reverse mathematics, we suggest to consider these principles with a computational point of view, and especially the question:

"*What is the computational content of each of these principles?*"

Addressing such a question raises several difficulties. First and foremost, we need to understand how to relate advanced logical principles with computational features. To that end, we rely on realizability interpretations, and especially on the recent literature in lines with Krivine approach which unveiled that adding programming instructions to a programming language can lead to interpreting new reasoning principles [21, 33, 36, 38, 19, 8, 5]

Besides, we need to figure out what it means to capture *exactly* the computational content of a logical principle. We believe that a satisfying answer to that question should address the three following points.

**(I)** Is the computational content exhibited enough? It should be the fact that the expected principle is validated not because of some implicit restrictions, but in a setting that is also compatible with further computational extensions. For instance, the Axiom of Choice is trivially valid in a setting with strong existentials (such as Martin-Löf Type Theory or intuitionistic realizability interpretations based on a purely functional language) but this is due to the implicit restrictions that a realizer of $A \to B$ is necessarily a function. This does not extend to extension with effects, allowing for a coin flip is enough to refute it [8]. In contrast, several works show how Countable Choice can be validated using memoization techniques in presence of effects [3, 23, 38, 8].

**(II)** How can we know that the computational features we consider are not too strong? For instance, it is clear that if memoization techniques allow to validate Countable Choice in presence of classical logic, such settings will also entail Kőnig's Lemma and the Fan Theorem. We therefore suggest to look for realizability models providing separation results: *e.g.*, to capture the exact computational content of Countable Choice, one should aim for a model refuting Dependent Choice. As the computational content does not, by itself, give a separation result, we want the following statement to hold for a computational content (a) to characterize the logical principle (b):
*" there exists a model exhibiting (a) and separating (b) from its stronger variants"*
which, in the present case, means that we search for a model separating FT from WKL.

**(III)** We should aim to offer robust interpretations, *i.e.* to ensure that the validity of the interpretation is only a consequence of the studied computational features (say, memoization) and not peculiar to specificities of the underlying language of realizers (say, some $\lambda$-calculus). To that end, we suggest relying on *evidenced frames* [12], which provide an algebraic representation of realizability models and allow us to express robust statements such as: *"any computational system providing an implementation of* (a) *will induce an evidenced frame validating* (b)". In particular, Cohen *et al.* proved that memoization techniques (a) in any of a variety of computational systems would yield a realizability model of Countable Choice (b) [8, 12], which is a stronger result than the definition of a specific model of the same axiom in [23, 38] based on an ad-hoc variant of the $\lambda$-calculus with some implementation of memoization.

### The computational content of the Fan Theorem

In this paper, we focus on the lowest principle of Brede-Herbelin hierarchy, namely Brouwer's Fan Theorem (FT). Considering trees as prefix-closed sets of lists of booleans, let $C$ be a set whose complement is a tree (hence, a set closed under list extension).

$C$ is said to be *barred* if it contains a finite prefix $\alpha\restriction_n$ of any infinite sequence $\alpha \in \mathbb{B}^{\mathbb{N}}$. It is *uniformly barred* if there exists a natural number $n$ such that for any $\alpha \in \mathbb{B}^{\mathbb{N}}$, $\alpha\restriction_n \in C$. The Fan Theorem states that if such a set is barred, then it is uniformly barred. This can be understood as the contrapositive of Weak Kőnig's Lemma (WKL), which states that if a tree is infinite then it contains an infinite path.

In particular, WKL entails FT. As explained, constructive reverse mathematics is much more subtle than classical mathematics, which explains that even though a principle such as Brouwer's continuity principle was first stated more than a century ago [7], continuity principles and their consequences, in particular FT, are still actively studied in constructive settings [41, 43, 4, 16, 17, 1, 13].

### Contributions of this paper

In this work, we provide a computational interpretation of the Fan Theorem in line with our manifesto. To this end, we construct a realizability model that validates FT while refuting WKL, and we subsequently show that the induced interpretation is itself robust. Our construction is inspired by the work of Lubarsky and Rathjen [35], who sketch a model combining Kripke semantics with realizability. Their approach relies on a sequence of reals of strictly increasing Turing degrees assigned to the worlds, which serve as oracles; at each world, the computable functions relative to the corresponding oracle are used to define a PCA-based realizability model.

To unveil a robust computational content of the Fan Theorem, we proceed as follows. First, in Section 2, we introduce a computationally meaningful syntax for logical statements about trees, designed to better capture the computational behavior of realizers. In Section 3, we then define a realizability interpretation based on a $\lambda$-calculus extended with oracles, in the spirit of Goodman's relativized realizability [20]. In Section 4, we establish a continuity property of this system and use it to exhibit a simple $\lambda$-term realizing FT. We also show that our model refutes WKL, thereby obtaining a separation result.

In a second phase, presented in Section 5, we isolate the key computational features of our construction – namely, continuity of realizers, strong existential quantification, and unbounded search. Using the notion of evidenced frames, we then prove a robust version of the validity of FT, showing that any computational system exhibiting these same features yields a valid interpretation of the Fan Theorem.

## 2 A computably relevant syntax for trees

The model we construct is based on higher-order logic (HOL). This is a multi-sorted logic with a sort Prop denoting the set of propositions (hence the higher-order qualification). The choice of HOL as our syntax is mainly motivated by its expressiveness. In HOL, it is possible to easily talk about predicates on sets over a given sort $S$, by the sort $S \to \mathrm{Prop}$.

In realizability models, universal quantifications are usually interpreted as intersections without computational content, for instance having truth values of first-order universal quantifications satisfying $|\forall x, \varphi(x)| = \bigcap_{n\in\mathbb{N}} |\varphi(n)|$. In particular, a realizer in $|\forall x, \varphi(x)|$ should realize $\varphi(n)$ for any $n \in \mathbb{N}$ *without* knowing the value of $n$. Dually, realizers of existential statements witness the existence of a valid instantiation for the existentially quantified variable without actually computing its value. However, we seek a computational interpretation of the Fan Theorem such that *its realizer actually computes* the uniform bound for the bar. To overcome this, we rely on a standard technique and use *relativized quantifications*: instead of $\forall x, \varphi(x)$, we consider formulas of the shape $\forall x, \mathrm{Nat}(x) \to \varphi(x)$, where the proposition $\mathrm{Nat}(x)$ states that $x$ is indeed a natural number. A realizer of this statement will now have to realize $\mathrm{Nat}(n) \to \varphi(n)$, that is to realize $\varphi(n)$ provided a realizer of $\mathrm{Nat}(n)$, *i.e.* a term computing the value of $n$. The realizability model is introduced in Section 3, we begin with the definition of HOL formulas for our interpretation.

In order for our realizers to compute with trees, that is to say with paths (via functions in $\mathbb{N} \to \mathbb{B}$), prefixes of paths (*i.e.* lists on booleans) and (decidable) sets of lists, we therefore consider sorts accounting for natural numbers, lists, booleans, propositions and functions, while formulas are usual HOL propositions extended with a relativization predicate for each sort.

▶ **Definition 1** (Sorts). *The set of sorts $\mathfrak{S}$ is inductively defined by:*

$$S, T ::= \mathrm{Nat} \mid \mathrm{Bool} \mid \mathrm{List}(S) \mid S \times T \mid S \to T \mid \mathrm{Prop}$$

▶ **Definition 2** (Formulas). *Formulas are inductively defined by:*

$$\varphi, \psi ::= S(\mathbf{t}) \mid \varphi \to \psi \mid \varphi \wedge \psi \mid \forall x^S, \varphi \mid \exists x^S, \varphi \qquad (S \in \mathfrak{S})$$

*where $\mathbf{t}$ is a HOL-term (c.f. Definition 3)*

With this set of sorts, we give a syntax for well-typed terms. We use bold font for the syntactic HOL terms, to distinguish them from the $\lambda$-calculus terms we introduce later. The only exception is to denote an element in Prop, for which we use $\varphi, \psi \ldots$

The HOL term constructors are akin to system T: there are constructors for abstraction and application for functions, constructors for pairing and projections, canonical constructors and recursor the different data types ($\mathrm{Nat}, \mathrm{Bool}, \mathrm{List}$). The main difference lies in the logical component, as we also add propositional constructors for the formulas above. As usual, we fix a denumerable set $\mathcal{X}$ of variables which we denote by $\mathbf{x}, \mathbf{y}, \ldots$ and we define *(HOL-)contexts* as lists of pairs $(\mathbf{x} : S)$ where $\mathbf{x} \in \mathcal{X}$ and $S \in \mathfrak{S}$. We denote the set of (HOL-)contexts by Ctx and will write its elements $\Gamma, \Delta, \ldots$

▶ **Definition 3** (HOL-terms). *A well-typed (HOL-)term $\mathbf{t}$ of sort $S$ in a context $\Gamma$, which we denote by $\Gamma \vdash \mathbf{t} : S$, is defined by the inductive relation given in Figure 1.*

A model of such a syntax must give a semantic interpretation of each sort. In particular, there must be an interpretation for the sort Prop. From the constructors of this sort, the interpretation of Prop must have a function interpreting $\wedge$ and $\to$, and functions interpreting

$$\frac{(\mathbf{x} : S) \in \Gamma}{\Gamma \vdash \mathbf{x} : S} \qquad \frac{\Gamma, \mathbf{x} : S \vdash \mathbf{t} : T}{\Gamma \vdash \boldsymbol{\lambda}\mathbf{x}.\mathbf{t} : S \to T} \qquad \frac{\Gamma \vdash \mathbf{t} : S \to T \qquad \Gamma \vdash \mathbf{u} : S}{\Gamma \vdash \mathbf{t}(\mathbf{u}) : T}$$

$$\frac{\Gamma \vdash \mathbf{t} : S \qquad \Gamma \vdash \mathbf{u} : T}{\Gamma \vdash \langle \mathbf{t}, \mathbf{u} \rangle : S \times T} \qquad \frac{\Gamma \vdash \mathbf{t} : S \times T}{\Gamma \vdash \boldsymbol{\pi}_1(\mathbf{t}) : S} \qquad \frac{\Gamma \vdash \mathbf{t} : S \times T}{\Gamma \vdash \boldsymbol{\pi}_2(\mathbf{t}) : T} \qquad \frac{}{\Gamma \vdash \mathbf{0} : \mathrm{Nat}}$$

$$\frac{\Gamma \vdash \mathbf{t} : \mathrm{Nat}}{\Gamma \vdash \mathbf{S}(\mathbf{t}) : \mathrm{Nat}} \qquad \frac{\Gamma \vdash \mathbf{t} : S \qquad \Gamma \vdash \mathbf{u} : \mathrm{Nat} \to S \to S \qquad \Gamma \vdash \mathbf{v} : \mathrm{Nat}}{\Gamma \vdash \mathbf{rec}_{\mathrm{Nat}}(\mathbf{t}, \mathbf{u}, \mathbf{v}) : S}$$

$$\frac{}{\Gamma \vdash \mathbf{tt} : \mathrm{Bool}} \qquad \frac{}{\Gamma \vdash \mathbf{ff} : \mathrm{Bool}} \qquad \frac{\Gamma \vdash \mathbf{t} : S \qquad \Gamma \vdash \mathbf{u} : S \qquad \Gamma \vdash \mathbf{v} : \mathrm{Bool}}{\Gamma \vdash \mathbf{rec}_{\mathrm{Bool}}(\mathbf{t}, \mathbf{u}, \mathbf{v}) : S}$$

$$\frac{\Gamma \vdash \mathbf{t} : T \qquad \Gamma \vdash \mathbf{u} : S \to \mathrm{List}(S) \to T \to T \qquad \Gamma \vdash \mathbf{v} : \mathrm{List}(S)}{\Gamma \vdash \mathbf{rec}_{\mathrm{List}(S)}(\mathbf{t}, \mathbf{u}, \mathbf{v}) : T} \qquad \frac{}{\Gamma \vdash [\,] : \mathrm{List}(S)}$$

$$\frac{\Gamma \vdash \mathbf{t} : S \qquad \Gamma \vdash \mathbf{u} : \mathrm{List}(S)}{\Gamma \vdash \mathbf{t} :: \mathbf{u} : \mathrm{List}(S)} \qquad \frac{\Gamma \vdash \varphi : \mathrm{Prop} \qquad \Gamma \vdash \psi : \mathrm{Prop}}{\Gamma \vdash \varphi \to \psi : \mathrm{Prop}} \qquad \frac{\Gamma, \mathbf{x} : S \vdash \varphi : \mathrm{Prop}}{\Gamma \vdash \forall \mathbf{x}^S, \varphi : \mathrm{Prop}}$$

$$\frac{\Gamma \vdash \varphi : \mathrm{Prop} \qquad \Gamma \vdash \psi : \mathrm{Prop}}{\Gamma \vdash \varphi \wedge \psi : \mathrm{Prop}} \qquad \frac{\Gamma, \mathbf{x} : S \vdash \varphi : \mathrm{Prop}}{\Gamma \vdash \exists \mathbf{x}^S, \varphi : \mathrm{Prop}} \qquad \frac{\Gamma \vdash \mathbf{t} : S}{\Gamma \vdash S(\mathbf{t}) : \mathrm{Prop}}$$

**Figure 1** Well-typed HOL terms.

the quantifications $\forall$ and $\exists$. We postpone the introduction of the exact interpretation of Prop, and parametrize the model by a Heyting prealgebra $H$ equipped with two functions $\bigcap, \bigcup : \mathscr{P}(H) \to H$ representing unions and intersections. Finally, to interpret Prop, we need an interpretation of $S(\mathbf{t})$, for which we add an abstract encoding function $\ulcorner - \urcorner$ giving for each interpretation of a term an object of $H$. The other sorts are given their natural set theoretic semantics.

▶ **Definition 4** (Semantics). *Let $(H, \leq)$ be a Heyting pre-algebra with an intersection and a union operation $\bigcup, \bigcap : \mathscr{P}(H) \to H$. For each sort $S \in \mathfrak{S}$, we associate a set $[\![S]\!]$ (which we will also write $\mathbb{S}$):*

- $[\![\mathrm{Nat}]\!] \triangleq \mathbb{N}$
- $[\![\mathrm{Bool}]\!] \triangleq \mathbb{B}$
- $[\![\mathrm{List}(S)]\!] \triangleq [\![S]\!]^\star$
- $[\![S \times T]\!] \triangleq [\![S]\!] \times_{\mathbf{Set}} [\![T]\!]$
- $[\![S \to T]\!] \triangleq [\![T]\!]^{[\![S]\!]}$
- $[\![\mathrm{Prop}]\!] \triangleq H$

*Suppose given a function associating to each object a set of codes (possibly empty):*

$$\begin{array}{rccc} \ulcorner - \urcorner & : & \bigcup_{S \in \mathfrak{S}} [\![S]\!] & \longrightarrow & H \\ & & s & \longmapsto & \ulcorner s \urcorner \end{array}$$

*Let $\mathbf{rec}_{\mathbb{N}}$ (resp. $\mathbf{rec}_{\mathbb{B}}$, resp. $\mathbf{rec}_{\mathbb{L}}$) be the recursors given by the initial algebra structure of $\mathbb{N}$ (resp. $\mathbb{B}$, resp. $[\![S]\!]^\star$); $\frown$ the operation which, given a word $u$ and a letter $a$, appends $a$ to $u$ to make the new word $ua$; $\Longrightarrow_H$ the implication of $H$ and $\wedge_H$ its meet. Given a well-typed term $\Gamma \vdash \mathbf{t} : S$, we map it to its semantics $[\![\Gamma \vdash \mathbf{t} : S]\!] : \prod_{T \in \Gamma} [\![T]\!] \longrightarrow [\![S]\!]$ by the usual set-theoretic interpretation:*

- $[\![\mathbf{x}_1 : S_1, \cdots, \mathbf{x}_n : S_n \vdash \mathbf{x}_i : S_i]\!] \triangleq \pi_i^n$
- $[\![\Gamma \vdash \boldsymbol{\lambda}\mathbf{x}.\mathbf{t} : S \to T]\!] \triangleq (x \in [\![S]\!]) \mapsto (\overrightarrow{x} \in [\![\Gamma]\!]) \mapsto [\![\Gamma, \mathbf{x} : S \vdash \mathbf{t} : T]\!](\overrightarrow{x}, x)$

- $[\![\Gamma \vdash \mathbf{t}(\mathbf{u}) : T]\!] \triangleq \mathrm{eval} \circ \langle [\![\Gamma \vdash \mathbf{t} : S \to T]\!], [\![\Gamma \vdash \mathbf{u} : S]\!] \rangle$
- $[\![\Gamma \vdash \langle \mathbf{t}, \mathbf{u} \rangle : S \times T]\!] \triangleq \langle [\![\Gamma \vdash \mathbf{t} : S]\!], [\![\Gamma \vdash \mathbf{u} : T]\!] \rangle$
- $[\![\Gamma \vdash \boldsymbol{\pi}_1(\mathbf{t}) : S]\!] \triangleq \pi_1 \circ [\![\Gamma \vdash \mathbf{t} : S \times T]\!]$
- $[\![\Gamma \vdash \boldsymbol{\pi}_2(\mathbf{t}) : T]\!] \triangleq \pi_2 \circ [\![\Gamma \vdash \mathbf{t} : S \times T]\!]$
- $[\![\Gamma \vdash \mathbf{0} : \mathrm{Nat}]\!] \triangleq (\overrightarrow{x} \in [\![\Gamma]\!]) \mapsto 0$
- $[\![\Gamma \vdash S(\mathbf{t}) : \mathrm{Nat}]\!] \triangleq (n \mapsto n+1) \circ [\![\Gamma \vdash \mathbf{t} : \mathrm{Nat}]\!]$
- $[\![\Gamma \vdash \boldsymbol{rec}_{\mathrm{Nat}}(\mathbf{t}, \mathbf{u}, \mathbf{v}) : S]\!] \triangleq \mathrm{rec}_{\mathbb{N}} \circ \langle [\![\Gamma \vdash \mathbf{t} : S]\!], [\![\Gamma \vdash \mathbf{u} : \mathrm{Nat} \to S \to S]\!], [\![\Gamma \vdash \mathbf{v} : \mathrm{Nat}]\!] \rangle$
- $[\![\Gamma \vdash \boldsymbol{tt} : \mathrm{Bool}]\!] \triangleq (\overrightarrow{x} \in [\![\Gamma]\!]) \mapsto \top$
- $[\![\Gamma \vdash \boldsymbol{ff} : \mathrm{Bool}]\!] \triangleq (\overrightarrow{x} \in [\![\Gamma]\!]) \mapsto \bot$
- $[\![\Gamma \vdash \boldsymbol{rec}_{\mathrm{Bool}}(\mathbf{t}, \mathbf{u}, \mathbf{v}) : S]\!] \triangleq \mathrm{rec}_{\mathbb{B}} \circ \langle [\![\Gamma \vdash \mathbf{t} : S]\!], [\![\Gamma \vdash \mathbf{u} : S]\!], [\![\Gamma \vdash \mathbf{v} : \mathrm{Bool}]\!] \rangle$
- $[\![\Gamma \vdash [\,] : \mathrm{List}(S)]\!] \triangleq (\overrightarrow{x} \in [\![\Gamma]\!]) \mapsto \varepsilon$
- $[\![\Gamma \vdash \mathbf{t} :: \mathbf{u} : \mathrm{List}(S)]\!] \triangleq^{\frown} \circ \langle [\![\Gamma \vdash \mathbf{u} : \mathrm{List}(S)]\!], [\![\Gamma \vdash \mathbf{t} : S]\!] \rangle$
- $[\![\Gamma \vdash \boldsymbol{rec}_{\mathrm{List}(S)}(\mathbf{t}, \mathbf{u}, \mathbf{v}) : T]\!] \triangleq \mathrm{rec}_{\mathbb{L}} \circ \langle [\![\Gamma \vdash \mathbf{t} : T]\!], [\![\Gamma \vdash \mathbf{u} : S \to \mathrm{List}(S) \to T \to T]\!], [\![\Gamma \vdash \mathbf{v} : \mathrm{List}(S)]\!] \rangle$
- $[\![\Gamma \vdash \varphi \to \psi : \mathrm{Prop}]\!] \triangleq \Longrightarrow_H \circ \langle [\![\Gamma \vdash \varphi : \mathrm{Prop}]\!], [\![\Gamma \vdash \psi : \mathrm{Prop}]\!] \rangle$
- $[\![\Gamma \vdash \varphi \wedge \psi : \mathrm{Prop}]\!] \triangleq \wedge_H \circ \langle [\![\Gamma \vdash \varphi : \mathrm{Prop}]\!], [\![\Gamma \vdash \psi : \mathrm{Prop}]\!] \rangle$
- $[\![\Gamma \vdash \forall \mathbf{x}^S, \varphi : \mathrm{Prop}]\!] \triangleq (\overrightarrow{x} \in [\![\Gamma]\!]) \mapsto \bigcap_{s \in [\![S]\!]} [\![\Gamma, \mathbf{x} : S \vdash \varphi : \mathrm{Prop}]\!](\overrightarrow{x}, s)$
- $[\![\Gamma \vdash \exists \mathbf{x}^S, \varphi : \mathrm{Prop}]\!] \triangleq (\overrightarrow{x} \in [\![\Gamma]\!]) \mapsto \bigcup_{s \in [\![S]\!]} [\![\Gamma, \mathbf{x} : S \vdash \varphi : \mathrm{Prop}]\!](\overrightarrow{x}, s)$
- $[\![\Gamma \vdash S(\mathbf{t}) : \mathrm{Prop}]\!] \triangleq \ulcorner - \urcorner \circ [\![\Gamma \vdash \mathbf{t} : S]\!]$

In all generality, the syntactic terms are given in a typing context, like $\Gamma \vdash \mathbf{t} : S$. Thus, to construct an element of $\mathbb{S}$, we need an environment $\rho$ which covers $\Gamma$.

▶ **Notation 5.** *Given a HOL-context* $\Gamma$ *and a function* $\rho : \mathcal{X} \rightharpoonup \bigcup_{S \in \mathfrak{S}} \mathbb{S}$*, we define*

$$\rho \models \Gamma \triangleq \forall (\mathbf{x} : S) \in \Gamma, \rho(\mathbf{x}) \in \mathbb{S}$$

*If* $\Gamma \vdash \mathbf{t} : S$ *and* $\rho \models \Gamma$*, then we write* $[\![\mathbf{t}]\!](\rho)$ *for the canonical element in* $\mathbb{S}$ *associated to this term.*

## 3 The realizability model

From a naive realizability model, the main issue to realize FT is that a barred tree can be very ill-behaved. For example, a tree can have arbitrary long finite paths, as long as any infinite path is non computable. The solution we adopt is to construct a realizability relation accounting for possibly non-computable infinite paths: using a semantics similar to Kripke semantics, the trees are forced to be compatible with paths yet to be defined. Then, we add infinite paths in the computational model by fixing primitives $\xi$ together with a reduction rule $\xi\ \overline{n} \mapsto \overline{\alpha(n)}$, where $\alpha$ is the infinite path we wish to add in the model.

Using this realizability notion, the algorithm realizing FT is an extensive search among the finite binary words using the bar $b$ given as parameter. For each natural number $n$, the algorithm tests whether $n$ is a uniform bound for the predicate $B$. Thus, two cases can appear:

- the algorithm can stop at some $n$. In this case, $n$ is a uniform bound, so the algorithm indeed realizes a uniform bound by returning $n$

- the algorithm can go on indefinitely. This means that for any $n$, not every word of size $n$ satisfies the predicate, hence that there is a word $w_n \in \mathbb{B}^n \setminus B$ for any $n$. In the meta-theory, we can then extract an infinite path $\alpha \notin B$ using WKL: as $B$ is taken as the complementary of a tree, the words not in $B$ constitute a binary tree. This is the point where the oracles come into play: we can now use an oracle $\xi$ to simulate $\alpha$, and the Kripke semantics allows $b$ to take as argument the term $\xi$. Thus, we wish our Kripke-like semantics to be defined such that $b$ realizing a bar on $B$ not only means that $b$ computes as a bar for some given oracles, but also that it does so when new oracles are added. Finally, using a continuity argument, computing $b(\xi)$ can be done only for a finite part of $\xi$, which was already treated in the computation of the original algorithm, giving that both $B(\xi)$ holds (because it extends the prefix given by a bar on $B$) and that $B(\xi)$ does not hold (by our choice of $\alpha$). This rules out this case, which shows that the algorithm necessarily terminates.

To build a realizability model replicating this argument, we thus need to implement oracles but also to structure the realizability relation in a Kripke-like way: realizing $\varphi \to \psi$ not only means that realizers of $\varphi$ are sent to realizers of $\psi$ for some given world (taken here to be context of functions working as oracles) but also for future worlds (by considering extensions of this context).

## 3.1 Lambda-calculus

In Lubarsky and Rathjen's model [35], the argument mentioned above is implemented by taking a Kripke model of IZF. In this model, a realizability model is constructed based on Turing machines with oracle. The Kripke model itself uses as frame the set $\omega^{<\omega}$, and each world $\sigma$ corresponds to a finite set of oracles. Through a well-designed construction, Lubarsky and Rathjen manage to construct the model so that the realizability model taken at some world $\sigma$ are Turing machine computing with oracles in $\sigma$.

In comparison, our model relies on the $\lambda$-calculus, mainly to allow more explicit constructions (*e.g.* real concrete terms can be given and proved to realize some given sentences).

▶ **Definition 6** (Lambda-terms). *We fix an enumerable set $\mathcal{X}_\Lambda$ of $\lambda$-variables which will be written $x, y, \dots$ and define inductively the set $\Lambda$ of lambda-terms by:*

$$t, u ::= x \mid \lambda x.t \mid t\ u \mid \langle t, u\rangle \mid \pi_1\ t \mid \pi_2\ t \mid 0 \mid S\ t \mid \mathrm{rec}_{\mathrm{Nat}}\ t\ u\ v \mid \mathrm{tt} \mid \mathrm{ff} \mid \mathrm{rec}_{\mathrm{Bool}}\ t\ u\ v$$
$$\mid [\,] \mid t :: u \mid \mathrm{rec}_{\mathrm{List}}\ t\ u\ v \mid \xi_n$$

*where $n$ ranges over $\mathbb{N}$. We assume the usual convention of $\alpha$-renaming of bound variables.*

Using our multiple worlds analogy, we parametrize the reduction on terms by a list of functions $\mathbb{N} \to \mathbb{B}$. In place of the frame $\omega^{<\omega}$ previously mentioned, our set of worlds is $(\mathbb{N} \to \mathbb{B})^{<\omega}$, but every definition could be restricted to a subset of $\mathbb{N} \to \mathbb{B}$, for example by taking only functions of some given degree. The model of [35] can be obtained by taking the following restriction: fix a family $\{f_i\}_{i\in\mathbb{N}}$ of functions $\mathbb{N} \to \mathbb{B}$ of strictly increasing Turing degrees, and only consider the contexts $\sigma$ where $\sigma_i$ agrees with $f_i$ on all but finitely many values.

▶ **Definition 7** (Reduction context). *Let $\mathcal{F}_\mathbb{B}$ be the set of set-functions $\mathbb{N} \to \mathbb{B}$. We denote by $\mathcal{O}$ the set of lists of set-functions $\mathbb{N} \to \mathbb{B}$: $\mathcal{O} \triangleq \mathrm{List}(\mathcal{F}_\mathbb{B})$. Elements of this set will be called reduction contexts. For any $\sigma \in \mathcal{O}$, the $i$-th function of the list will be written $\sigma[i]$ and the length of $\sigma$ will be written $|\sigma|$.*

We assume the usual convention for substitution without capture of free variables.

▶ **Notation 8.** *Given a partial map $\rho : \mathcal{X}_\Lambda \rightharpoonup \Lambda$ and a term $t$, the simultaneous substitution of $t$ by $\rho$ is denoted $\rho(t)$. If $\rho$ is the function defined on the only point $x \in \mathcal{X}_\Lambda$ by $x \mapsto u$, then we will write $t[u/x]$. If $\rho$ is $x_1 \mapsto t_1, x_2 \mapsto t_2, \cdots, x_n \mapsto t_n$ then we write $t[t_i/x_i]$.*

The reduction itself is the expected $\beta$-reduction, enriched by the reduction, for any set-function $f : \mathbb{N} \to \mathbb{B}$ in the $i$-th place of the reduction context: $\xi_i\ \overline{n} \mapsto \overline{f(n)}$ where $\overline{n}$ means $S^n\ 0$, the canonical encoding of integers in this $\lambda$-calculus, and the encoding of booleans is $\mathtt{tt}$ and $\mathtt{ff}$, corresponding respectively to $\top \in \mathbb{B}$ and $\bot \in \mathbb{B}$.

▶ **Definition 9** (Reduction). *Let $\sigma \in \mathcal{O}$. We define the relation $\mapsto_\sigma$ of immediate reduction by the following rules:*

$$
\begin{array}{lll|lll}
(\lambda x.t)u & \mapsto_\sigma & t[u/x] & \mathrm{rec}_{\mathrm{Bool}}\ t\ u\ \mathtt{tt} & \mapsto_\sigma & t \\
\pi_i\langle t_1, t_2\rangle & \mapsto_\sigma & t_i \quad \forall i \in \{1,2\} & \mathrm{rec}_{\mathrm{Bool}}\ t\ u\ \mathtt{ff} & \mapsto_\sigma & u \\
\xi_i\ \overline{n} & \mapsto_\sigma & \overline{\sigma[i](n)} \quad \forall i < |\sigma| & \mathrm{rec}_{\mathrm{List}}\ t\ u\ [\,] & \mapsto_\sigma & t \\
\mathrm{rec}_{\mathrm{Nat}}\ t\ u\ 0 & \mapsto_\sigma & t & \mathrm{rec}_{\mathrm{List}}\ t\ u\ (v :: w) & \mapsto_\sigma & u\ v\ w\ (\mathrm{rec}_{\mathrm{List}}\ t\ u\ w) \\
\mathrm{rec}_{\mathrm{Nat}}\ t\ u\ (S\ v) & \mapsto_\sigma & u\ v\ (\mathrm{rec}_{\mathrm{Nat}}\ t\ u\ v) & & &
\end{array}
$$

*The reduction relation $\rhd_\sigma$ is the smallest relation containing $\mapsto_\sigma$ and which is compatible, meaning that it is closed by subterms. We write $\rhd_\sigma^\star$ for the reflexive and transitive closure of the relation $\rhd_\sigma$.*

As the reduction depends on the reduction context, we verify that extending the reduction context also extends the reduction.

▶ **Notation 10.** *For $\sigma, \tau \in \mathcal{O}$, we write $\sigma \preccurlyeq \tau$ when $\sigma$ is a prefix of $\tau$.*

▶ **Proposition 11.** *If $\sigma, \tau \in \mathcal{O}$ are such that $\sigma \preccurlyeq \tau$, then $\rhd_\sigma \subseteq \rhd_\tau$.*

**Proof.** It suffices to show that $\mapsto_\sigma \subseteq \mapsto_\tau$, which is straightforward by comparing the list of redexes of both reductions. ◀

## 3.2 Saturated sets

Propositions in realizability models are interpreted via the set of their realizers. Following the Brouwer-Heyting-Kolmogorov interpretation, realizers of a formula are expected to *compute* accordingly to the formula they realize. In particular, the interpretation of propositions is usually made in terms of *saturated* set of terms, *i.e.* closed under anti-reduction. In this setting, we therefore parametrize the notion of saturation by a reduction context.

The notion of truth value in our model is thus, in a world $\sigma$, the set of saturated sets for the relation $\mapsto_\sigma$. This truth value is local, in the sense that it only speaks about $\sigma$, so the adapted notion of truth value (for the Kripke interpretation) is given by family of saturated sets, for each $\sigma$.

▶ **Definition 12** (Saturated set). *Let $\sigma \in \mathcal{O}$ and $A \subseteq \Lambda$. We call $A$ a $\sigma$-saturated set if it is closed by anti-reduction for $\rhd_\sigma$, and write $\mathbf{SAT}_\sigma$ the set of $\sigma$-saturated sets:*

$$A \in \mathbf{SAT}_\sigma \triangleq \forall t, u \in \Lambda, t \rhd_\sigma u \implies u \in A \implies t \in A$$

*We define the set of global saturated subsets as follows:*

$$\mathbf{SAT} \triangleq \left\{ (A_\sigma) \in \prod_{\sigma \in \mathcal{O}} \mathbf{SAT}_\sigma \,\middle|\, \forall \sigma, \tau \in \mathcal{O}, \sigma \preccurlyeq \tau \implies A_\sigma \subseteq A_\tau \right\}$$

As with ordinary saturated sets, the different notions of saturated sets define complete lattices.

▶ **Proposition 13.** *For any $\sigma \in \mathcal{O}$, $\mathbf{SAT}_\sigma$ is a complete sub-lattice of $\mathscr{P}(\Lambda)$. $\mathbf{SAT}$ is also a complete lattice for the pointwise inclusion and with the pointwise union and intersection.*

The fact that $\mathbf{SAT}$ is taken to be the set of truth values means that we will use it to interpret Prop in the semantic interpretation of our HOL syntax. We thus show that this set can be equipped with a Heyting pre-algebra structure.

▶ **Proposition 14.** *Let $(A_\sigma), (B_\sigma) \in \mathbf{SAT}$. Let then*

$$(A_\sigma) \subseteq_{\mathbf{SAT}} (B_\sigma) \triangleq \exists t \in \Lambda, \forall \sigma \in \mathcal{O}, \forall u \in A_\sigma, tu \in B_\sigma$$
$$(A_\sigma) \Longrightarrow_{\mathbf{SAT}} (B_\sigma) \triangleq \sigma \mapsto \{t \in \Lambda \mid \forall \tau \succcurlyeq \sigma, \forall u \in A_\tau, tu \in B_\tau\}$$
$$(A_\sigma) \wedge_{\mathbf{SAT}} (B_\sigma) \triangleq \sigma \mapsto \{t \in \Lambda \mid (\pi_1\ t \in A_\sigma) \wedge (\pi_2\ t \in B_\sigma)\}$$

*This defines a Heyting pre-algebra.*

**Proof.** The fact that the operations are well defined is standard and uses the compatibility of the relation.

We show that $\subseteq_{\mathbf{SAT}}$ and $\wedge_{\mathbf{SAT}}$ are adjoints, meaning the following:

$$(A_\sigma) \subseteq_{\mathbf{SAT}} (B_\sigma) \Longrightarrow_{\mathbf{SAT}} (C_\sigma) \iff (A_\sigma) \wedge_{\mathbf{SAT}} (B_\sigma) \subseteq_{\mathbf{SAT}} (C_\sigma)$$

- Suppose there is $t$ such that for any $\sigma, \tau \in \mathcal{O}$, $u \in A_\sigma, v \in B_\tau$ and such that $\sigma \preccurlyeq \tau$, then $t\ u\ v \in C_\tau$. Then, for $\sigma \in \mathcal{O}$ and $u \in A_\sigma \wedge_{\mathbf{SAT}} B_\sigma$, it holds that $\pi_1\ u \in A_\sigma$ and $\pi_2\ u \in B_\sigma$, so $t\ (\pi_1\ u)\ (\pi_2\ u) \in C_\sigma$. By antireduction, $\lambda x.t\ (\pi_1\ x)\ (\pi_2\ x)$ thus witnesses that $(A_\sigma) \wedge_{\mathbf{SAT}} (B_\sigma) \subseteq_{\mathbf{SAT}} (C_\sigma)$.
- Suppose there exists $t$ such that for any $\sigma \in \mathcal{O}$ and $u \in A_\sigma \wedge_{\mathbf{SAT}} B_\sigma$, $tu \in C_\sigma$. Then, let $\sigma, \tau \in \mathcal{O}$ be such that $\sigma \preccurlyeq \tau$ and $u \in A_\sigma$. By inclusion, $u \in A_\tau$, so for any $v \in B_\tau$, we have $t\ \langle u, v\rangle \in C_\tau$. By antireduction, we therefore conclude that $\lambda x\ y.t\ \langle x, y\rangle$ is a witness that $(A_\sigma) \subseteq_{\mathbf{SAT}} (B_\sigma) \Longrightarrow_{\mathbf{SAT}} (C_\sigma)$.

Hence $\mathbf{SAT}_\sigma$ and $\mathbf{SAT}$ are Heyting pre-algebras. ◀

▶ Remark 15. The preorder $\subseteq_{\mathbf{SAT}_\sigma}$ contains $\subseteq$ with the witness $t = \lambda x.x$. For this reason, the operator $\bigcap$ defines a lower bound, even though not the greatest.

## 3.3 Realizability relation

The realizability relation relates the $\lambda$-calculus and the HOL model. The first step to relate the two is to instantiate the semantic defined previously with a Heyting prealgebra interpreting Prop. As is expected, this instantiation is done by taking $[\![\mathrm{Prop}]\!]$ to be the set of saturated sets. Be it $\mathbf{SAT}_\sigma$ or $\mathbf{SAT}$, those sets are equipped with a Heyting prealgebra structure and both intersection and union operations.

The only thing left to be defined is the $\ulcorner - \urcorner$ operation.

▶ **Definition 16** (Code of an element)**.** *We define by induction, for any $S \in \mathfrak{S}$ and any element $s \in \mathbb{S}$, the set $\ulcorner s \urcorner \in \mathbf{SAT}$:*

- *if $S = \mathrm{Nat}$, then for $n \in \mathbb{N}$, $\ulcorner n \urcorner \triangleq \sigma \mapsto \{t \in \Lambda \mid t \rhd^\star_\sigma \overline{n}\}$;*
- *if $S = \mathrm{Bool}$, then $\ulcorner \top \urcorner \triangleq \sigma \mapsto \{t \in \Lambda \mid t \rhd^\star_\sigma \mathrm{tt}\}$ and $\ulcorner \bot \urcorner \triangleq \sigma \mapsto \{t \in \Lambda \mid t \rhd^\star_\sigma \mathrm{ff}\}$;*
- *if $S = \mathrm{List}(T)$, then $\ulcorner \varepsilon \urcorner \triangleq \sigma \mapsto \{t \in \Lambda \mid t \rhd^\star_\sigma [\,]\}$ and for all $x \in \mathbb{T}^\star$, $y \in \mathbb{T}$, $\ulcorner x \frown y \urcorner \triangleq \sigma \mapsto \{t \in \Lambda \mid \exists u \in \ulcorner x \urcorner^\sigma, \exists v \in \ulcorner y \urcorner^\sigma, t \rhd^\star_\sigma v :: u\}$;*

- *if* $S = T \times U$*, then for all* $x \in \mathbb{T}$*,* $y \in \mathbb{U}$*,* $\ulcorner (x,y) \urcorner \triangleq \ulcorner x \urcorner \wedge_{\mathbf{SAT}} \ulcorner y \urcorner$*;*
- *if* $S = T \to U$*, then for all* $f : \mathbb{T} \to \mathbb{U}$*,* $\ulcorner f \urcorner \triangleq \bigcap_{x \in \mathbb{T}} \ulcorner x \urcorner \Longrightarrow_{\mathbf{SAT}} \ulcorner f(x) \urcorner$*;*
- *if* $S = \mathrm{Prop}$*, then for all* $(A_\sigma) \in \mathbf{SAT}$*,* $\ulcorner (A_\sigma) \urcorner \triangleq (A_\sigma)$*.*

*For each* $\sigma \in \mathcal{O}$*, we define* $\ulcorner s \urcorner^\sigma$ *by* $\ulcorner s \urcorner^\sigma \triangleq (\ulcorner s \urcorner)_\sigma$*.*

The fact that $\ulcorner s \urcorner^\sigma \in \mathbf{SAT}_\sigma$ is a straightforward induction on the set of sorts. We postpone the proof that $\sigma \preccurlyeq \tau \implies \ulcorner s \urcorner^\sigma \subseteq \ulcorner s \urcorner^\tau$ to the Proposition 18, as it is a particular case of a proposition of the realizability interpretation.

We now have a semantic interpretation of our initial HOL syntax, where propositions are sets of terms. The realizability relation then defines, for a syntactic proposition $\vdash \varphi : \mathrm{Prop}$, an object $[\![\varphi]\!] \in [\![\mathrm{Prop}]\!]$. In a way, the relation is simply $t \in \ulcorner [\![\varphi]\!](\rho) \urcorner$ for $\Gamma \vdash \varphi : \mathrm{Prop}$ and $\rho \models \Gamma$, but we prefer to give an explicit definition of this relation.

▶ **Definition 17** (Realizability relation)**.** *Let* $\sigma \in \mathcal{O}$*,* $\Gamma \in \mathrm{Ctx}$*,* $\rho \models \Gamma$ *and* $\Gamma \vdash \varphi : \mathrm{Prop}$*. We define by induction the realizability relation:*

$$\begin{array}{lcl} t \Vdash^\rho_\sigma S(\mathbf{t}) & \triangleq & t \in \ulcorner [\![\mathbf{t}]\!](\rho) \urcorner^\sigma \\ t \Vdash^\rho_\sigma \varphi \to \psi & \triangleq & \forall \tau \succcurlyeq \sigma, u \Vdash^\rho_\tau \varphi \implies tu \Vdash^\rho_\tau \psi \\ t \Vdash^\rho_\sigma \varphi \wedge \psi & \triangleq & (\pi_1\, t \Vdash^\rho_\sigma \varphi) \wedge (\pi_2\, t \Vdash^\rho_\sigma \psi) \end{array} \qquad \begin{array}{lcl} t \Vdash^\rho_\sigma \forall \mathbf{x}^S, \varphi & \triangleq & \forall s \in \mathbb{S}, t \Vdash^{\rho[\mathbf{x} \mapsto s]}_\sigma \varphi \\ t \Vdash^\rho_\sigma \exists \mathbf{x}^S, \varphi & \triangleq & \exists s \in \mathbb{S}, t \Vdash^{\rho[\mathbf{x} \mapsto s]}_\sigma \varphi \end{array}$$

*If* $\rho = \varnothing$ *(in which case* $\Gamma = [\,]$*), then we only write* $t \Vdash_\sigma \varphi$*. Finally, we define the universal realizability relation by* $t \Vdash^\rho \varphi \triangleq \forall \sigma \in \mathcal{O}, t \Vdash^\rho_\sigma \varphi$*.*

The following proposition justifies that the definitions give saturated sets, in the sense that they are monotone with regard to the order on reduction contexts.

▶ **Proposition 18.** *Let* $\sigma, \tau \in \mathcal{O}$*,* $\sigma \preccurlyeq \tau$*,* $\Gamma \in \mathrm{Ctx}, \rho \models \Gamma, \Gamma \vdash \varphi : \mathrm{Prop}$*. Then, for every* $t$*, we have the following implication:* $t \Vdash^\rho_\sigma \varphi \implies t \Vdash^\rho_\tau \varphi$*.*

**Proof.** We prove this by induction on $\varphi$ (and thus, for the case where $\varphi = S(\mathbf{t})$, by induction on $S$):

- if $S = \mathrm{Nat}$, then $t \Vdash^\rho_\sigma \mathrm{Nat}(\mathbf{t})$ means that $t \rhd^\star_\sigma \overline{n}$, thus $t \rhd^\star_\tau \overline{n}$, so $t \Vdash^\rho_\tau \mathrm{Nat}(\mathbf{t})$.
- if $S \in \{\mathrm{Bool}, \mathrm{List}(T), T \times U, \mathrm{Prop}\}$, the argument is the same.
- if $S = T \to U$, then this is just a consequence of the case of implication and universal quantification.
- if $\varphi = \psi \to \chi$, then let $\tau' \succcurlyeq \tau$ and $u \Vdash^\rho_{\tau'} \psi$. By transitivity, $\sigma \preccurlyeq \tau'$, so by hypothesis on $t$, $tu \Vdash^\rho_{\tau'} \psi$.
- if $\varphi = \psi \wedge \chi$, then we can conclude directly by induction hypothesis.
- the cases $\forall, \exists$ are the same: the induction hypothesis goes through without issue.

Hence the monotonicity of realizability with regard to the order on reduction contexts. ◀

▶ **Notation 19.** *For a formula* $\varphi$*, we write*

$$[\![\varphi]\!]^\rho \triangleq \{t \in \Lambda \mid t \Vdash\!\Vdash^\rho \varphi\} \qquad\qquad [\![\varphi]\!] \triangleq \{t \in \Lambda \mid t \Vdash\!\Vdash \varphi\}$$

▶ **Notation 20.** *We define the relativized universal quantification of a proposition* $\varphi$ *as* $\forall \mathbf{x}^{\{S\}}, \varphi \triangleq \forall \mathbf{x}^S, S(\mathbf{x}) \to \varphi$*. Thus, when realizing a formula like* $\forall \mathbf{x}^{\{S\}}, \varphi$*, it suffices to suppose that there is an object* $s \in \mathbb{S}$ *with a code* $t$ *and to realize* $\varphi$ *using the data* $t$*.*

*The relativized existential quantification is defined by* $\exists \mathbf{x}^{\{S\}}, \varphi \triangleq \exists \mathbf{x}^S, S(\mathbf{x}) \wedge \varphi$*. This type is naturally equipped with both* $\pi_1 \Vdash\!\Vdash \varphi \wedge \psi \to \varphi$ *and* $\pi_2 \Vdash\!\Vdash \varphi \wedge \psi \to \psi$*. This means that, if* $t \Vdash^\rho_\sigma \exists \mathbf{x}^{\{S\}}, \varphi(\mathbf{x})$*, then there exists some* $s \in \mathbb{S}$ *such that* $\pi_1\, t$ *is a code of* $s$ *(for the reduction context* $\sigma$*) and* $\pi_2\, t \Vdash^{\rho[\mathbf{y} \mapsto s]}_\sigma \varphi(\mathbf{y})$ *with* $\mathbf{y}$ *a fresh variable.*

## 3.4 Adequacy

The main result to ensure the soundness of a realizability model is called the adequacy lemma. This lemma has several implications and is based on the idea that typed terms are realizers. If typing is seen as a syntactic and decidable specification on programs, then the adequacy means that programs satisfying the syntactic specification associated to a proposition $\varphi$ also satisfy the semantic specification associated to $\varphi$.

From a logical point of view, the Curry-Howard correspondence also tells us that these typing rules translate to inference rules. Indeed, formal proofs of a proposition $\varphi$, based on natural deduction, can be associated to $\lambda$-terms $t$ such that $\vdash t : \varphi$. Using this correspondence, the adequacy lemma also means that, for any two statements $\varphi, \psi$ such that $\varphi$ is realized and $\psi$ is logically entailed by $\varphi$, $\psi$ is also realized. This makes the set $\mathscr{T}_{\Vdash_\sigma}$ of realized propositions at some context $\sigma$, and the set $\mathscr{T}_{\Vdash}$ of uniformly realized propositions, into a logical theory, closed by deduction. This also means that the theory is consistent, as there are predicates with no realizers (hence non realized formulas, in particular $\bot$). The adequacy lemma is essential to show that our model separates FT from WKL, as this shows that there is a consistent theory (namely $\mathscr{T}_{\Vdash}$) containing FT and not containing WKL.

The proof of the adequacy lemma proceeds by induction on the typing relation. For the cases involving universal (resp. existential) quantification, an additional lemma is required to relate the realizability of $\varphi[\mathbf{t}/\mathbf{x}]$ and the realizability of $\varphi$ where $\rho$ is enriched by $\mathbf{x} \mapsto \rho(\mathbf{t})$.

▶ **Lemma 21.** *Let $\sigma \in \mathcal{O}$, $\Gamma \in \mathrm{Ctx}$, $\rho \models \Gamma$ and suppose that $\Gamma, \mathbf{x} : S \vdash \varphi : \mathrm{Prop}$ and $\Gamma \vdash \mathbf{t} : S$. The following equivalence holds: $t \Vdash^\rho_\sigma \varphi[\mathbf{t}/\mathbf{x}] \iff t \Vdash^{\rho[\mathbf{x}\mapsto\rho(\mathbf{t})]}_\sigma \varphi$.*

**Proof.** We must first prove by induction on terms $\mathbf{u}$ that $\rho(\mathbf{u}[\mathbf{t}/\mathbf{x}]) = \rho[\mathbf{x} \mapsto \rho(\mathbf{t})](\mathbf{u})$. Then, by induction on $\varphi$:

- if $\varphi = S(\mathbf{u})$, the equality on terms entails $t \in \ulcorner \rho(\mathbf{u}[\mathbf{t}/\mathbf{x}]) \urcorner^\sigma \iff t \in \ulcorner \rho[\mathbf{x} \mapsto \rho(\mathbf{t})](\mathbf{u}) \urcorner^\sigma$.
- the other four cases are straightforward by using the induction hypothesis to replace $\Vdash^\rho_\sigma \varphi[\mathbf{t}/\mathbf{x}]$ by $\Vdash^{\rho[\mathbf{x}\mapsto\rho(\mathbf{t})]}_\sigma \varphi$. ◀

We also introduce the typing system we consider for this model. It can be seen as rules for logical deductions in HOL and rules describing the behavior of Nat, Bool, List, *i.e.* constructor rules saying that standard elements are indeed in the expected sort, and induction principles.

▶ **Definition 22** (Typing system). *We define a ($\lambda$-)context as a list $\Xi \in \mathrm{List}(\mathcal{X}_\Lambda \times \mathrm{Prop})$, whose elements we will write $(x : \varphi)$.*

*The typing relation $\Gamma \mid \Xi \vdash t : \varphi$ is defined by induction by the rules in Figure 2.*

▶ **Theorem 23** (Adequacy). *Let $\sigma \in \mathcal{O}$, $\Gamma \in \mathrm{Ctx}$, $\rho \models \Gamma$, $\varphi_i, \varphi$ be formulas and $t_i, t$ be terms such that*

$$\begin{cases} \Gamma \vdash \varphi_i : \mathrm{Prop} & \forall i = 1, \dots, n \\ \Gamma \vdash \varphi : \mathrm{Prop} \end{cases} \quad \textit{and} \quad \begin{cases} t_i \Vdash^\rho_\sigma \varphi_i & \forall i = 1, \dots, n \\ \Gamma \mid x_1 : \varphi_1, \dots, x_n : \varphi_n \vdash t : \varphi \end{cases}$$

*then $t[t_i/x_i] \Vdash^\rho_\sigma \varphi$.*

**Proof.** The proof is by induction on the typing derivation:

- *Case* (Ax) *with* $t = x_i$. The result follows from the hypotheses.
- *Case* ($\to_{\mathrm{i}}$) *with* $\lambda x.t : \psi \to \chi$. Let $\tau \succcurlyeq \sigma$ and $u \Vdash^\rho_\tau \psi$, let us prove that $(\lambda x.t)u \Vdash^\rho_\tau \chi$. By hypothesis, $t[t_i/x_i][u/x] \Vdash^\rho_\tau \chi$, but then it suffices to use the fact that $[\![\chi]\!]^\rho_\tau$ is closed by antireduction.

$$\frac{(x:\varphi)\in\Xi}{\Gamma\mid\Xi\vdash x:\varphi}\,\mathrm{Ax} \qquad \frac{\Gamma\mid\Xi,x:\varphi\vdash t:\psi}{\Gamma\mid\Xi\vdash\lambda x.t:\varphi\to\psi}\,{\to_{\mathrm{i}}} \qquad \frac{\Gamma\mid\Xi\vdash t:\varphi\to\psi\qquad\Gamma\mid\Xi\vdash u:\varphi}{\Gamma\mid\Xi\vdash t\;u:\psi}\,{\to_{\mathrm{e}}}$$

$$\frac{\Gamma\mid\Xi\vdash t:\varphi\qquad\Gamma\mid\Xi\vdash u:\psi}{\Gamma\mid\Xi\vdash\langle t,u\rangle:\varphi\wedge\psi}\,{\wedge_{\mathrm{i}}} \qquad \frac{\Gamma\mid\Xi\vdash t:\varphi\wedge\psi}{\Gamma\mid\Xi\vdash\pi_1\;t:\varphi}\,{\wedge^1_{\mathrm{e}}} \qquad \frac{\Gamma\mid\Xi\vdash t:\varphi\wedge\psi}{\Gamma\mid\Xi\vdash\pi_2\;t:\psi}\,{\wedge^2_{\mathrm{e}}}$$

$$\frac{\Gamma,\mathbf{x}:S\mid\Xi\vdash t:\varphi}{\Gamma\mid\Xi\vdash t:\forall\mathbf{x}^S,\varphi}\,{\forall_{\mathrm{i}}} \qquad \frac{\Gamma\mid\Xi\vdash t:\forall\mathbf{x}^S,\varphi\qquad\Gamma\vdash\mathbf{t}:S}{\Gamma\mid\Xi\vdash t:\varphi[\mathbf{t}/\mathbf{x}]}\,{\forall_{\mathrm{e}}}$$

$$\frac{\Gamma\mid\Xi\vdash t:\varphi[\mathbf{t}/\mathbf{x}]\qquad\Gamma\vdash\mathbf{t}:S}{\Gamma\mid\Xi\vdash t:\exists\mathbf{x}^S,\varphi}\,{\exists_{\mathrm{i}}} \qquad \frac{\Gamma\mid\Xi\vdash t:\exists\mathbf{x}^S,\varphi\qquad\Gamma,\mathbf{x}:S\mid\Xi\vdash u:\varphi\to\psi}{\Gamma\mid\Xi\vdash u\;t:\psi}\,{\exists_{\mathrm{e}}}$$

$$\frac{}{\Gamma\mid\Xi\vdash 0:\mathrm{Nat}(\mathbf{0})}\,\mathrm{Nat}^0_{\mathrm{i}} \qquad \frac{\Gamma\mid\Xi\vdash t:\mathrm{Nat}(\mathbf{t})}{\Gamma\mid\Xi\vdash S\;t:\mathrm{Nat}(S(\mathbf{t}))}\,\mathrm{Nat}^S_{\mathrm{i}}$$

$$\frac{}{\Gamma\mid\Xi\vdash\mathrm{rec}_{\mathrm{Nat}}:\forall X^{\mathrm{Nat}\to\mathrm{Prop}},X(\mathbf{0})\to(\forall\mathbf{n}^{\{\mathrm{Nat}\}},X(\mathbf{n})\to X(\mathbf{S}(\mathbf{n})))\to\forall\mathbf{n}^{\{\mathrm{Nat}\}},X(\mathbf{n})}\,\mathrm{Nat}_{\mathrm{e}}$$

$$\frac{}{\Gamma\mid\Xi\vdash\mathrm{tt}:\mathrm{Bool}(\mathbf{tt})}\,\mathrm{Bool}^{\top}_{\mathrm{i}} \qquad \frac{}{\Gamma\mid\Xi\vdash\mathrm{ff}:\mathrm{Bool}(\mathbf{ff})}\,\mathrm{Bool}^{\bot}_{\mathrm{i}}$$

$$\frac{}{\Gamma\mid\Xi\vdash\mathrm{rec}_{\mathrm{Bool}}:\forall X^{\mathrm{Bool}\to\mathrm{Prop}},X(\mathbf{tt})\to X(\mathbf{ff})\to\forall\mathbf{x}^{\{\mathrm{Bool}\}},X(\mathbf{x})}\,\mathrm{Bool}_{\mathrm{e}}$$

$$\frac{}{\Gamma\mid\Xi\vdash[\,]:\mathrm{List}(S)([\,])}\,\mathrm{List}^{[]}_{\mathrm{i}} \qquad \frac{\Gamma\mid\Xi\vdash t:S(\mathbf{t})\qquad\Gamma\mid\Xi\vdash u:\mathrm{List}(S)(\mathbf{u})}{\Gamma\mid\Xi\vdash t::u:\mathrm{List}(S)(\mathbf{t}::\mathbf{u})}\,\mathrm{List}^{::}_{\mathrm{i}}$$

$$\frac{}{\Gamma\mid\Xi\vdash\mathrm{rec}_{\mathrm{List}}:\forall X^{\mathrm{List}(S)\to\mathrm{Prop}},X([\,])\to(\forall\mathbf{x}^{\{S\}},\forall\mathbf{y}^{\{\mathrm{List}(S)\}},X(\mathbf{y})\to X(\mathbf{x}::\mathbf{y}))\to\forall\mathbf{x}^{\{\mathrm{List}(S)\}},X(\mathbf{x})}\,\mathrm{List}_{\mathrm{e}}$$

**Figure 2** Typing rules.

- *Case* $(\to_{\mathrm{e}})$. This case is straightforward by definition of realizing $\psi\to\chi$.
- *Case* $(\wedge_{\mathrm{i}})$ *with* $\langle t,u\rangle:\psi\wedge\chi$. We know by induction hypothesis that $t[t_i/x_i]\Vdash^{\rho}_{\sigma}\psi$ and that $u[t_i/x_i]\Vdash^{\rho}_{\sigma}\chi$. Moreover, $\pi_1\;\langle t,u\rangle\rhd_\sigma t$ and $\pi_2\;\langle t,u\rangle\rhd_\sigma u$, so $\pi_1\;(\langle t,u\rangle[t_i/x_i])\Vdash^{\rho}_{\sigma}\psi$ and $\pi_2\;(\langle t,u\rangle[t_i/x_i])\Vdash^{\rho}_{\sigma}\chi$ by saturation, hence $\langle t,u\rangle[t_i/x_i]\Vdash^{\rho}_{\sigma}\psi\wedge\chi$.
- *Cases* $(\wedge_{\mathrm{e}})$. Both cases are straightforward by definition of realizing a conjunction.
- *Case* $(\forall_{\mathrm{i}})$ *with* $t:\forall\mathbf{x}^S,\psi$. Using the induction hypothesis, for any $\rho'\models\Gamma,\mathbf{x}:S$, we get that $t[t_i/x_i]\Vdash^{\rho'}_{\sigma}\psi$. For any $s\in\mathbb{S}$, $\rho[\mathbf{x}\mapsto s]\models\Gamma$, so $t[t_i/x_i]\Vdash^{\rho[\mathbf{x}\mapsto s]}_{\sigma}\psi$ which is, by definition, $t[t_i/x_i]\Vdash^{\rho}_{\sigma}\forall\mathbf{x}^S,\psi$.
- *Case* $(\forall_{\mathrm{e}})$ *with* $t:\psi[\mathbf{t}/\mathbf{x}]$. By definition of realizing $\forall\mathbf{x}^S,\psi$ and using the substitution lemma.
- *Case* $(\exists_{\mathrm{i}})$ *with* $t:\exists\mathbf{x}^S,\varphi$. By induction hypothesis, we know that $[\![\mathbf{t}]\!](\rho)\in\mathbb{S}$ and that $t\Vdash^{\rho[\mathbf{x}\mapsto[\![\mathbf{t}]\!](\rho)]}_{\sigma}\varphi$, so we indeed have some $s\in\mathbb{S}$ (namely $[\![\mathbf{t}]\!](\rho)$) such that $t\Vdash^{\rho[\mathbf{x}\mapsto s]}_{\sigma}\varphi$, which is the expected result.
- *Case* $(\exists_{\mathrm{e}})$ *with* $u\;t:\psi$. By induction hypothesis, we know that for any $\rho\models\Gamma,\mathbf{x}:S$, we have $u\Vdash^{\rho}_{\sigma}\varphi\to\psi$. In particular, by induction hypothesis there exists $s\in\mathbb{S}$ such that $t\Vdash^{\rho[\mathbf{x}\mapsto s]}_{\sigma}\varphi$, but then $\rho[\mathbf{x}\mapsto s]\models\Gamma,\mathbf{x}:S$ so we can apply $t$ to $u$ to find that $u\;t\Vdash^{\rho}_{\sigma}\psi$.
- The cases for Nat or Bool are analogous to the cases for List.
- *Case* $(\mathrm{List}^{[]}_{\mathrm{i}})$. Indeed, $[\,]\rhd^{\star}_{\sigma}[\,]$.
- *Case* $(\mathrm{List}^{::}_{\mathrm{i}})$. By induction hypothesis.

- *Case* ($\mathrm{List}_e$). Let $X$ be a predicate on $\mathrm{List}(S)$, $\tau \succcurlyeq \sigma$ and

$$u \Vdash^{\rho}_{\tau} X([\,]) \qquad v \Vdash^{\rho}_{\tau} \forall \mathbf{x}^{\{X\}}, \forall \mathbf{y}^{\{\mathrm{List}(S)\}}, X(\mathbf{y}) \to X(\mathbf{x} :: \mathbf{y}) \qquad w \Vdash^{\rho}_{\tau} \mathrm{List}(S)(\mathbf{x})$$

we want to prove that $t = \mathrm{rec}_{\mathrm{List}}\ u\ v\ w$ realizes $X(\mathbf{x})$.
The object $[\![\mathbf{x}]\!](\rho)$ is a list. By induction, it is either $[\,]$ or $s \frown s'$ for two elements $s \in \mathbb{S}^\star, s' \in \mathbb{S}$:
  - if $[\![\mathbf{x}]\!](\rho) = [\,]$, then $w \rhd^\star_\tau [\,]$, $\mathrm{rec}_{\mathrm{List}}\ u\ v\ w \rhd^\star_\tau u$ and $u \Vdash^\rho_\tau X([\,])$. In particular, we get that $\mathrm{rec}_{\mathrm{List}}\ u\ v\ w \Vdash^\rho_\tau X([\,])$.
  - if $[\![\mathbf{x}]\!](\rho) = s \frown s'$, then $w \rhd^\star_\tau w_1 :: w_2$ for $w_1 \in {}^{\ulcorner} s'^{\urcorner\tau}$ and $w_2 \in {}^{\ulcorner} s^{\urcorner\tau}$. In this case, by induction hypothesis, $\mathrm{rec}_{\mathrm{List}}\ u\ v\ w_2 \Vdash^{\rho[\mathbf{y}\mapsto s']}_\tau \mathrm{List}(S)(\mathbf{y})$, and

$$\mathrm{rec}_{\mathrm{List}}\ u\ v\ w \rhd^\star_\tau \mathrm{rec}_{\mathrm{List}}\ u\ v\ (w_1 :: w_2) \rhd^\star_\tau v\ w_2\ (\mathrm{rec}_{\mathrm{List}}\ u\ v\ w_2) \qquad \blacktriangleleft$$

As the adequacy is defined for any $\sigma \in \mathcal{O}$, it is straightforward that $\Vdash\!\!\vdash$ is also adequate.

▶ **Corollary 24.** *For any $\sigma \in \mathcal{O}$, the sets*

$$\mathscr{T}_{\Vdash_\sigma} \triangleq \{\varphi \in \mathrm{Prop} \mid \exists t \in \Lambda, t \Vdash_\sigma \varphi\} \quad \textit{and} \quad \mathscr{T}_{\Vdash\!\!\vdash} \triangleq \{\varphi \in \mathrm{Prop} \mid \exists t \in \Lambda, t \Vdash\!\!\vdash \varphi\}$$

*are closed by logical consequence. Moreover, as* **SAT** *contains $\varnothing$, the false proposition $\bot \triangleq \forall \varphi^{\mathrm{Prop}}, \varphi$ is not realized, so $\mathscr{T}_{\Vdash\!\!\vdash}$ (resp. $\mathscr{T}_{\Vdash_\sigma}$) is consistent.*

▶ **Notation 25.** *A lot of functions of our model are not described by any HOL-term but can have a realizer. Thus, we introduce the following notation for $t \in \Lambda, S \in \mathfrak{S}, s \in \mathbb{S}$:*

$$t \Vdash\!\!\vdash S(s) \quad \triangleq \quad t \Vdash\!\!\vdash^{[\mathbf{x}\mapsto s]} S(\mathbf{x})$$

*Given a $\lambda$-term $t$, there can be several objects $s, s' \in S$ such that $t$ is a code of $s$ (resp. $s'$). However, the intended behavior of $s$ (resp. $s'$) is given by the term $t$, and the choice of the object will not be relevant. For this reason, we will write*

$$t \Vdash\!\!\vdash S \quad \triangleq \quad \exists s \in \mathbb{S}, t \Vdash\!\!\vdash S(s)$$

*These notations can also be used with a reduction context, writing for instance $t \Vdash_\sigma S$ to denote that $\exists s \in \mathbb{S}, t \Vdash_\sigma S(s)$.*

## 4 A Realizer for the Fan Theorem

With our model constructed, and the proof that it describes a consistent theory done, we are now ready to prove that this model satisfies FT. We also show that WKL fails with a standard argument using a Kleene tree. We start by defining precisely these principles, before going through a comprehensive construction of the program that we use to realize FT.

### 4.1 Fan Theorem & Weak Kőnig's Lemma

Now that we know both the logic we use and the model we consider, let us give a formal definition of FT. We assume given the function $\alpha \mapsto \alpha\restriction_{\mathbf{n}}$, where $\alpha\restriction_{\mathbf{n}}$ is the prefix of the $n$ first values of $\alpha$, given in a list, and write $\preccurlyeq_{\mathrm{pref}}$ for the prefix order on lists.

As we mentioned before, FT applies to the complement of a tree. The way trees are formalized can vary, but in reverse mathematics and computability it is standard to consider a tree as a predicate $T \subseteq \mathrm{List}(\mathbb{N})$ (where List is used in place of the set of finite words) closed by prefix, meaning that $\ell \preccurlyeq_{\mathrm{pref}} \ell' \implies \ell' \in T \implies \ell \in T$. This means that the complement of a tree must satisfy the converse, which we call a *monotone* predicate:

$$\mathrm{Mono}^S(B) \triangleq \forall \ell^{\mathrm{List}(S)}\, \ell'^{\mathrm{List}(S)}, \ell \preccurlyeq_{\mathrm{pref}} \ell' \implies B(\ell) \implies B(\ell')$$

Naturally, when writing $\forall B^{\mathrm{Mono}^S}, \varphi$, we use the same relativization procedure as before.

The statement we use as FT (resp WKL) are

$$\mathrm{FT} \triangleq \forall B^{\mathrm{Mono}^{\mathrm{Bool}}}, (\forall \alpha^{\{\mathrm{Nat}\to\mathrm{Bool}\}}, \exists \mathbf{n}^{\{\mathrm{Nat}\}}, B(\alpha\restriction_{\mathbf{n}})) \Rightarrow \exists \mathbf{n}^{\{\mathrm{Nat}\}}, \forall \alpha^{\{\mathrm{Nat}\to\mathrm{Bool}\}}, B(\alpha\restriction_{\mathbf{n}})$$
$$\mathrm{WKL} \triangleq \forall T^{\mathrm{Tree}^{\mathrm{Bool}}}, (\forall \mathbf{n}^{\{\mathrm{Nat}\}}, \exists \ell^{\{\mathrm{List}(\mathrm{Bool})\}}, |\ell| = \mathbf{n} \wedge T(\ell)) \Rightarrow \exists \alpha^{\{\mathrm{Nat}\to\mathrm{Bool}\}} \forall \mathbf{n}^{\{\mathrm{Nat}\}}, T(\alpha\restriction_{\mathbf{n}})$$

FT is thus a rewriting of the contrapositive of WKL, replacing a tree by its complement, a monotone predicate. However, the statement of FT varies in the literature. In [35] on which we base our model, FT is given on binary trees the same way we do here. In [6], however, FT is stated with a finite set instead of Bool: in fact, our construction perfectly works in this setting because there is no more computational difficulty in the finite case, as long as the bound on the finite set is known *a priori*. Another statement, which would make FT dual to KL, would be to consider $T : \mathrm{List}(\mathbb{N}) \to \mathrm{Prop}$ (the complement of $B$, a tree) with a finite branching condition saying that $T \cap \mathbb{N}^n$ is always a finite set, in which case our models would not satisfy the principle.

## 4.2 First exploration of the model

Given our model, any closed term typable in system T can be written as a closed term and be realized by a term canonically associated to it. This means that, for example, the functions + or × can be expressed both as objects of the sort $\mathrm{Nat} \to \mathrm{Nat} \to \mathrm{Nat}$ and as realizers computing these functions.

For example, every boolean operation $\wedge, \vee, \neg$ can be written using $\mathrm{rec}_{\mathrm{Bool}}$. Similarly, functions on lists that can be defined by a fold function can be written inside our system both in HOL and as realizer. We thus assume given functions like the length $|\ell|$ of a list $\ell$.

In particular, for this reason, any sort on the grammar

$$S, T ::= \mathrm{Bool} \mid \mathrm{Nat} \mid \mathrm{List}(S) \mid S \times T$$

has decidable equality: they even have decidable order $\leq$ where

- if $b, b' : \mathrm{Bool}$, $b \leq b'$ means that $b$ is false or $b'$ is true
- if $n, m : \mathrm{Nat}$, $n \leq m$ is the natural inequality on integers
- if $\ell, \ell' : \mathrm{List}(S)$, $\ell \preccurlyeq_{\mathrm{pref}} \ell'$ is the prefix order
- if $\langle x, y\rangle, \langle x', y'\rangle : S \times T$, $\langle x, y\rangle \leq \langle x', y'\rangle$ is defined as $(x \leq x') \wedge (y \leq y')$

Saying they are decidable order means that there exist a term $\mathbf{t}_\leq$ together with a λ-term $t_\leq$ such that $t_\leq \Vdash (S \to S \to \mathrm{Bool})(\mathbf{t}_\leq)$ and $\mathbf{t}_\leq(\mathbf{x}, \mathbf{y}) \iff \mathbf{x} \leq \mathbf{y}$. We will call those the discrete sorts.

### 4.2.1 Bounded formulas

Using boolean operations, it is straightforward that formulas of propositional logic over the atomic formulas $t = u$ and $t \leq u$ are themselves decidables, as soon as $t$ and $u$ are typed in a sort $S$ which is discrete.

Bounded formulas are defined in a similar way as for $\Sigma^1_0$ formulas of arithmetic, but with only relativized quantification (we want to be able to access the data we quantify on). Using the recursors of Nat, Bool and List, as well as projections, any bounded formula implying only quantifications on discrete sorts and atomic formulas on discrete sorts (parameters can be functions) is decidable.

For a decidable formula $\varphi(\mathbf{x}^S)$, we write $\mathrm{eval}_\varphi$ for a term deciding $\varphi$. This is a term with witnesses that for $s \in \mathbb{S}$ and $t \in \ulcorner s \urcorner$, $\mathrm{eval}_\varphi\ t$ realizes either $\varphi(s)$ if it reduces to $\mathtt{tt}$, or $\neg\varphi(s)$ if it reduces to $\mathtt{ff}$.

### 4.2.2 Binary lists

However, for lists, the quantification $\forall \ell \preccurlyeq_{\mathrm{pref}} \ell'$ with parameter $\ell'$ is not the one we will use. We wish to quantify for lists of length below some fixed integer. To be able to define $\forall \ell, |\ell| \leq n \implies \varphi$, we define a function enum given by the binary decomposition of the number $n$. If $\overline{n}^2$ is the binary decomposition of $n$ in base 2 as a list of booleans, then let $\mathrm{enum}(n)$ be the list $\overline{n+1}^2$ in which we leave out the strongest bit 1. The function enum is a bijection from $\mathbb{N}$ to $\mathrm{List}(\mathbb{B})$ which goes by increasing size, allowing the use of the quantification $\forall \ell^{\mathrm{List(Bool)}}, |\ell| \leq n \implies \varphi(\ell)$ by the encoding $\forall m^{\mathrm{Nat}}, m \leq 2^n \implies \varphi(\mathrm{enum}(m))$.

### 4.2.3 Finite and infinite path

Given a function $\alpha : \mathrm{Nat} \to \mathrm{Bool}$ and a number $n$, we write $\alpha\restriction_n$: $\mathrm{List(Bool)}$ for the list of the $n$ first values of $\alpha$. Conversely, given a list $\ell : \mathrm{List(Bool)}$, we write $\ell 0^\infty$ for the function $\mathrm{Nat} \to \mathrm{Bool}$ which associates $\ell_i$, the $i$-th value of $\ell$, to $i < |\ell|$, and 0 to any $i \geq |\ell|$.

### 4.2.4 Predicate given a bar

We will now define the term realizing FT. Fix a monotone predicate $B : \mathrm{List(Bool)} \to \mathrm{Prop}$, *i.e.* such that $\forall \ell, \ell'^{\mathrm{List(Bool)}}, \ell \preccurlyeq_{\mathrm{pref}} \ell' \implies B(\ell) \implies B(\ell')$. A bar for this predicate is a realizer (in some reduction context $\sigma$) $b \Vdash_\sigma \forall \alpha^{\{\mathrm{Nat}\to\mathrm{Bool}\}}, \exists \mathbf{n}^{\{\mathrm{Nat}\}}, B(\alpha\restriction_{\mathbf{n}})$. Suppose given such a bar $b$.

By the definition of realizing $\exists \mathbf{n}^{\{\mathrm{Nat}\}}$, for any $\alpha \Vdash \mathrm{Nat} \to \mathrm{Bool}$, we can retrieve

$$\pi_1(b(\alpha)) \Vdash \mathrm{Nat}(\mathbf{n}) \quad \text{such that} \quad \pi_2(b(\alpha)) \Vdash B(\alpha\restriction_{\mathbf{n}})$$

To lighten the notations, we write $\alpha\restriction_b$ for $\alpha\restriction_{\pi_1(b(\alpha))}$, which is thus a realizer of some list $\ell \in \mathrm{List(Bool)}$ coming with a realizer of $B(\alpha\restriction_b)$.

Remark that if $\alpha\restriction_b \preccurlyeq_{\mathrm{pref}} \ell$, then by the realizer of the monotonicity of $B$ and the fact that $B(\alpha\restriction_b)$, we can deduce (in the realizability sense) that $B(\ell)$ holds.

Our goal is to construct a monotone subset $C \subseteq B$ which is decidable. For this, it suffices to find a bounded formula such that any $\ell$ satisfying this formula is in $B$. From the previous remark, we only need to show that if $C(\ell)$, then there is a prefix of $\ell$ of the form $\alpha\restriction_b$.

We then use this with a bounded quantification: we enumerate every list $\ell$ and, each time, add to $C$ the list $\ell 0^\infty\restriction_b$ and every extension, as long as those extensions are of size greater than $|\ell|$. This can also be rewritten as follows: given a list $\ell$, checking whether $\ell \in C$ amounts to checking whether there exists a list $\ell'$ of size lower than $|\ell|$ such that $\ell' 0^\infty\restriction_b \preccurlyeq_{\mathrm{pref}} \ell$.

This automatically gives a term

$$C_b(\ell) \triangleq \mathrm{eval}_{\exists \ell'^{\{\mathrm{List(Bool)}\}}, |\ell'|<|\ell| \wedge \ell' 0^\infty\restriction_b \preccurlyeq_{\mathrm{pref}} \ell}\ \ell$$

Now, we define the following predicate

$$C'_b(n) \triangleq \mathrm{eval}_{\forall \ell^{\{\mathrm{List(Bool)}\}}, |\ell|=n \implies C(\ell)}\ n$$

$C'$ is decidable as $C$ is and the quantification is bounded. If $C'(n)$ is realized, then $n$ is a uniform bound for $C$, hence for $B$. Thus, the term we construct is an unbounded search to find a $n$ satisfying $C'_b(n)$. Let $Y \triangleq \lambda f.(\lambda x.f\ (x\ x))(\lambda x.f\ (x\ x))$, we define

$$n_{C_b} \triangleq Y(\lambda f.\lambda n.\mathrm{rec}_{\mathrm{Bool}}\ n\ (f\ (S\ n))\ C'_b(n))$$

▶ Remark 26. By computing reduction, we see that $n_{C_b}\, n \rhd^\star n$ if $C'_b(n)$ holds, and otherwise $n_{C_b}\, n \rhd^\star n_{C_b}\, (n+1)$. Hence, either $n_{C_b}\, 0$ runs infinitely many steps, and for every $n$, $C'_b(n)$ is false (meaning that there exists $\ell_n \notin C$ of size $n$) or $n_{C_b} 0 \Vdash \mathrm{Nat}$ and it is a uniform bound.

## 4.3 Fan Theorem is realized

Before proving that FT is realized, a continuity lemma is needed.

▶ **Lemma 27** (Continuity). *Let $\sigma \in \mathcal{O}$ be a reduction context and $t \in \Lambda$ be a term such that*

$$t \Vdash_\sigma (\mathrm{Nat} \to \mathrm{Nat}) \to \mathrm{Nat}$$

*Then for any $\alpha \Vdash_\sigma \mathrm{Nat} \to \mathrm{Nat}$, there exists $m \in \mathbb{N}$ such that*

$$\forall (\beta \Vdash_\sigma \mathrm{Nat} \to \mathrm{Nat}), (\forall i < m, \beta\ \overline{i} =_\sigma \alpha\ \overline{i}) \implies t\ \beta =_\sigma t\ \alpha$$

**Proof.** By hypothesis, we know that $t\ \alpha \rhd^\star_\sigma \overline{n}$ for some $n \in \mathbb{N}$. By confluence, it is possible to chose any reduction strategy which is normalizing. In particular, we can choose the leftmost reduction strategy: $t\ \alpha \underset{l}{\rhd^\star_\sigma} \overline{n}$. Without loss of generality, we assume $t$ to be of the form $\lambda x.u$ as $t$ will reduce to such a term, given it realizes a function and, applied to $\alpha$, returns a value.

Thus, we are left to prove the result with $t[\alpha/x] \underset{l}{\rhd^\star_\sigma} \overline{n}$. By induction on the number of reduction steps, we prove that there exists modulus $m$ such that for all $\beta$ such which agrees on the prefix $\alpha\restriction_m$, then $t[\beta/x] \underset{l}{\rhd^\star_\sigma} \overline{n}$. Let us proceed by case analysis on the first step reduction:

- the redex can be outside $\alpha$, meaning that there is $u$ such that $t[\alpha/x] \underset{l}{\rhd_\sigma} u[\alpha/x]$, then $t[\beta/x] \underset{l}{\rhd_\sigma} u[\beta/x]$ for any $\beta$. Using the induction hypothesis, the result follows.
- the redex can involve $\alpha$. In this case, because of the reduction strategy, we find a right context $E[\ ]$ and a term $u$ such that $t[\alpha/x] = E[\alpha\ u[\alpha/x]][\alpha/x]$.
  By regrouping reductions, we only focus on the inner term, namely $\alpha\ u[\alpha/x]$: the induction hypothesis will be applied to the resulting term because $\alpha\ u[\alpha/x]$ reduces at least once. Let $v \triangleq u[\alpha/x]$. We claim that $v \rhd^\star_\sigma \overline{i}$ for some $i \in \mathbb{N}$. Indeed, we can assume without loss of generality that $\alpha$ does not reduce on a term which does not realizes Nat: it suffices to replace the term $\alpha$ with the term $\mathrm{rec}_{\mathrm{Nat}}\ (\alpha\ 0)\ (\lambda n\ x.\alpha(S\ n))$ to have a function which still realizes $\mathrm{Nat} \to \mathrm{Nat}$ but no longer reduces on terms which do not realize Nat.
  Thus, $v \rhd^\star_\sigma \overline{i}$, which implies that $\alpha\ v \rhd^\star_\sigma \alpha\ \overline{i}$, but then $\alpha\ \overline{i} \rhd^\star_\sigma \overline{\alpha(i)}$ so, in the end, $\alpha\ v \rhd^\star_\sigma \overline{\alpha(i)}$ and (by confluence) $\alpha\ v \underset{l}{\rhd^\star_\sigma} \overline{\alpha(i)}$.
  Now, by induction hypothesis, we find a prefix $\alpha\restriction_m$ such that for any $\beta$ with this prefix, $E[\overline{\alpha(i)}][\beta/x] \underset{l}{\rhd^\star_\sigma} \overline{n}$ and $u[\beta/x] \underset{l}{\rhd^\star_\sigma} \overline{i}$. If $i > m$, we can strengthen the condition by considering $i$ and the longer prefix $\alpha\restriction_i$. Then, for $\beta$ with this prefix:

  $$t[\beta/x] = E[\beta\ u[\beta/x]][\beta/x] \rhd^\star_\sigma E[\beta\ \overline{i}][\beta/x] \underset{l}{\rhd^\star_\sigma} E[\overline{\alpha(i)}][\beta/x] \underset{l}{\rhd^\star_\sigma} \overline{n}$$

  Thus, by induction, we deduce that there exists such a prefix. ◀

Using this lemma and the $\lambda$-terms from the previous subsection, we can now realize FT.

▶ **Theorem 28** (Realization of Fan Theorem). *The exists a term* $\mathsf{fan}$ *such that* $\mathsf{fan} \Vdash\!\Vdash \mathrm{FT}$.

**Proof.** Let us prove that there exists a term $t_0$ such that $\mathsf{fan} \triangleq \lambda b.\langle n_{C_b}\ 0, t_0\rangle \Vdash\!\Vdash \mathrm{FT}$. Let $\sigma \in \mathcal{O}$ be a reduction context of length $m$ and

$$b \Vdash_\sigma \forall \alpha^{\{\mathrm{Nat}\to\mathrm{Bool}\}}, \exists n^{\{\mathrm{Nat}\}}, B(\alpha\restriction_n)$$

for some function $B : \mathrm{List}(\mathbb{B}) \to \mathbf{SAT}$ closed by extension. By saturation, it suffices to show that $\langle n_{C_b}\ 0, t_0\rangle \Vdash_\sigma \exists n^{\{\mathrm{Nat}\}}, \forall \alpha^{\{\mathrm{Nat}\to\mathrm{Bool}\}}, B(\alpha\restriction_n)$ but, from Remark 26, we already know that $n_{C_b}\ 0$ either reduces to an integer or diverges. If it reduces to an integer, then this integer is a uniform bound for $C$, but as $C \subseteq B$, it follows that $n_{C_b}\ 0$ is a uniform bound for $B$ as well. Assuming that $n_{C_b}\ 0$ indeed reduces to an integer $n_0$, let us describe how to implement a term $t_0$ such that $t_0 \Vdash_\sigma \forall \alpha^{\{\mathrm{Nat}\to\mathrm{Bool}\}}, B(\alpha\restriction_{n_0})$. Let $\alpha : \mathrm{Nat} \to \mathrm{Bool}$ be any function realized by a code $e_\alpha$. By definition of $n_{C_b}$, in particular $C(\alpha\restriction_{n_0})$ holds. This means that during the enumeration, the algorithm found a list $\ell'$ such that $|\ell'| < n_0$ and $\ell' 0^\infty \restriction_b \preccurlyeq_{\mathrm{pref}} \alpha\restriction_{n_0}$. In particular, $\pi_2\,(b\,(\ell' 0^\infty)) \Vdash_\sigma B(\ell' 0^\infty\restriction_b)$, and we finally obtain a realizer of $B(\alpha\restriction_{n_0})$ using that $B$ is monotone. Using again a fixpoint and an enumeration of lists to find $\ell'$, this mechanism can indeed be implemented by a term $t_0$ which only depends on $b$.

Now, suppose $n_{C_b}\ 0$ diverges. This means that for any $n \in \mathbb{N}$, there exists some word $w_n \in \mathbb{B}^\star$ such that $\neg C(w_n)$. By applying Weak König's Lemma (in our meta-theory), we find an infinite path $\alpha : \mathbb{N} \to \mathbb{B}$ such that for all $n \in \mathbb{N}, \neg C(\alpha\restriction_n)$. We now enrich our reduction context with this alpha: $\tau \triangleq \sigma \frown \alpha$, so that $\xi_m \Vdash_\tau (\mathrm{Nat} \to \mathrm{Bool})(\alpha)$. By definition of $b$, we know that $b\ \xi_m \Vdash_\tau \exists n^{\{\mathrm{Nat}\}}, B(\alpha\restriction_n)$, which we can destruct as some $n \in \mathbb{N}$ such that $\pi_1\,(b\,\xi_m) \rhd_\tau^\star n$ and $\pi_2\,(b\,\xi_m) \Vdash_\tau B(\alpha\restriction_n)$. Using Lemma 27, we can find an index $p \in \mathbb{N}$ such that

$$\forall(\beta \Vdash_\tau \mathrm{Nat} \to \mathrm{Bool}), (\forall i < p, \beta\ \overline{i} \rhd_\tau^\star \overline{\alpha(i)}) \implies \pi_1(b\ \beta) \rhd_\tau^\star n$$

Let $M \triangleq \max(n, p)$. By continuity, we know that $\pi_1\ (b\ (\alpha\restriction_M\ 0^\infty)) \rhd_\tau^\star n$, so when computing $C_b\ (\alpha\restriction_M)$, there exists a list $\ell$ of length $\leq M$ such that $\alpha\restriction_{\pi_1\ (b\ \ell 0^\infty)}$ is a prefix of $\alpha\restriction_M$, so $C(\alpha\restriction_M)$ is realized. But by hypothesis, $\neg C(\alpha\restriction_M)$; hence a contradiction.

This means that there is no case in which $n_{C_b}\,0$ diverges, so

$$\langle n_{C_b}\ 0, \lambda\alpha.\pi_2\,(b\,\alpha)\rangle \Vdash_\sigma \exists n^{\{\mathrm{Nat}\}}, \forall \alpha^{\{\mathrm{Nat}\to\mathrm{Bool}\}}, B(\alpha\restriction_n)$$ ◀

Not only does the model satisfy FT, but it also refutes WKL. By this, we mean that $\mathrm{WKL} \notin \mathscr{T}_{\Vdash}$, making our model a separating model between the two principles.

To prove that WKL is not realized, we use the Kleene tree [30] construction (which can be relativized without issue). This tree is a standard construction in computability theory: it is a decidable tree which is infinite but has no computable infinite path. In the case of our model, every realized function in a context $[f_1, \dots, f_p]$ is $f_1 - \cdots - f_p$-computable, and the Kleene tree construction can be relativized, giving a $f_1 - \cdots - f_p$-computable infinite tree with no $f_1 - \cdots - f_p$ infinite path. Hence, in every context $\sigma$, there is no $t$ such that $t \Vdash_\sigma \mathrm{WKL}$.

▶ **Theorem 29.** *For every $\sigma \in \mathcal{O}$,* $\mathrm{WKL} \notin \mathscr{T}_{\Vdash_\sigma}$, *hence* $\mathrm{WKL} \notin \mathscr{T}_{\Vdash}$.

## 5 A robust interpretation of the Fan Theorem

The realizability model developed in Section 4 relies on the ideas of Lubarsky and Rathjen's work [35] to prove FT. For the argument in this proof to work, the two main conditions that emerge are

- a notion of continuity for the individuals realized by a code,
- having a Kripke-style notion of future worlds in which, given a tree $T$ at some world $\sigma$, there exists a future $\tau$ in which a path in $T$ can be computed.

| | Logical Const. | Program Const. | Evidence Relation |
|---|---|---|---|
| *Reflexivity* | | $e_{\mathrm{id}} \in E$ | $e_{\mathrm{id}} \Vdash \varphi \leq \varphi$ |
| *Transitivity* | | $\circ \in E \times E \to E$ | $e \Vdash \varphi_1 \leq \varphi_2 \implies e' \Vdash \varphi_2 \leq \varphi_3 \implies e' \circ e \Vdash \varphi_1 \leq \varphi_3$ |
| *Top* | $\top \in \Phi$ | $e_\top \in E$ | $e_\top \Vdash \varphi \leq \top$ |
| *Conjunction* | $\wedge \in \Phi \times \Phi \to \Phi$ | $(\!|\cdot,\cdot|\!) \in E \times E \to E$<br>$e_{\pi_1}, e_{\pi_2} \in E$ | $e_1 \Vdash \psi \leq \varphi_1 \implies e_2 \Vdash \psi \leq \varphi_2 \implies (\!|e_1, e_2|\!) \Vdash \psi \leq \varphi_1 \wedge \varphi_2$<br>$e_{\pi_1} \Vdash \varphi_1 \wedge \varphi_2 \leq \varphi_1 \qquad e_{\pi_2} \Vdash \varphi_1 \wedge \varphi_2 \leq \varphi_2$ |
| *Universal Implication* | $\supset \in \Phi \times \mathscr{P}(\Phi) \to \Phi$ | $\lambda \in E \to E$<br>$e_{\mathrm{eval}} \in E$ | $(\forall \varphi \in \vec{\varphi},\ e \Vdash \varphi_1 \wedge \varphi_2 \leq \varphi) \implies \lambda\, e \Vdash \varphi_1 \leq \varphi_2 \supset \vec{\varphi}$<br>$\forall \varphi \in \vec{\varphi},\ e_{\mathrm{eval}} \Vdash (\varphi_1 \supset \vec{\varphi}) \wedge \varphi_1 \leq \varphi$ |

**Figure 3** Evidenced Frame constructs, where $\vec{\varphi} \in \mathscr{P}(\Phi)$ and the evidence relations are universally quantified.

We shall now emphasize the robustness of this approach: we show that the interpretation is actually independent of the particular choice of a computational system as long as these conditions are satisfied. To that end, we will use the notion of *evidenced frames* introduced in [12]. Evidenced frames (which we will shorten as EF) are combinatorial structures encompassing the notion of a realizability model: roughly speaking, an EF is a triple $(\Phi, E, \cdot \Vdash \cdot \leq \cdot)$ where $\Phi$ is a set of formulas describing the set of truth values, $E$ is a set of evidences describing the realizers, and the relation $\cdot \Vdash \cdot \leq \cdot$ is akin to the logical inequality $\subseteq_{\mathbf{SAT}}$ where the witness is made explicit. In particular, it has the advantage of allowing one to abstract away from the implementation details to show that any computational system satisfying some specifications (*e.g.*, memoization features) will induce a realizability model satisfying the expected logical principles (*e.g.*, countable choice [12]). In this section, we therefore seek for a general way to state the two conditions above (as well as a few others which will arise later) using EFs.

In contrast with previous works using EFs [12, 18, 11, 10], where concrete structures used to define realizability interpretations (*e.g.*, monadic combinatorial algebras) are proven to induce an EF, here we will rather focus only on EFs: the concrete structure we study will be introduced as an additional structure over EFs.

This section ends with the statement of Theorem 41, which is a more abstract variant of Theorem 28: in this version, the realizability structure realizing FT can have realizers of any form ($\lambda$-terms, Turing machines…) as long as it gives an implementation of integers with its recursor, satisfies the two conditions mentioned above for the argument to work, plus two additional technical conditions.

## 5.1 The induced Evidenced Frame

We first recall the definition of an Evidenced Frame [12].

▶ **Definition 30** (Evidenced Frame). *An* evidenced frame *is a triple* $\mathcal{E} = (\Phi, E, \cdot \Vdash \cdot \leq \cdot)$, *where* $\Phi$ *is a set of propositions,* $E$ *is a set of evidence, and* $e \Vdash \phi_1 \leq \phi_2$ *is a ternary evidence relation on* $E \times \Phi \times \Phi$, *along with the structure captured in Figure 3.*

As we want to generalize the argument of Theorem 28 to Evidenced Frames, we first show that the model we defined indeed induces one. As a corollary, this shows besides that our model also gives rise to a tripos and a topos [12].

▶ **Proposition 31.** *We define* $E \triangleq \Lambda$ *and* $\Phi \triangleq \mathbf{SAT}$, *together with the relation*

$$t \Vdash (A_\sigma) \leq (B_\sigma) \triangleq \forall \sigma \in \mathcal{O}, \forall u \in A_\sigma, tu \in B_\sigma$$

*Then the triple* $(\Phi, E, \cdot \Vdash \cdot \leq \cdot)$ *is an evidenced frame.*

**Proof.** Observe that $(A_\sigma) \subseteq_{\mathbf{SAT}} (B_\sigma)$ is exactly the proposition $t \Vdash (A_\sigma) \leq (B_\sigma)$ for which we erase the witness $t$. The different components are thus defined by using the structure of **SAT** (but taking care of evidences provided by the expected $\lambda$-terms), which is a Heyting pre-algebra with intersection, and those properties (including the Remark 15) suffice to show that $\mathcal{E}_{\mathbf{SAT}}$ is an Evidenced Frame. As an example, the conjunction function is simply provided by $\cdot \wedge \cdot \triangleq \cdot \wedge_{\mathbf{SAT}} \cdot$ while the pairing function is defined by $(\!|t, u|\!) \triangleq \lambda x.\langle t\ x, u\ x\rangle$ together with $e_{\pi_1} \triangleq \pi_1$ and $e_{\pi_2} \triangleq \pi_2$. It is then easy to check that these definitions satisfy the necessary rules for the evidence relation $\cdot \Vdash \cdot \leq \cdot$. ◀

▶ Remark 32. For any $\sigma \in \mathcal{O}$, it is possible to construct the Evidenced Frame $\mathcal{E}_{\mathbf{SAT}_\sigma}$ where

$$E \triangleq \Lambda \qquad \Phi \triangleq \left\{ (A_\sigma) \in \prod_{\tau \succcurlyeq \sigma} \mathbf{SAT}_\tau \;\middle|\; \forall \tau, \tau' \succcurlyeq \sigma, \tau \preccurlyeq \tau' \implies A_\tau \subseteq A_{\tau'} \right\}$$

and with the same functions and evidenced.

From our definition, the second sanity check to do is to show that for any proposition $\vdash \varphi : \mathrm{Prop}$ in HOL, $t \Vdash\!\Vdash \varphi$ if and only if $\lambda\_.t \Vdash \top \leq [\![\varphi]\!]$. The proof is by induction on $\varphi$, and is mostly straightforward.

## 5.2 A robust generalization of the proof

We will now generalize our proof of FT by abstracting its core over the abstract framework provided by evidenced frames. Scrutinizing the argument in the proof of Theorem 28, besides the whole Kripke-like structure of our model, we identify the following key ingredients:

1. The realizability model accounts at least for system T functions (for instance, encodings allow one to talk about functions $\mathrm{List}(\mathrm{Nat}) \to \mathrm{Bool}$).
2. A function $f$ realized in some context $\sigma$ can be applied to an argument $x$ realized in some extension of the context, $\tau \succcurlyeq \sigma$, where $x$ is possibly non realized in $\sigma$.
3. The functions $(\mathrm{Nat} \to \mathrm{Nat}) \to \mathrm{Nat}$ are continuous, in the sense that for any function $f : (\mathrm{Nat} \to \mathrm{Nat}) \to \mathrm{Nat}$ and $\alpha : \mathrm{Nat} \to \mathrm{Nat}$ with a code, there exists a modulus $n$ such that $\forall \beta, \alpha \restriction_n = \beta \restriction_n \implies f(\alpha) = f(\beta)$.
4. Realizers can do an unbounded search, as long as the meta-theory can prove that the search will terminate.

A first step to the generalization is to account for the Kripke semantics we used in our model. In fact, the category semantic way of giving a Kripke model is to consider a category $\mathbf{C}$, a preordered set $(W, \leq)$ and to give a functor $F : W \to \mathbf{C}$. This gives a guideline for our generalization: we construct a well-designed category for our work, where morphisms preserve the information we expect, and consider a functor from a preorder to such a category.

### 5.2.1 Evidenced frames with data types

Observe that the functions which the former conditions refer to are actually individuals witnessed by codes in our model. Technically, this was handled by means of HOL-terms of the appropriated sorts and the relativization predicate $S(\mathbf{t})$. Again, we would like to maintain the distinction between terms witnessing individuals (the *data types*) and realizers for formulas. Therefore, we extend the definition of evidenced frames to account for data types: those are evidenced frames for which we explicitly add a representation of $\mathbb{N}$. This can be seen as taking an evidenced frame $\mathcal{E}$ and pulling back from its category of assemblies

a natural number object. For this definition, as we want at least system T functions, we also require an encoding (which is the one we expect for assemblies) of functions, and thus a minimal system T types syntax.

▶ **Definition 33** (Evidenced Frame with data types)**.** *An evidenced frame with data types (EFdata, for short), is a tuple* $(\Phi, E, \cdot \Vdash \cdot \leq \cdot, \ulcorner\cdot\urcorner, e_0, e_S, e_{\mathrm{rec}})$ *where:*

- $(\Phi, E, \cdot \Vdash \cdot \leq \cdot)$ *is an evidenced frame.*
- $\ulcorner\cdot\urcorner : \mathbb{N} \to \Phi$ *is an encoding function.*
- *for all functions* $f : \mathbb{N} \to \Phi$,

$$e_{\mathrm{rec}} \Vdash \top \leq f(0) \supset \left( \prod_{n\in\mathbb{N}} \ulcorner n \urcorner \supset f(n) \supset f(n+1) \right) \supset \prod_{n\in\mathbb{N}} \ulcorner n \urcorner \supset f(n)$$

- $e_0 \Vdash \top \leq \ulcorner 0 \urcorner$
- *for all* $n \in \mathbb{N}$, $e_S \Vdash \ulcorner n \urcorner \leq \ulcorner n+1 \urcorner$

▶ Remark 34. For any $n \in \mathbb{N}$, $\ulcorner n \urcorner$ is realized by $S^n 0$. Also, any system T definable function is realized by the naturally associated term: we can define the encoding of a system T type $T \to U$ by $\ulcorner f \urcorner \triangleq \prod_{x\in T} \ulcorner x \urcorner \supset \ulcorner f(x) \urcorner$.

To construct the category of EFdatas, we also need to define morphisms. The notion of morphism we introduce is only intended to state a correct generalized version of Theorem 28, they are (very) strict morphisms which preserve every construction (they preserve the evidences, the functions on propositions, *etc.*) and are not to be considered as the natural notion of morphism for EFdata. For example, EF morphisms as defined in [12] do not require the constructions and morphism to commute on the nose.

▶ **Definition 35** (Strict EFdata morphism)**.** *Let* $\mathcal{E}_1, \mathcal{E}_2$ *be two EFdata (each component of which will be written respectively* $E_1, \Phi_1, \ldots$ *and* $E_2, \Phi_2, \ldots$*). A strict morphism of EFdata is the data of two functions* $F_\Phi : \Phi_1 \to \Phi_2$ *and* $F_E : E_1 \to E_2$ *(we will just write* $F$*) commuting with every constructor of an EFdata, that is :*

- $e \Vdash \varphi \leq \psi \implies F(e) \Vdash F(\varphi) \leq F(\psi)$
- $F(\top_1) = \top_2$ *and* $F(e_{\top 1}) = e_{\top 2}$
- $F((\varphi \wedge \psi)_1) = (F(\varphi) \wedge F(\psi))_2$
- $F(\langle\!| e, e' |\!\rangle_1) = \langle\!| F(e), F(e') |\!\rangle_2$
- $F(e_{\pi_1 1}) = e_{\pi_1 2}$ *and* $F(e_{\pi_2 1}) = e_{\pi_2 2}$
- $F((\varphi \supset \vec{\psi})_1) = (F(\varphi) \supset \{F(\psi) \mid \psi \in \vec{\psi}\})_2$
- $F(e_{\mathrm{eval}1}) = e_{\mathrm{eval}2}$ *and* $F(\lambda\, e_1) = \lambda\, F(e)_2$
- $F(e_{01}) = e_{02}$, $F(e_{S1}) = e_{S2}$ *and* $F(e_{\mathrm{rec}1}) = e_{\mathrm{rec}2}$
- $F_\Phi(\ulcorner n \urcorner_1) = \ulcorner n \urcorner_2$

*where the rules hold for all* $\varphi, \psi, \chi \in \Phi_1$,$\vec{\varphi} \in \mathscr{P}(\Phi_1)$, $e, e' \in E_1$, $n \in \mathbb{N}$.

With this notion of morphism, we can formalize the *future world* notion by taking a preorder $\mathbf{C}$ and constructing a functor $\mathbf{C} \to \mathbf{EFData}$. This makes it easy to define the condition that in any world and infinite binary tree $T$ at this world, there is a future in which a path in $T$ exists.

To generalize our argument in the setting of EF, we need some computational conditions. To begin with, we need to have a strong notion underlying the quantifier $\exists$, that we call strong existential. The strong existential allows one to use a proof of $\exists x, \varphi$ to extract an object $a$ such that $\varphi$ is true on $a$. In the presence of effects, and even more so with classical logic, the witness used to prove $\exists x, \varphi$ can be impossible to retrieve, making this strong existential not realized [22]. The strong existential is thus a natural condition: when taking the relativized version of $\exists$, we expect the proof to be able to give an explicit witness by its code, and strong existential ensures that this code of witness indeed retrieves the witness.

▶ **Definition 36** (Strong existential)**.** *An EFdata is said to have strong existential if there is an evidence* $e$ *such that for any system* $T$ *type* $\tau$ *and any function* $\varphi(-) : \tau \to \mathrm{Prop}$ *there exists* $a \in \tau$ *such that* $e \Vdash \exists x^\tau, \ulcorner x \urcorner \wedge \varphi(x) \leq \ulcorner a \urcorner \wedge \varphi(a)$.

### 5.2.2 Continuity

Now, we formalize the continuity condition in the setting of EFdatas. Continuity in computable systems is a well-studied notion, for which a wide literature exists especially in constructive settings [40, 26, 42, 14, 1]. For the purpose of this work, we will only consider the following (weak) version of continuity.

▶ **Definition 37** (Continuity). *Let* $(\Phi, E, \cdot \Vdash \cdot \leq \cdot, \ulcorner\cdot\urcorner, e_0, e_S, e_{\mathrm{rec}})$ *be an EFdata. This EFdata is said to have continuity if*

$$\forall f : (\mathbb{N} \to \mathbb{N}) \to \mathbb{N}, \forall \alpha : \mathbb{N} \to \mathbb{N}, \forall t \Vdash \top \leq \ulcorner f \urcorner,$$
$$\forall u \Vdash \top \leq \ulcorner \alpha \urcorner, \exists n \in \mathbb{N}, \forall \beta : \mathbb{N} \to \mathbb{N}, \alpha \restriction_n = \beta \restriction_n \implies f(\alpha) = f(\beta)$$

▶ **Example 38.** Instead of oracles $\xi_n$ interpreting functions $\mathbb{N} \to \mathbb{N}$, we could add as an oracle a program $\xi$ such that $\xi\ t$ reduces to tt or ff depending on whether $t$ (weakly) terminates or not, supposing $t$ is closed and does not contain $\xi$. In this case, it is possible to realize the function deciding if a path $\alpha : \mathbb{N} \to \mathbb{B}$ is the constant path 0 by reading each bit of $\alpha$ until its first 1. This function is not continuous, so the realizability model we could build on this computational system does not have continuity. For most reasonable computational systems not adding non computable higher-order function and without `quote` instruction, continuity is satisfied.

### 5.2.3 Unbounded search

Another principle that is used in Theorem 28, but not necessarily realized in an arbitrary EF, is the ability to conduct an unbounded search. To find the uniform bound, we use a fixpoint akin to the $\mu$ operator from [31]. This is a reasonable assumption: for instance, every PCA implements unbounded search thanks to the fixpoint combinator $Y$.

▶ **Definition 39** (Unbounded search). *An EFdata is said to have unbounded search if there exists* $e$ *such that, for any* $f : \mathbb{N} \to \mathbb{N}$ *having at least one zero,*

$$e \Vdash \ulcorner f \urcorner \leq (\exists (n : \mathbb{N}), \ulcorner n \urcorner \wedge f(n) = 0)$$

▶ **Example 40.** Typed realizability settings based on the $\lambda$-calculus or any other computational system where all realizers have to be terminating would forbid such unbounded search, but extension to PCF or untyped settings naturally allow for such an operator.

### 5.2.4 The robust theorem

We can now state our generalized theorem about FT. The conditions are the ones introduced above, but they are not needed to be satisfied in every world. In the proof of Theorem 28, there are two worlds: the one from which we get the bar, and the one in which we get the path avoiding $C(b)$. The continuity and strong existential only need to be satisfied in the latter and, up to going to an even further future, can be checked only on a dense set of worlds. The unbounded search is used on the base world (the one containing our model), which is the only use of this hypothesis. As for Theorem 28, we implicitly assume a classical meta-theory satisfying WKL, which is in line with the standard approach in realizability to show the validity of realizer by assuming a principle equivalent in the meta-theory.

▶ **Theorem 41.** *Let* $\mathcal{E}$ *be an EFdata with unbounded search. Let* $(W, \leq, \mathbb{1})$ *be an ordered set with lower bound* $\mathbb{1}$ *and* $F : W \to \mathbf{EFData}$ *be a functor such that* $F(\mathbb{1}) = \mathcal{E}$*. If the following conditions are satisfied:*

- *there is a dense subset* $W' \subseteq W$ *such that for any* $w \in W', F(w)$ *has continuity and strong existential.*
- *for any* $w \in W$*, function* $B : \mathrm{List}(\mathbb{B}) \to \mathbb{B}$ *with an infinite path and a code in* $w$*, there is an element* $w' \geq w$*, a code* $e_\alpha$ *of an infinite path* $\alpha$ *in* $B$ *and a code* $e \in E_{w'}$ *such that for any* $n \in \mathbb{N}$*,* $e \Vdash \ulcorner n \urcorner \leq \alpha\restriction_n \in B$

*then* $\mathcal{E} \Vdash \mathrm{FT}$*.*

**Proof.** We apply the proof of Theorem 28 in the general setting. To make the proof easier to read, we consider the types Bool and List(Bool), as they can be encoded inside integers and the usual manipulations can be simulated on the encodings.

The proposition we want to prove is that there exists $e \in E$ such that for any predicate $B : \mathrm{List}(\mathrm{Bool}) \to \Phi$ closed by extension,

$$e \Vdash (\forall \alpha^{\{\mathrm{Nat}\to\mathrm{Bool}\}}, \exists n^{\{\mathrm{Nat}\}}, \alpha\restriction_n \in B) \implies \exists n^{\{\mathrm{Nat}\}}, \forall \alpha^{\{\mathrm{Nat}\to\mathrm{Bool}\}}, \alpha\restriction_n \in B$$

We assume there is $b$ witnessing the bar on $B$, and the definition of $C(b)$ from it is system T defined: this ensures that we can define the same function using the witnesses $e_0, e_S, e_{\mathrm{rec}}$. In the definition of $n_{C(b)}$, we cannot define the fixpoint by itself, but we can define the function $C'_b$ which, for any $n$, computes whether every $\ell \in \mathrm{List}(\mathrm{Bool})$ of size $n$ is inside $C(b)$. This function has a code $e_{C'_b}$.

By hypothesis, the evidenced frame has unbounded search, so we can apply it to $\neg \circ C'_b$ (because we search for a $n$ such that $C'_b(n) = 1$). Combining it with the code $e_{C'_b}$, this gives an evidence of a uniform bound $n_0$. By a similar reasoning, we can mimic the definition of $t_0$ (in the proof of Theorem 28). Thus, all that is left to prove is that $\neg \circ C'_b$ indeed has a 0.

Suppose that $C'_b(n) = 0$ for any $n$. This means that for any $n \in \mathbb{N}$, there is a list of size $n$ avoiding $C$. Using Weak König's Lemma on $\overline{C}$ the complement of $C$ (which is a tree), we find a path $\alpha : \mathbb{N} \to \mathbb{B}$ such that $\alpha\restriction_n \notin C$ for any $n \in \mathbb{N}$. By hypothesis on our functor $W \to \mathbf{EFData}$, we find $w' \in W$ and a code $e_\alpha \in \ulcorner \alpha \urcorner$ in $w'$. By the other hypothesis, we can assume (up to going to a future world) that $F(w')$ has continuity.

Let $g$ be the morphism from $\mathcal{E}$ to $F(w')$. By construction of morphisms, we have

$$g(b) \Vdash g(\forall \alpha^{\{\mathrm{Nat}\to\mathrm{Bool}\}}, \exists n^{\{\mathrm{Nat}\}}, \alpha\restriction_n \in B)$$

but as $g$ commutes with $\ulcorner - \urcorner$, with $\implies$, and with quantification $\forall$, we deduce that

$$g(b) \Vdash \forall \alpha^{\{\mathrm{Nat}\to\mathrm{Bool}\}}, \exists n^{\{\mathrm{Nat}\}}, \alpha\restriction_n \in g(B)$$

hence, $g(b)e_\alpha \Vdash \exists n^{\{\mathrm{Nat}\}}, \alpha\restriction_n \in g(B)$. Using the fact that $g(b)$ is a code of a function $(\mathbb{N} \to \mathbb{N}) \to \mathbb{N}$, we can apply continuity to find $m$ such that $\alpha\restriction_m = \beta\restriction_m \implies g(b)\alpha = g(b)\beta$ for any $\beta : \mathbb{N} \to \mathbb{N}$ with a code. This means that we can apply the same argument as in Theorem 28 to have a contradiction : taking $M$ the maximum of this $m$ and of the encoded $n^{\{\mathrm{Nat}\}}$ by $g(b)e_\alpha$ (that we can take by strong existential), we get $\alpha\restriction_M \in C(b)$, which is contradicting the fact that $\alpha$ always avoids $C(b)$. ◀

▶ Remark 42. In fact, while the previous proof mimics the proof of Theorem 28, it should even be possible to weaken the hypotheses. Indeed, in the proof, we only need that infinite computable trees with a code inside $\mathcal{E}$ have a future in which there is an infinite path (with a code).

## 5.3 Application, limitations

Having Theorem 41 at hands, we give examples of realizability models satisfying the conditions (and thus, realizing FT):

- The first example is the one given before. For each $\sigma \in \mathcal{O}$, we can define the EFdata $\mathcal{E}_{\mathbf{SAT}_\sigma}$ from Remark 32. For $\sigma, \tau \in \mathcal{O}$ such that $\sigma \preccurlyeq \tau$, we have a morphism of EFdata given by inclusion. For each $\sigma$, the EFdata $\mathcal{E}_{\mathbf{SAT}_\sigma}$ has continuity, strong existential and unbounded search, hence the fact that it satisfies FT.
- The second example is the historical one: the second Kleene algebra $K_2$ [32] satisfies the premises. The preorder is the trivial one, with only one element. Continuity is by definition of the PCA (it is the set of continuous functions $(\mathbb{N} \to \mathbb{N}) \to \mathbb{N}$), unbounded search can be defined for any PCA by encoding the combinator $Y$ in combinatory logic. Any function $\mathbb{N} \to \mathbb{N}$ has a code, hence the condition that a path exists for any decidable tree encoded. Finally, strong existential is true for intuitionistic PCAs. We thus find back the usual result that $K_2$ satisfies FT.
- A third example is the model we based our generalization on: the model by Lubarsky and Rathjen [35].

Observe that the existence of a path in a possible future is essential. For example, taking the realizability model given by $\mathbf{SAT}_\sigma$, with local computation at some context $\sigma \in \mathcal{O}$, the model does not realize FT. To see this, consider the Kleene tree relativized to $\sigma$-computable functions. For any $\sigma$-computable path, given by a code $e$, we can simulate $e$ for increasing time duration until the finite path gets out of the tree, which will always happen because the infinite path in the Kleene tree are all uncomputable. This thus gives a bar for the Kleene tree, but this bar cannot be uniform as the length of lists in this tree is unbounded.

Finally, we advocate that the restriction of working on EFdatas instead of just EF is not a limitation in practice. To see this, take an EF $(E, \Phi, \cdot \Vdash \cdot \leq \cdot)$. Using the UFam construction [12, 28] to get a tripos, and then using the tripos to topos construction, we can describe the objects of this topos. We focus on one of its full subcategories given by the assemblies. In this category, objects are given by a set $X$ and a function $E_X : X \to \Phi$, while morphisms $(X, E_X) \to (Y, E_Y)$ are given by a function $f : X \to Y$ and an evidence $e \Vdash E_X(x) \leq E_Y(f(x))$ uniform on $x \in X$.

This exactly corresponds to the way we define codes for system T types, so every construction in the EFdata can be translated inside the realizability topos. The object associated to $(\mathbb{N}, \ulcorner - \urcorner)$ in the realizability topos is a natural number object (NNO) [34]: $e_0$ defines the element $0 \in \mathbb{N}$, $e_S$ defines the function $\mathbb{N} \to \mathbb{N}$ and $e_{\mathrm{rec}}$ ensures the initiality of $1 + \mathbb{N} \xrightarrow{[0,S]} \mathbb{N}$ among the algebras for $X \mapsto 1 + X$.

Hence, to get an EFdata from an EF $\mathcal{E}$, one needs to find a NNO inside the realizability topos of $\mathcal{E}$, for which the initial algebra property is assured in a uniform way (*i.e.* by the same witness $e \in E$), and to pull back the associated data to make $e_0, e_S, e_{\mathrm{rec}}$. In practice, any realizability model with an implementation of integers (such as Church integers) possesses such an object, which is already a broad class of models.

# 6 Conclusion, future work

This article introduces a realizability model separating FT from WKL based on a $\lambda$-calculus extended with oracles.

We provide a concrete realizer for FT, which performs a naive extensive search for a uniform bound. More importantly, we identify the key features of this realizability interpretation that allow to satisfy FT:

**(i)** realizers are continuous,
**(ii)** the computational system can be extended with infinite paths for any computable tree,
**(iii)** it accounts for strong existential,
**(iv)** it features an unbounded search.

To support our claim that this computational interpretation is indeed robust and does not rely on other peculiarities of the underlying $\lambda$-calculus, we formulate it in the general framework of evidenced frames. This provides us with a result which applies to any realizability model exhibiting those properties. This generality implies that the computational content we outline is explicit and is not a degenerate case of a particular model.

Besides, we know that the argument cannot work without the second property, because the Kleene tree in the standard realizability model (with no oracle) has a bar which is not uniform. Nonetheless, these conditions allow the construction of models (*e.g.* the one explicitly constructed in this article) which satisfy FT without satisfying WKL, indicating that they are parcimonious at least with regard to the hierarchy established in [6].

Beyond the specific case of the Fan Theorem, we advocate for a computational approach to constructive reverse maths, and we expect that an exploration of logical principles can be conducted by this methodology. Especially, we believe that a reasonable answer to the problem of finding the computational content of a logical principle $\varphi$ should follow the same path:

- a general definition of computational system must be given, and this computational system can be made into an EF (or another general realizability framework, like implicative algebra [37])
- on this notion of computational system, we can define a condition $(a)$ such that a computational system satisfying $(a)$ always satisfies $\varphi$ in the induced realizability model
- among the computational systems satisfying $(a)$, we can find ones that do not satisfy a strictly stronger principle than $\varphi$.

This methodology has been applied to FT in this article, but can be transposed to any other logical principle: the hierarchy of choice principles in [6] is a natural candidate, as it gives both a list of principles and a good criterion to exhibit what strictly stronger principle we do not expect to satisfy (namely the principles above in the hierarchy, and the classical counterpart of intuitionistic principles).

Another interesting question arises when considering alternative versions of FT. For example, it is possible to weaken the premise that *any bar must be uniform* by only considering bars on $\Pi^1_0$ trees or on computable trees (such alternatives are considered in [35]). Is there a variation of the Theorem 41 accounting for such weakenings?

## References


1 Martin Baillon, Yannick Forster, Assia Mahboubi, Pierre-Marie Pédrot, and Matthieu Piquerez. A Zoo of Continuity Properties in Constructive Type Theory. In *10th International Conference on Formal Structures for Computation and Deduction (FSCD 2025)*, Birmingham, France, July 2025. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.FSCD.2025.9.

2 Andrej Bauer. Instance reducibility and Weihrauch degrees. *Logical Methods in Computer Science*, Volume 18, Issue 3, August 2022. doi:10.46298/lmcs-18(3:20)2022.

3 Stefano Berardi, Marc Bezem, and Thierry Coquand. On the computational content of the axiom of choice. *JSL*, 1998. doi:10.2307/2586854.

4 Josef Berger. A decomposition of brouwer's fan theorem. *J. Log. Anal.*, 1, 2009. doi:10.4115/JLA.2009.1.6.

5 Valentin Blot. A direct computational interpretation of second-order arithmetic via update recursion. In *LICS '22*, 2022. doi:10.1145/3531130.3532458.

6 Nuria Brede and Hugo Herbelin. On the logical structure of choice and bar induction principles . In *2021 36th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–13, Los Alamitos, CA, USA, July 2021. IEEE Computer Society. doi:10.1109/LICS52264.2021.9470523.

7 L.E.J. Brouwer. *Begruündung der Mengenlehre unabhängig vom logischen Satz vom ausgeschlossenen Dritten 2. Tl., Theorie der Punktmenge*. Müller, Amsterdam, 1919.

8 Liron Cohen, Sofia A. Faro, and Ross Tate. The effects of effects on constructivism. *MFPS*, 347, 2019. doi:10.1016/j.entcs.2019.09.006.

9 Liron Cohen, Yannick Forster, Dominik Kirst, Bruno Da Rocha Paiva, and Vincent Rahli. Separating markov's principles. In *Proceedings of the 39th Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS '24, New York, NY, USA, 2024. Association for Computing Machinery. doi:10.1145/3661814.3662104.

10 Liron Cohen, Ariel Grunfeld, Dominik Kirst, and Étienne Miquey. From Partial to Monadic: Combinatory Algebra with Effects. In *Proceedings of FSCD 2025*, volume 337 of *Leibniz International Proceedings in Informatics (LIPIcs)*, 2025. doi:10.4230/LIPIcs.FSCD.2025.14.

11 Liron Cohen, Ariel Grunfeld, Dominik Kirst, and Étienne Miquey. Syntactic effectful realizability in higher-order logic. In *40th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2025, Singapore, June 23-26, 2025*, pages 16–30. IEEE, 2025. doi:10.1109/LICS65433.2025.00009.

12 Liron Cohen, Étienne Miquey, and Ross Tate. Evidenced frames: A unifying framework broadening realizability models. In *LICS'21*, 2021. doi:10.1109/LICS52264.2021.9470514.

13 Liron Cohen and Vincent Rahli. Realizing Continuity Using Stateful Computations. In Bartek Klin and Elaine Pimentel, editors, *31st EACSL Annual Conference on Computer Science Logic (CSL 2023)*, volume 252 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 15:1–15:18, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.CSL.2023.15.

14 Martín Escardó. Continuity of Gödel's System T definable functionals via effectful forcing. *Electronic Notes in Theoretical Computer Science*, 298:119–141, 2013. Proceedings of the Twenty-ninth Conference on the Mathematical Foundations of Programming Semantics, MFPS XXIX. doi:10.1016/j.entcs.2013.09.010.

15 Yannick Forster, Dominik Kirst, and Dominik Wehr. Completeness theorems for first-order logic analysed in constructive type theory. *J. Log. Comput.*, 31(1):112–151, 2021. doi:10.1093/logcom/exaa073.

16 Makoto Fujiwara. An Extension of the Equivalence Between Brouwer's Fan Theorem and Weak König's Lemma with a Uniqueness Hypothesis. In Ulrich Berger, Johanna N. Y. Franklin, Florin Manea, and Arno Pauly, editors, *Revolutions and Revelations in Computability*, volume 13359, pages 115–124. Springer International Publishing, Cham, 2022. doi:10.1007/978-3-031-08740-0_10.

17 Makoto Fujiwara and Takako Nemoto. Choice principles characterizing the difference between König's lemma and weak König's lemma in constructive reverse mathematics. *Computability*, 14(1):30–37, February 2025. doi:10.3233/COM-230478.

18 Samuel Gardelle and Étienne Miquey. Do CPS translations also translate realizers? In Timothy Bourke and Delphine Demange, editors, *JFLA 2023 - 34èmes Journées Francophones des Langages Applicatifs*, pages 134–151, Praz-sur-Arly, France, January 2023. URL: https://hal.inria.fr/hal-03910311.

19 Guillaume Geoffroy. Classical realizability as a classifier for nondeterminism. In Anuj Dawar and Erich Grädel, editors, *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2018, Oxford, UK, July 09-12, 2018*, pages 462–471. ACM, 2018. doi:10.1145/3209108.3209140.

**20** Nicolas D. Goodman. Relativized realizability in intuitionistic arithmetic of all finite types. *Journal of Symbolic Logic*, 43(1):23–44, 1978. doi:10.2307/2271946.

**21** Timothy Griffin. A formulae-as-type notion of control. In *POPL '90*, New York, NY, USA, 1990. ACM. doi:10.1145/96709.96714.

**22** Hugo Herbelin. On the degeneracy of sigma-types in presence of computational classical logic. In *Typed Lambda Calculi and Applications, 7th International Conference, TLCA 2005, Nara, Japan, April 21-23, 2005, Proceedings*, pages 209–220, 2005. doi:10.1007/11417170_16.

**23** Hugo Herbelin. A constructive proof of dependent choice, compatible with classical logic. In *Proceedings of the 27th Annual IEEE Symposium on Logic in Computer Science, LICS 2012, Dubrovnik, Croatia, June 25-28, 2012*, 2012. doi:10.1109/LICS.2012.47.

**24** Hugo Herbelin and Danko Illik. An analysis of the constructive content of Henkins's proof of Gödel's completeness theorem. *The Journal of Symbolic Logic*, pages 1–43, 2025. doi:10.1017/jsl.2025.10137.

**25** Marc Hermes and Dominik Kirst. An analysis of tennenbaum's theorem in constructive type theory. *Logical Methods in Computer Science*, Volume 20, Issue 1, March 2024. doi:10.46298/lmcs-20(1:19)2024.

**26** Hajime Ishihara. Continuity properties in constructive mathematics. *Journal of Symbolic Logic*, 57(2):557–565, 1992. doi:10.2307/2275292.

**27** Hajime Ishihara. Reverse mathematics in Bishop's constructive mathematics. *Philosophia Scientiæ*, 6:43–59, 2006. doi:10.4000/philosophiascientiae.406.

**28** B. Jacobs. *Categorical Logic and Type Theory*. Number 141 in Studies in Logic and the Foundations of Mathematics. North Holland, Amsterdam, 1999. URL: http://www.elsevierdirect.com/product.jsp?isbn=9780444508539.

**29** Dominik Kirst and Haoyi Zeng. The blurred drinker paradox: Constructive reverse mathematics of the downward löwenheim-skolem theorem. In *2025 40th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 926–940, 2025. doi:10.1109/LICS65433.2025.00073.

**30** S. C. Kleene. Recursive functions and intuitionistic mathematics. *Journal of Symbolic Logic*, 18(2):181–182, 1953. doi:10.2307/2268961.

**31** Stephen Cole Kleene. Recursive predicates and quantifiers. *Transactions of the American Mathematical Society*, 53:41–73, 1943. doi:10.2307/2267986.

**32** Stephen Cole Kleene and Richard Eugene Vesley. *The Foundations of Intuitionistic Mathematics: Especially in Relation to Recursive Functions*. North-Holland Pub. Co., Amsterdam, 1965.

**33** Jean-Louis Krivine. A call-by-name lambda-calculus machine. In *Higher Order and Symbolic Computation*, 2004. doi:10.1007/s10990-007-9018-9.

**34** F. William Lawvere. Functorial semantics of algebraic theories. *Proceedings of the National Academy of Sciences*, 50(5):869–872, 1963. doi:10.1073/pnas.50.5.869.

**35** Robert S. Lubarsky and Michael Rathjen. Realizability models separating various fan theorems. In Paola Bonizzoni, Vasco Brattka, and Benedikt Löwe, editors, *The Nature of Computation. Logic, Algorithms, Applications*, pages 306–315, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. doi:10.1007/978-3-642-39053-1_35.

**36** Alexandre Miquel. Existential witness extraction in classical realizability and via a negative translation. *Logical Methods in Computer Science*, 7(2), 2011. doi:10.2168/LMCS-7(2:2)2011.

**37** Alexandre Miquel. Implicative algebras: a new foundation for realizability and forcing. *Mathematical Structures in Computer Science*, 30(5), 2020. doi:10.1017/S0960129520000079.

**38** Étienne Miquey. A sequent calculus with dependent types for classical arithmetic. In *LICS 2018*. ACM, 2018. doi:10.1145/3209108.3209199.

**39** Stephen G. Simpson. *Subsystems of second order arithmetic*. Perspectives in mathematical logic. Springer, 2 edition, 1999. doi:10.1017/CBO9780511581007.

**40** A.S. Troelstra and D. van Dalen. *Constructivism in Mathematics: An Introduction*, volume 121 of *Studies in Logic and the Foundations of Mathematics*. Elsevier, 1988. doi:10.1016/S0049-237X(09)70529-4.

**41** Mark van Atten and Dirk van Dalen. Arguments for the continuity principle. *Bulletin of Symbolic Logic*, 8(3):329–347, 2002. doi:10.2178/bsl/1182353892.

**42** Jaap van Oosten. Partial Combinatory Algebras of Functions. *Notre Dame Journal of Formal Logic*, 52(4):431–448, 2011. doi:10.1215/00294527-1499381.

**43** Wim Veldman. On some of brouwer's axioms. *The Bulletin of Symbolic Logic*, 31(1):pp. 1–52, 2025. doi:10.1017/bsl.2024.52.