Open Bar — a Reconciliation between Intuitionistic and Classical Logic

Mark Bickford[a] Liron Cohen[b] Robert L. Constable[a] Vincent Rahli[c]

[a] *Computer Science Department, Cornell University, Ithaca, NY, USA*

[b] *Computer Science Department, Ben-Gurion University, Israel*

[c] *School of Computer Science, University of Birmingham, Birmingham, UK*

Abstract

One the differences between intuitionistic logic and classical logic is their treatment of time. In classical logic truth is atemporal, while in intuitionistic logic truth is time-relative. Due to this difference, intuitionistic logic can derive counterexamples to standard axioms of classical logic. This is because in intuitionistic logic it is possible to acquire new knowledge as time progresses, whereas the classical Law of Excluded Middle (LEM) is essentially flattening the notion of time stating that it is possible to decide whether or not some knowledge will ever be acquired.This seems to indicate an incompatibility between classical logic and intuitionistic logic. However, this paper demonstrates that this is not necessarily so by introducing an intuitionistic type theory along with a Beth-like model for it which provide some middle ground. On one hand they incorporate a notion of time and include evolving mathematical entities in the form of choice sequences, and on the other hand they are consistent with versions of classical axioms such as LEM. Thus, this new type theory provides the basis for a more classically inclined intuitionistic type theory.

Keywords: Intuitionism, Extensional type theory, Realizability, Choice sequences, Classical Logic, Law of Excluded Middle, Theorem proving, Coq, Nuprl

1 Introduction

Classical logic and Intuitioinistic logic are commonly viewed as distinct philosophies. Much of the difference between the two philosophies can in fact be pinned down to the way they handle the notion of *time*. In intuitionistic logic time plays a major role. In fact, the intuitionistic notions of knowledge and truth evolve over time. The seminal concept of intuitionistic mathematics as developed by Brouwer is that of *infinitely proceeding* sequences of choices (called choice sequences) from which the continuum is defined [7, Ch.3]. Choice sequences are a primitive concept of finite sequences of entities (such as natural numbers for example) that are never complete, and can always be extended further with new choices [23; 8; 39; 40; 27; 42; 30]. This embedding of the evolving time in intuitionistic logic entails a notion of computability that goes far beyond that of Church-Turing. Moreover, the concept of evolving knowledge in intuitionistic logic is grounded in Krikpe's Schema, which in turn relies on the notion of choice sequences, and is inconsistent with Church's Thesis [17, Sec.5]. Classical logic, on the other hand, is time-invariant. That is, its notions of knowledge and truth are constant and so the aspect of time is, intuitively speaking, flattened. As mentioned by van Atten, "Many people believe, unlike Brouwer, that mathematical truths are not tensed but eternal—either because such truths are outside time altogether (atemporal) or because they hold in all time (omnitemporal)" [7, p.19].

This critical difference between the two philosophies was in fact used extensively to refute classical results in intuitionistic logic. Brouwer himself used his concept of choice sequences to provide *weak counterexamples* to classical results such as "any real number different from 0 is also apart from 0" [21, Ch.8]. Those counterexamples are called *weak* (or *Brouwerian*) in the sense that they depend on the fact that some formulas have not been either proved or disproved yet (such as the Goldbach conjecture). As explained for example in [13, Ch.1, Sec.1], by defining a choice sequence where 1 can only be picked once such an undecided conjecture has been resolved (proved or disproved), then one could resolve those undecided conjectures using the law of excluded

This paper is electronically published in
Electronic Notes in Theoretical Computer Science
URL: `www.elsevier.nl/locate/entcs`

Fig. 1 Syntax (top) and operational semantics (bottom) of a subset of Nuprl

$$
\begin{array}{llllllllll}
v \in \texttt{Value} ::= & vt & \text{(type)} & \mid \texttt{inl}(t) & \text{(left injection)} & \mid \star & \text{(axiom)} & \mid \langle t_1, t_2 \rangle & \text{(pair)} \\
& \mid \lambda x.t & \text{(lambda)} & \mid \texttt{inr}(t) & \text{(right injection)} & \mid \underline{i} & \text{(integer)} & & \\
vt \in \texttt{Type} ::= & \mathbb{Z} & \text{(integers)} & \mid \mathbf{\Pi} x{:}t_1.t_2 & \text{(product)} & \mid t_1 = t_2 \in t & \text{(equality)} & \mid t_1 < t_2 & \text{(less than)} \\
& \mid \mathbb{U}_i & \text{(universe)} & \mid \mathbf{\Sigma} x{:}t_1.t_2 & \text{(sum)} & \mid t_1 \simeq t_2 & \text{(bisimulation)} & \mid t_1 /\!/ t_2 & \text{(quotient)} \\
& \mid t_1 {+} t_2 & \text{(disjoint union)} & \mid \{x : t_1 \mid t_2\} & \text{(set)} & & & & \\
t \in \texttt{Term} ::= & x & \text{(variable)} & \mid \texttt{let } x := \boxed{t_1} \texttt{ in } t_2 & \text{(call-by-value)} & \mid \texttt{fix}(\boxed{t}) & \text{(fixpoint)} & & \\
& \mid v & \text{(value)} & \mid \texttt{let } x, y = \boxed{t_1} \texttt{ in } t_2 & \text{(spread)} & \mid \texttt{iflam}(\boxed{t_1}, t_2, t_3) & \text{(lambda test)} & & \\
& \mid \boxed{t_1}\ t_2 & \text{(application)} & \mid \texttt{case } \boxed{t_1} \texttt{ of inl}(x) \Rightarrow t_2 \mid \texttt{inr}(y) \Rightarrow t_3 & \text{(decide)} & & & &
\end{array}
$$

$$
\begin{array}{lll}
(\lambda x.F)\ a \mapsto F[x \backslash a] & \texttt{let } x := v \texttt{ in } t \mapsto t[x \backslash v] & \texttt{iflam}(\lambda x.t, t_1, t_2) \mapsto t_1 \\
\texttt{fix}(v) \mapsto v\ \texttt{fix}(v) & \texttt{let } x, y = \langle t_1, t_2 \rangle \texttt{ in } F \mapsto F[x \backslash t_1; y \backslash t_2] & \texttt{iflam}(v, t_1, t_2) \mapsto t_2, \text{ if } v \text{ is not a } \lambda\text{-term}
\end{array}
$$

$$
\texttt{case inl}(t) \texttt{ of inl}(x) \Rightarrow F \mid \texttt{inr}(y) \Rightarrow G \mapsto F[x \backslash t] \qquad \texttt{case inr}(t) \texttt{ of inl}(x) \Rightarrow F \mid \texttt{inr}(y) \Rightarrow G \mapsto G[y \backslash t]
$$

middle (LEM), which leads to a counterexample of LEM. Kripke [28, Sec.1.1] also used the unconstrained nature of free choice sequences to refute other classical results, namely Kuroda's conjecture and Markov's principle in Kreisel's FC system. [1] As it turns out, LEM is also false in a realizability theory such as The one implemented by the Nuprl proof assistant [14; 5] (a constructive type theory described in Sec 2), because it allows deciding the halting problem, which is known to be undecidable (as opposed to the above mentioned conjectures) [34, Sec.6.3]. However, a weaker version of LEM that does not require providing a realizer of either its left or right disjuncts, was proved to be consistent with Nuprl [6; 16; 25; 35, Appx.F; 34, Sec.6.3]. But using a similar technique to Brouwer's, even this weak version of LEM was shown to be inconsistent with BITT, an intuitionistic extension of Nuprl with a notion of choice sequences [10, Appx.A].

The use of the growing-over-time nature of choice sequences to refute classical axioms, and in particular LEM which is the key component of classical reasoning, seem to indicate an incompatibility between classical logic and intuitionistic logic. However, in this paper we show that this does not have to be the case. To this end, we present a relaxed model of time that mitigates the two approaches. Namely, on one hand it supports the evolving nature of choice sequences, and on the other hand it enables variants of standard classical axioms.

Concretely, we develop OpenTT, which is a novel intuitionistic type theory that incorporates choice sequences, and is inspired by BITT [10]. Those are often interpreted w.r.t. Beth-like models. Beth models were originally developed to provide meaning to intuitionistic formulas [43; 9; 20, Sec.145; 19, Sec.5.4], and they have proven especially well-suited to interpret choice sequences [17]. In such models, formulas are interpreted w.r.t. infinite trees of elements (such as numbers). They are typically formulated using a forcing interpretation where the forcing conditions are finite elements of those trees that provide meaning to choice sequences at a given point in time. By allowing access within the logic to the infinitely proceeding elements of the forcing layer, i.e., the branches of the Beth trees formulas are interpreted against, it ensues that one can then use the undecided nature of those elements to derive the negation of otherwise classically valid formulas such as LEM.

OpenTT goes beyond and departs from BITT in several ways. First, it is validated w.r.t. a novel Beth-like model, which we call the *open bar* model, that is significantly simpler than the one presented in [10]. Moreover, the notion of time induced by the model is flexible enough to capture an intuitionistic theory of choice sequences, and in particular the axiom of Open Data (a continuity axiom) that was missing from [10] and which is a key axiom of choice sequence theories, as well as weak forms of classical axioms, e.g. LEM. In other words, OpenTT together with the open bar model presented in the paper enable a more relaxed notion of time, providing a basis for a more classically-inclined intuitionistic theory.

Roadmap. We start by describing CTT (Nuprl's type theory), which is the theory OpenTT is based on (Sec. 2). We then describe the core components of OpenTT (Sec. 3), after which we present the open bar model, which we use to validate OpenTT (Sec. 4). Then, we show that OpenTT captures both a theory of choice sequences (Sec. 5), as well as a version of LEM (Sec. 6). Finally, we conclude by discussing related and future work (Sec. 7). All the results presented in this paper have been formalized in Coq, and we provide clickable hyperlinks to our formalization throughout the paper—all files are accessible from `https://github.com/vrahli/NuprlInCoq/blob/ls3/`.

2 Background

The Nuprl proof assistant implements an extensional, constructive, dependent type theory called Constructive Type Theory (CTT). This section presents some key aspects of CTT that OpenTT builds upon.

Computation system. Nuprl's programming language is an untyped, lazy λ-calculus with pairs, injections, a fixpoint operator, etc. For efficiency, integers are primitive and Nuprl provides operations on integers as well

[1] This method to refute classical axioms was reused via forcing methods (see, e.g., [18, Sec.7.2.4] for the relation between forcing and choice sequences). E.g., the independence of Markov's Principle with Martin-Löf's type theory was proven using a forcing method where the "free" nature of forcing conditions replaces the "free" nature of free choice sequences in Kripke's proof [15].

Fig. 2 Extended syntax and operational semantics

$\eta \in \texttt{CSName} ::= \langle \mathit{id}, \mathit{space} \rangle$	(C.S. name)	$v \in \texttt{Value} ::= \cdots \mid \texttt{seq}(\eta)$	
$\mathit{id} \in \texttt{String}$	(C.S. identifier)	$vt \in \texttt{Type} ::= \cdots \mid \texttt{Free}(n) \mid \wr t \mid \mathbb{N}_{\wr} \mid t_1 <_{\wr} t_2 \mid t_2 \# t_1$	
$\mathit{space} \in \mathbb{N}$	(C.S. space)	$t \in \texttt{Term} ::= \cdots \mid \texttt{if}\ \boxed{t_1}=\boxed{t_2}\ \texttt{then}\ t_3\ \texttt{else}\ t_4$	

$\texttt{if seq}(\eta_1)\texttt{=seq}(\eta_2)\ \texttt{then}\ t_1\ \texttt{else}\ t_2 \mapsto_w t_1$	, if $\eta_1 = \eta_2$	$\texttt{seq}(\eta)(i) \mapsto_w w[\eta][i]$	, if η has a i's choice in w
$\texttt{if seq}(\eta_1)\texttt{=seq}(\eta_2)\ \texttt{then}\ t_1\ \texttt{else}\ t_2 \mapsto_w t_2$	, if $\eta_1 \neq \eta_2$	$\texttt{wDepth} \mapsto_w \lvert w \rvert$	

as comparison operators. We write $\underline{i}$ for a CTT number, where i is a metatheoretical number.

Fig. 1 presents a subset of Nuprl's syntax and small-step operational semantics. We only show in it the part that is either mentioned or used in this paper. A term is either (1) a variable; (2) a canonical form, i.e., a value or an exception (see [33]); or (3) a non-canonical term. A non-canonical term t has one or two *principal arguments*—marked using boxes in Fig. 1—which are terms that have to be evaluated to canonical forms before t can be reduced further. For example, the application $f\ a$, often written as $f(a)$, diverges if f diverges. In Fig. 1 we omit rules that reduce principal arguments such as: if $t_1 \mapsto t_2$ then $t_1\ u \mapsto t_2\ u$.

In the rest of this paper, we will often write $a =_T b$ for the type $a = b \in T$, $\lambda x_1, \ldots, x_n.t$ for $\lambda x_1. \ldots \lambda x_n.t$, and $t_1 \to t_2$ for the non-dependent product type (i.e. the function type). In addition, we will use the following abstractions: $\texttt{True} = (0 = 0 \in \mathbb{Z})$, $\texttt{False} = (0 = 1 \in \mathbb{Z})$, $\neg T = (T \to \texttt{False})$, and $\mathbb{N} = \{x : \mathbb{Z} \mid \neg(x < 0)\}$.

Library. Nuprl, like other proof assistants maintains a library in which it stores all of its current definitions. A definition entry is of the form A == B, which stipulates that the expression A unfolds to B. In fact, all computation rules are implicitly dependent on the particular state of the library, see [36] for further details.

Type system. Nuprl's types are interpreted as partial equivalence relations (PERs) on closed terms [3; 4; 16]. The PER semantics can be seen as an inductive-recursive definition of: (1) an inductive relation $T_1 \equiv T_2$ that expresses type equality; (2) a recursive function $a \equiv b \in T$ that expresses equality in a type. For example, one case in the definition of $T_1 \equiv T_2$ states that (i) T_1 computes to $\mathbf{\Pi} x_1{:}A_1.B_1$; (ii) T_2 computes to $\mathbf{\Pi} x_2{:}A_2.B_2$; (iii) $A_1 \equiv A_2$; and (iv) for all closed terms t_1, t_2 such that $t_1 \equiv t_2 \in A_1$, $B_1[x_1 \backslash t_1] \equiv B_2[x_2 \backslash t_2]$. We say that a term t *inhabits* or *realizes* a type T if t is equal to itself in the PER interpretation of T, i.e., $t \equiv t \in T$. In addition, let $\texttt{inh}(T) = \exists t.\ t \equiv t \in T$. It follows from the PER interpretation of types that an equality type of the form $a = b \in T$ is true (i.e. inhabited) iff $a \equiv b \in T$ holds. [6; 32]. Note that an equality type can only be inhabited by the constant $\star$, i.e., they do not have computational content, unlike in Homotopy type theory [41].

Computational equivalence relation. Nuprl is closed under Howe's computational equivalence $\sim$, which was is a congruence [22]. In general, computing and reasoning about computation in Nuprl involves reasoning about Howe's computational equivalence relation. It is commonly used to reduce expressions by proving that terms are computationally equivalent and using the fact that $\sim$ is a congruence. For that, Nuprl provides the type $t_1 \simeq t_2$, which is the theoretical counterpart of the metatheoretical relation $t_1 \sim t_2$.

Squashing. Nuprl has a *squashing* mechanism, which we use among other things use to validate some the axioms in Sec. 5 and 6. It throws away the evidence that a type is inhabited and squashes it down to a single inhabitant using set types [14, pp.60]: $\downarrow T = \{\texttt{Unit} \mid T\}$. The only member of this type is the constant $\star$, which is $\texttt{Unit}$'s single inhabitant, and which is similar to `()` in languages such as OCaml. The constant $\star$ inhabits $\downarrow T$ if T is true/inhabited, but we do not keep the proof that it is true. See [33] for more details on squashing.

Sequents and rules. Sequents are of the form $h_1, \ldots, h_n \vdash T\ \lfloor \mathbf{ext}\ t \rfloor$. The term t is a member of the type T, which in this context is called the *extract* or *evidence* of T. Extracts are programs that are computed by the system once a proof is complete. Proof extracts are sometimes omitted when irrelevant to the discussion. An hypothesis h is of the form $x : A$, where the variable x stands for the name of the hypothesis and A its type. Such a sequent states, among other things, that T is a type and t is a member of T. A rule is a pair of a conclusion sequent S and a list of premise sequents, $S_1, \cdots, S_n$ (written as usual using a fraction notation, with the premises on top). There are several equivalent definitions for the validity of sequents [14; 16; 24; 6]. Our results are invariant to the specific semantics, thus we do not repeat them here. The sequent semantics induces a standard notion of validity of a rule, i.e., the validity of the premises entails the validity of the conclusion.

Coq formalization. CTT is formalized in Coq [6; 32; 33]. The implementation includes: (1) Nuprl's computation system; (2) Howe's congruent computational equivalence relation; (3) a definition of CTT's PER semantics; (4) definitions of Nuprl's inference rules, and their soundness proofs w.r.t. the PER semantics; and (5) a proof of Nuprl's consistency. We use this formalization here to formalize and validate OpenTT.

3 OpenTT and Choice Sequences

Choice sequences are the seminal component in Brouwer's intuitionistic theory, and the one manifesting the notion of time, and in particular, growth over time. Choice sequences are, roughly speaking, infinitely proceeding sequences of elements, which are chosen over time from a previously well-defined collection. There are two main classes of choice sequences, which are often referred to as *lawlike* and *lawless* [38]. The lawlike ones are "completed constructions" [38, Sec.1.2], where the choices must be chosen w.r.t. a pre-determined "law" (e.g., a general recursive program). The lawless ones, by contrast, are never fully completed and can always be extended over time with further choices that are not constrained by any law, that is, they can be chosen "freely" (hence the name *free choice sequences*). This is another manifestation of the fact that time is an essential component of Brouwer's intuitionistic theory because unlike lawlike sequences that are time-invariant, lawless ones keep on evolving over time. In this paper we especially focus on a theory with free choice sequences, which is a key distinguishing feature in Brouwer's intuitionistic logic.

This section describes OpenTT that extends the type theory presented in Sec. 2 to support the notion of free choice sequences. OpenTT relies on a particular notion of time, which is captured through the use of worlds. The worlds discussed in Sec. 3.2 constitute, as is standard practice, a poset, and are concretely defined as states where one stores definitions as well as choice sequences' choices. Thus, a world captures a state at a given point in time. The evolving nature of time is then captured via a notion of world extension, allowing to add new definitions, choice sequences, and choices. Fig. 2 summarizes OpenTT's extension to CTT's syntax and operational semantics, which we describe in details below.

OpenTT is inspired by BITT [10], which is also an extension of CTT with choice sequences. To make the paper self-contained we shall also review the components that are identical to those in BITT, noting the differences, which we summarize here. In addition to the standard inference rules for the types described in Sec. 2, which can be found for example in [14], OpenTT also contains inference rules that capture a theory of choice sequences. Those are described in Sec. 5. Among those, the Open Data inference rules are new compared to BITT. Furthermore, OpenTT also contains a rule for the Law of Excluded Middle (the salient principle of classical logic), described in Sec. 6. As mentioned above, this rule is not valid in BITT.

3.1 Choice sequences

As defined in Fig. 2, a choice sequence is of the form $\texttt{seq}(\eta)$, where η is a choice sequence name, i.e., a pair composed of an identifier (implemented as a string) and a space (implemented as a number). A choice sequence space $n \in \mathbb{N}$ enforces *restrictions* on the choices allowed to capture typical classes of choice sequences, such as choice sequences of numbers and choice sequences of Booleans. In particular, the space 0 constrains the choices to be numbers, 1 indicates that they must be Booleans, and any other number does not confine the choices. Choice sequences with space 0 (and similarly for 1) are therefore free choice sequences of numbers, because, except form the fact that choices must be numbers, they are not constrained further. Note that the spaces in OpenTT are simpler than in BITT for reasons discussed in Sec. 5.2. We also include a comparison operator on choice sequences, $\texttt{if } t_1\texttt{=}t_2 \texttt{ then } t_3 \texttt{ else } t_4$, which reduces to the *then* branch if t_1 and t_2 are two choice sequences with the same name, and otherwise reduces to the *else* branch.

Moreover, OpenTT includes a type $\texttt{Free}(n)$ of free choice sequences, where $n \in \mathbb{N}$ is a space used to constrain the space of its inhabitants—$\texttt{Free}(n)$ only contains sequences of the form $\texttt{seq}(\langle id, n\rangle)$. It also includes the type $t\#T$, which indicates that t is a member of T and is free from definitions and choice sequences, i.e., it is equal to a term t' in T, such that no definitions and no choice sequences occur in t', which we denote by $\texttt{noDefs}(t')$. For example $\texttt{True}\#\mathbb{U}_i$, $\texttt{False}\#\mathbb{U}_i$, and $\mathbb{N}\#\mathbb{U}_i$ are all inhabited types because $\texttt{True}$, $\texttt{False}$, and $\mathbb{N}$ are all free from definitions and choice sequences, while $(\boldsymbol{\Sigma} x{:}\texttt{Free}(0).x =_{\texttt{Free}(0)} \texttt{seq}(\eta))\#\mathbb{U}_i$ is not inhabited because this sum type mentions the choice sequence η. This is a standard hypothesis of one of the axioms for choice sequences, and so this type is used in Sec. 5.1. Note that $t\#T$ and $\texttt{noDefs}(t)$ did not appear in BITT.

3.2 Worlds

Choice sequences are recorded in a state, in which the choices of values that have been made for a particular choice sequence at a given point in time are stored. In mainstream programming languages such information can be stored, e.g., using mutable references. To enable such stateful computations in OpenTT we use its underlying digital library of definitions, which is the part of the system which is allowed to evolve over time. We call such a state a *world*.

Definition 3.1 (Worlds) *A world w is a list of entries, where an entry is either (1) a definition,*[2] *or (2) a choice sequence entry. A choice sequence entry is a pair of a choice sequence name, and a list of choices (i.e. terms).*[3] *We denote by* $\texttt{World}$ *the type of worlds.*

For example, the pair $\langle\langle id, 0\rangle, [4, 8, 15]\rangle$ is a choice sequence entry for the choice sequence named $\langle id, 0\rangle$, where 0 indicates that it must be a free choice sequence of numbers, and where $[4, 8, 15]$ is its list of choices so far.

[2] As definitions are irrelevant to the present discussion, we do not discuss them here and direct the reader to [36] instead.

[3] Our formalization also includes mechanisms to impose further restrictions on choice sequences which are not discussed here as they are irrelevant to the present discussion (see computation/library.v for further details).

Let us now introduce some necessary properties and operations on worlds.

Definition 3.2 (World properties and operations) *Let $w \in$ `World`.*

- *w is called safe, denoted* `safe`(w)*, if all the choices satisfy the corresponding restrictions (see Sec. 3.1).*
- *w is called singular, denoted* `sing`(w)*, if it does not have two entries with the same name.*
- *$|w|$ denotes the depth of w, that is the length (i.e. the number of choices) of its longest choice sequence.*

We sometimes require `sing` to prove properties such as Lem. 5.1; and the depth of worlds is used in Sec. 5.1 to approximate the modulus of continuity of a predicate at a choice sequence.

A world (or a particular snapshot of the library) can be seen as a the state of knowledge at a given point in time. It may grow over time by the addition of more definitions or choice sequence entries, or the addition of more terms to an already existing choice sequence entry. Accordingly, a world w_2 is said to extend a world w_1, essentially if it contains more entries and choices, but does not override the ones in w_1. Note that the extension relation on worlds defines a partial order on `World`.

Definition 3.3 (World extension) *A world w_2 is said to extend a w_1, denoted $w_2 \succeq w_1$, if w_2 is of the form w++w' (i.e., the list w prepended to the list w'), and if w_1 is a list of the form $[e_1, \ldots, e_n]$, then w' must be a list of the form $[e'_1, \ldots, e'_n]$ such that for all $1 \leq i \leq n$, either $e_i = e'_i$ or e_i and e'_i are choice sequence entries for the same choice sequence name such that e_i is an initial segment of e'_1.*

3.3 Computing with Worlds

In order to compute with respect to worlds, the computation relation $t_1 \mapsto t_2$ is parametertized by worlds. That is, $t_1 \mapsto_w t_2$ expresses that t_1 reduces to t_2 in one step of computation *w.r.t. the world w*. Furthermore, the application of a choice sequence `seq`(η) to a number i, i.e., `seq`$(\eta)(\underline{i})$, reduces to $w[\eta][i]$, i.e., η's i's choice recorded in w, if such a choice exists, and otherwise the computation gets stuck, i.e., `seq`$(\eta)(\underline{i})$ does not reduce.

In addition to the above world-dependent computations, we also allow computing the depth of a world w, that is, the number of choices recorded in its longest choice sequence entry. This is another addition to BITT. The nullary expression `wDepth` reduces to $|w|$ in one computation step. We use `wDepth` to realize an axiom of the theory of choice sequences in Sec. 5.1.2. It is important to note that before introducing this new computation, all computations were *time-preserved computations* in the sense that if a term t computes to a value v in a world w_1, then it will compute to a value computationally equivalent to v in any world w_2 that extends w_1. For example, for numbers, this means that if a term t computes to a number $\underline{n}$ in some world w, then it also computes to $\underline{n}$ in all extensions of w. We call such terms, *time-preserved numbers*. It is straightforward to see that `wDepth` is not a time-preserved number, as it can compute to different numbers in different extensions of a world. For example, if w_1 contains two choice sequences only: η_1 for which 3 choices have been made, and η_2 for which 4 choices have been made, then the expression `wDepth` reduces to $\underline{4}$ in w_1. Now, adding another choice to η_2 gives us a world w_2 such that $w_2 \succeq w_1$, and in which `wDepth` reduces to $\underline{5}$ instead of $\underline{4}$. However, we say that this operator is *weakly monotonic* in the sense that if it returns $\underline{k}$ in w_1, and $w_2 \succeq w_1$, then it can only return a value greater than or equal to $\underline{k}$ in w_2. The weak monotonicity of the computation system will be critical in Sec. 5.1.2 define well-formed types that depend on non-time-preserved numbers.

3.4 Time-Squashing

OpenTT inherits CTT's $\downarrow$ operator, that, as discussed in Sec. 2, allows squashing a type by discarding its inhabitants. We call this *space-squashing*, in the sense that it squashes the PER of a type to a single element, namely the constant $\star$. In addition, OpenTT also features another form of squashing, which we call *time-squashing*. As discussed in Sec. 3.3, some computations are *time-preserved*, while others, such as `wDepth`, are not. Because those two kinds of computations have different properties,[4] we wish to capture this distinction at the level of types. To this end, OpenTT includes type constructors such as the time-squashing operator $\lightning$, which given a type T, builds the type $\lightning T$, which in addition to T's members, also contains terms that behave like members of T at a particular instant of time (in a particular world). For the purpose of this paper, instead of using this general purpose time-squashing operator, we only make use of a particular form of time-squashing for numbers. Therefore, we now focus on that particular case, omitting the general construction.[5]

Accordingly, OpenTT features a $\mathbb{N}_{\lightning}$ type of non-time-preserved (or time-squashed) numbers. While $\mathbb{N}$ is required to only be inhabited by time-preserved numbers, $\mathbb{N}_{\lightning}$ is not, and allows for terms (such as `wDepth`) to compute to different numbers in different world extensions. For example, $\mathbb{N}_{\lightning}$ is allowed to be inhabited by a

[4] E.g., if t is a time-preserved number that computes to a number $\underline{m}$ less than $\underline{n}$ in a world w, then t will also be less than $\underline{n}$ in all $w' \succeq w$. However, if t is a non-time-preserved number, t might be less than $\underline{n}$ in some extensions of w, and larger in others.

[5] See `per_qtime` in per/per.v for further details on $\lightning$'s sematics.

term t that computes to $\underline{3}$ in some world w, and to $\underline{4}$ in some world w', such that $w' \succeq w$. This distinction between $\mathbb{N}$ and $\mathbb{N}_{\wr}$ will especially be useful to validate a choice sequences axiom in Sec. 5.1.2, where we make use of the depth of worlds, which, as mentioned in Sec. 3.3, is not a time-preserved computation.

In addition to the time-squashed $\mathbb{N}_{\wr}$ type, OpenTT also features the less than relation $t_1 <_{\wr} t_2$ on time-squashed numbers, whose semantics is described in Sec. 4. Although it is similar to the $t_1 < t_2$ type, one crucial difference is that, as for $\mathbb{N}_{\wr}$, it does not require t_1 and t_2 to be time-preserved. However, this type is not "well-behaved" without further restrictions (by "well-behaved" we mean monotonic is the sense of Lem. 4.10). For example, if t_1 and t_2 were allowed to compute to $\underline{3}$ and $\underline{4}$, respectively, in w_1, and to $\underline{4}$ and $\underline{3}$, respectively, in $w_2 \succeq w_1$, then $t_1 <_{\wr} t_2$ would be true in w_1 and false w_2. To avoid such "misbehaviors", we impose additional restrictions on both $\mathbb{N}_{\wr}$ and $t_1 <_{\wr} t_2$. Namely, we require that the inhabitants of $\mathbb{N}_{\wr}$, as well as t_1 and t_2, be weakly monotonic (see Sec. 3.3). This allows us to derive, among other things, that $t_1 <_{\wr} t_2$ is true in w when $t_1 \in \mathbb{N}$, $t_2 \in \mathbb{N}_{\wr}$, and t_2 computes to a number larger than t_1 in w.

4 Open Bar Model

This section describes a novel Beth-style model, called the open bar model, used below to validate OpenTT, which as mentioned above contains both a theory of choice sequences and a weak version of the classical LEM. As is standard in Beth models (or Kripke models for that matter), formulas are interpreted w.r.t. worlds.

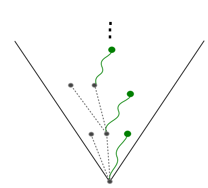
Using Beth models such as the one used in [10], a syntactic expression T is given meaning at a world w if there exists a collection B of worlds that covers all possible extensions of w, such that T corresponds to a legal type in all worlds in B. Such a collection is called *a bar* of w. In these models one has to construct such bars to prove that expressions are types or that types are inhabited. For example, to prove that choice sequences have type $\mathbb{N} \to \mathbb{N}$, given a choice sequence η and a number $\underline{n}$, one must exhibit a bar where $\texttt{seq}(\eta)(\underline{n})$ indeed computes to a number. In this paper we take a different approach, one that avoids having to build bars altogether, and only requires building individual extensions of worlds. Intuitively, instead of requiring that a property P be true at a bar of a given world w, we require that for each extension w' of w, P holds for some extension of w'. Therefore, a major distinction between traditional Beth models and our model is that in the former the semantics of a logical formula is computed based on the interpretation of that formula at a bar for the current world, while the latter only requires that in any possible extension of the current world there is always a way to find an extension where the formula is given some meaning. Thus, our model only requires exhibiting *open bars* in the sense that not all infinite extensions of the current world necessarily have a finite prefix in the bar. Therefore, open bars are derivable from "standard" bars, but the converse does not hold. For the proof that choice sequences have type $\mathbb{N} \to \mathbb{N}$, this means that given an extension w' of the current world w, one must exhibit a further extension w'' where $\texttt{seq}(\eta)(\underline{n})$ computes to a number, which can be done by constructing w'' in which η contains at least $n + 1$ choices. [6] In traditional Beth models, in addition to this construction one has to also construct the bar. Thus, the notion of open bars seems to provide a more relaxed connection between truth with constructions than in the traditional Beth-like interpretation of intuitionistic logic, where one must *construct* bars to establish validity. By not having to make the full construction, the open bar model provides some middle ground between classical and intuitionistic logic. Furthermore, note that in a traditional Beth model, depending on how the bar is defined, it is not always possible to constructively exhibit a point in the bar, whereas in the open bar model, the existence of the open bar directly gives a point at the open bar. This makes the construction of building bars from other bars generally simpler.

Let us start by introducing the concept of open bars, which is used below to interpret types.

Definition 4.1 (Open Bars) *Let w be a world and f be a (metatheoretical) predicate on worlds. We say that f is true at an* open bar *of w if:*

$$\texttt{inOpenBar}(w, f) = \forall_{\texttt{EXT}}(w, \lambda w'.\exists_{\texttt{EXT}}(w', \lambda w''.\forall_{\texttt{EXT}}(w'', f)))$$

$$\textit{where} \quad \forall_{\texttt{EXT}}(w, f) = \forall w'.\ w' \succeq w \Rightarrow f(w')$$
$$\exists_{\texttt{EXT}}(w, f) = \exists w'.\ w' \succeq w \land f(w')$$



Informally, an open bar can be thought of as an object such as the one depicted on the right. There, the large green points indicate worlds, which we already know to be at the bar, while the small gray ones indicate worlds not yet at the bar from which the open bar gives us a way to obtain worlds at the bar. For example, given the root of the tree, the open bar might give us the lowest green world w. Given a world w', such as the one left to w, where different choices have been made from w, we can ask the bar to produce another world at the bar compatible with w' (i.e., that extends w'), and we might get the middle green world.

[6] See rules/rules_choice1.v for a proof of this statement.

We now use open bars to provide meaning to OpenTT's types. We follow a similar method as the one used to validate other Nuprl-like theories [3; 4; 16; 6; 10], which we revise here to account for open bars and for OpenTT's new features. Each syntactic form denoting a type, such as $\mathbb{Z}$, $\mathbf{\Pi} x{:}A.B$, etc. (i.e., all the types introduced in Sec. 2 and Sec. 3), will be interpreted below by *type systems*, which are 4-ary relations between (1) a current world w; two types (2) T and (3) T'; and (4) a binary relation on closed terms ϕ.[7] We then write $\tau(w,T,T',\phi)$ to indicate that the two types T and T' are equal types with PER ϕ in the type system τ w.r.t. the world w. For example, $\mathbb{Z}$ is interpreted below by the `INT` operator (see Def. 4.4), and $\mathbb{N}_{\wr}$ is interpreted by the `QNAT` operator (see Def. 4.5). Note that some of those operators, such as `INT`, have an additional type system parameter, which stands for the type system defined so far. Crucially, the `OBAR` operator defined in Def. 4.9, allows giving meaning to an expression at a world w, based on its interpretation at an open bar of w.

We now follow a bottom-up presentation of OpenTT's semantics, providing the full formal definitions of the necessary operators below. Using these operators, and following the method used in [3; 4; 16; 6; 10] for example, we can then formulate a PER semantics for OpenTT as follows.

Definition 4.2 (Open Bar Semantics of OpenTT) *We define OpenTT's semantics as follows:*[8]

(i) *We first define a* `CLOSE` *operator on type systems as the smallest fixpoint such that:*
$$\mathtt{CLOSE}(\tau)(w,T,T',\phi) = \tau(w,T,T',\phi) \lor \mathtt{INT}(\tau)(w,T,T',\phi) \lor \mathtt{QNAT}(\tau)(w,T,T',\phi) \lor \mathtt{OBAR}(\tau)(w,T,T',\phi) \lor \cdots$$
which defines all the members of a universe $\mathbb{U}_i$, assuming that τ interprets all types in universes up to i.

(ii) *Using* `CLOSE`*, we then recursively define over i, where i is a number denoting a universe level, a* `UNIVi`(i) *operator that interprets OpenTT's hierarchy of universes.*

(iii) *Finally, we define the collection of all universes* `UNIV`*, the* `OPENTT` *type system, and the equality in and between* `OPENTT` *types as follows:*
$$\mathtt{UNIV} = \mathtt{OBAR}(\exists i.\ \mathtt{UNIVi}(i)) \qquad a{\equiv_w}b{\in}T = \exists\phi.\ \mathtt{OPENTT}(w,T,T,\phi) \land a\ \phi\ b$$
$$\mathtt{OPENTT} = \mathtt{CLOSE}(\mathtt{UNIV}) \qquad T_1{\equiv_w}T_2 = \exists\phi.\ \mathtt{OPENTT}(w,T_1,T_2,\phi)$$

The rest of this section defines some of these type system operators, which illustrate key aspects of the new semantics. The other operators are defined similarly, see per/per.v. Let us first define some useful abstractions.

Definition 4.3 *Let $a \Downarrow_w b$ stand for 'a computes to b w.r.t. w'. This is the reflexive and transitive closure of $\mapsto$. Moreover, let $a \Downarrow\!\!\Downarrow_w b$ stand for $\forall_{\mathtt{EXT}}(w, \lambda w'.a \Downarrow_{w'} b)$, which captures that a is time-preserved.*[9] *We use ϕ to denote binary relations on closed terms (PERs), and ψ to denote a function that map worlds to PERs. We then write $\phi_1 \sqsubseteq \phi_2$ for $\forall t,t'.\ t\ \phi_1\ t' \Rightarrow t\ \phi_2\ t'$, and $\mathtt{monPer}(w,\psi)$ for $\forall w'.\ w' \geq w \Rightarrow \psi(w) \sqsubseteq \psi(w')$. We write $\wr^w\psi$ for $\lambda t,t'.\mathtt{inOpenBar}(w,\lambda w'.t\ \psi(w')\ t')$, i.e., the lifting of the world-indexed PER ψ at an open bar of w. Finally, let $\phi_1 \iff \phi_2$ stand for the extensional equality between the two binary relations ϕ_1 and ϕ_2.*

Let us now turn to the definitions of some of the above mentioned operators under the open bar semantics. We start with demonstrating the type of integers which is in the core types of CTT. We use open bars to interpret this type as follows. Note that in this definition as in the other definitions below, through the use of $\wr^w\psi$, members of types are only required to exist at open bars of the current world.

Definition 4.4 (Integers) *The integer type is interpreted by the* `INT` *operator as follows:*
$$\mathtt{INT}(\tau)(w,T,T',\phi) = T \Downarrow\!\!\Downarrow_w \mathbb{Z} \land T' \Downarrow\!\!\Downarrow_w \mathbb{Z} \land (\phi \iff \wr^w\mathtt{INTper}) \quad \textit{where} \quad \mathtt{INTper}(w) = \lambda t,t'.\exists \underline{i}.\ t \Downarrow\!\!\Downarrow_w \underline{i} \land t' \Downarrow\!\!\Downarrow_w \underline{i}$$

Note the use of $\Downarrow\!\!\Downarrow$ above, in particular in `INTper`'s definition. As mentioned in Sec. 3.4, the reason is that we require here that such numbers are time-preserved.

As opposed to the above definition, we relax the time-preservation constraint in the next definition, where inhabitants of $\mathbb{N}_{\wr}$ are allowed to compute to different numbers in different world extensions.

Definition 4.5 (Time-Squashed Numbers) *The $\mathbb{N}_{\wr}$ type is interpreted by the* `QNAT` *operator as follows:*
$$\mathtt{QNAT}(\tau)(w,T,T',\phi) = T \Downarrow\!\!\Downarrow_w \mathbb{N}_{\wr} \land T' \Downarrow\!\!\Downarrow_w \mathbb{N}_{\wr} \land (\phi \iff \wr^w\mathtt{QNATper})$$
$$\textit{where} \quad \mathtt{QNATper}(w) = \lambda t,t'.t\ \mathtt{sameNats}(w)\ t' \ \land\ \mathtt{wMonNat}(w,t)$$
$$\mathtt{sameNats}(w) = \lambda t,t'.\forall_{\mathtt{EXT}}(w,\lambda w'.\exists k.\ t \Downarrow_w \underline{k} \land t' \Downarrow_w \underline{k})$$
$$\mathtt{wMonNat}(w,t) = \forall_{\mathtt{EXT}}(w,\lambda w_1.\forall_{\mathtt{EXT}}(w_1,\lambda w_2.\forall k_1,k_2.\ t \Downarrow_{w_1} \underline{k_1} \Rightarrow t \Downarrow_{w_2} \underline{k_2} \Rightarrow \underline{k_1} \leq \underline{k_2}))$$

[7] Instead of using induction-recursion (not yet fully supported by Coq) to define $T{\equiv_w}T'$ and $a{\equiv_w}b{\in}T$, we use Allen's method [4], and define the `OPENTT` type system in Def. 4.2 as a 4-ary relation from which we derive $T{\equiv_w}T'$ and $a{\equiv_w}b{\in}T$.

[8] See: per/per.v and per/nuprl.v for formal definitions of the operators presented here.

[9] We omit some technical details for readability. See `ccomputes_to_valc_ext`'s definition in per/per.v for the full definition.

Note that such numbers do not need to be time-preserved (see Sec. 3.3) For example, a term that computes to $\underline{3}$ in the current world w, and to $\underline{4}$ in all extensions of w (different from w), inhabits $\mathbb{N}_{\wr}$ but not $\mathbb{N}$. As it turns out, $\mathbb{N}$ is a subtype of $\mathbb{N}_{\wr}$ in the sense that all equal members of $\mathbb{N}$ are equal members of $\mathbb{N}_{\wr}$, but the other direction is not true. For example, `wDepth` is in $\mathbb{N}_{\wr}$ but not in $\mathbb{N}$. However, as mentioned in Sec. 3.3 and 3.4, we require that such numbers are weakly monotonic through the use of `wMonNat`, i.e., if t computes to a number $\underline{k_1}$ in some world w_1 and to a number $\underline{k_2}$ in an extension w_2 of w_1, then it must be that $\underline{k_1} \le \underline{k_2}$. It is straightforward to prove that this property is satisfied by `wDepth`.

As mentioned in Sec. 3.4, in addition to the $\mathbb{N}_{\wr}$ type, OpenTT also provides a 'less-than' operator on such numbers, which is interpreted as follows.

Definition 4.6 (Time-Squashed Less-Than) *The $t_1 <_{\wr} t_2$ is interpreted by the* `QLT` *operator as follows:*

$$\begin{aligned}\mathtt{QLT}(\tau)(w,T,T',\phi) = \exists t_1,t_2,t_1',t_2'.\ & T \Downarrow_w (t_1 <_{\wr} t_2) \wedge T' \Downarrow_w (t_1' <_{\wr} t_2') \\ & \wedge\ t_1\ \mathtt{QNATper}(w')\ t_1' \wedge t_2\ \mathtt{QNATper}(w')\ t_2' \wedge (\phi \iff \wr^{w}\mathtt{QLTper})\end{aligned}$$
$$\textit{where}\quad \mathtt{QLTper}(w) = \lambda t,t'.\forall_{\mathtt{EXT}}(w,\lambda w'.\exists k_1,k_2.\ t \Downarrow_w \underline{k_1} \wedge t' \Downarrow_w \underline{k_2} \wedge k_1 < k_2)$$

As mentioned in Sec. 3.1, OpenTT includes a type of choice sequences, which we interpret as follows.

Definition 4.7 (Choice Sequences) *The* `Free`(n) *type is interpreted by the* `FREE` *operator as follows:*

$$\mathtt{FREE}(\tau)(w,T,T',\phi) = \exists n.\ T \Downarrow_w \mathtt{Free}(n) \wedge T' \Downarrow_w \mathtt{Free}(n) \wedge (\phi \iff \wr^{w}(\lambda w'.\mathtt{FREEper}(w',n)))$$
$$\textit{where}\quad \mathtt{FREEper}(w',n) = \exists id.\ t \Downarrow_w \mathtt{seq}(\langle id,n\rangle) \wedge t' \Downarrow_w \mathtt{seq}(\langle id,n\rangle)$$

As mentioned in Sec. 3.1, OpenTT includes a type, namely $t\#T$ which states that a term is free from definitions and choice sequences. We interpret this type as follows.

Definition 4.8 (Free From Definitions) *The $a\#A$ type is interpreted by the* `FFD` *operator as follows:*

$$\begin{aligned}\mathtt{FFD}(\tau)(w,T,T',\phi) = \exists A,B,a,b,\psi.\ & T \Downarrow_w a\#A \wedge T' \Downarrow_w b\#B \wedge \forall_{\mathtt{EXT}}(w,\lambda w'.\tau(w',A,B,(\psi(w')))) \\ & \wedge\ \forall_{\mathtt{EXT}}(w,\lambda w'.a\ (\psi(w'))\ b) \wedge (\phi \iff \wr^{w'}(\lambda w'.\mathtt{FFDperb}(w',\psi(w'),a)))\end{aligned}$$
$$\textit{where}\quad \mathtt{FFDper}(w,\phi,a) = \lambda t,t'.\exists x.\ a\ \phi\ x \wedge \mathtt{noDefs}(x)$$

As mentioned above, the other type operators of OpenTT are interpreted in a similar fashion. Let us now turn to a crucial operator of our semantics, namely the `OBAR` operator, which allows defining types (and not just their PERs) in terms of an open bar.

Definition 4.9 (Open Bar Operator) *The* `OBAR` *constructor is defined as follows:*

$$\mathtt{OBAR}(\tau)(w,T,T',\phi) = \exists\psi.\ \mathtt{inOpenBar}(w,\lambda w'.\tau(w',T,T',(\psi(w')))) \wedge (\phi \iff \wr^{w}\psi)$$

Note that in the above definition ψ captures T's PER at the various points of the open bar.

This semantics of OpenTT satisfies the following properties, which are the standard properties expected for such a semantics [3; 4; 16; 6; 10], including the monotonocity and locality properties expected for such a possible-world semantics [43; 20; 19, Sec.5.4]—here monotonicity refers to types, and not to computations.[10]

Proposition 4.10 (Type System Properties) `OPENTT` *(see Def. 4.2) satisfies the following properties (where free variables are universally quantified):*

Uniqueness:	$\mathtt{OPENTT}(w,T,T',\phi) \Rightarrow \mathtt{OPENTT}(w,T,T',\phi') \Rightarrow (\phi \iff \phi')$
Extensionality:	$\mathtt{OPENTT}(w,T,T',\phi) \Rightarrow (\phi \iff \phi') \Rightarrow \mathtt{OPENTT}(w,T,T',\phi')$
Type transitivity:	$\mathtt{OPENTT}(w,T_1,T_2,\phi) \Rightarrow \mathtt{OPENTT}(w,T_2,T_3,\phi) \Rightarrow \mathtt{OPENTT}(w,T_1,T_3,\phi)$
Type symmetry:	$\mathtt{OPENTT}(w,T,T',\phi) \Rightarrow \mathtt{OPENTT}(w,T',T,\phi)$
Type computation:	$\mathtt{OPENTT}(w,T,T,\phi) \Rightarrow \forall_{\mathtt{EXT}}(w,\lambda w'.T \sim_{w'} T') \Rightarrow \mathtt{OPENTT}(w,T,T',\phi)$
Term transitivity:	$\mathtt{OPENTT}(w,T,T',\phi) \Rightarrow t_1\ \phi\ t_2 \Rightarrow t_2\ \phi\ t_3 \Rightarrow t_1\ \phi\ t_3$
Term symmetry:	$\mathtt{OPENTT}(w,T,T',\phi) \Rightarrow t\ \phi\ t' \Rightarrow t'\ \phi\ t$
Term computation:	$\mathtt{OPENTT}(w,T,T',\phi) \Rightarrow t\ \phi\ t \Rightarrow \forall_{\mathtt{EXT}}(w,\lambda w'.t \sim_{w'} t') \Rightarrow t\ \phi\ t'$
Monotonicity:	$\mathtt{OPENTT}(w,T,T',\phi) \Rightarrow \exists\psi.\ \forall w'.\ w' \succeq w \Rightarrow (\mathtt{OPENTT}(w',T,T',\psi(w')) \wedge \phi \sqsubseteq \psi(w') \wedge \mathtt{monPer}(w',\psi))$
Locality:	$\mathtt{inOpenBar}(w,\lambda w'.\mathtt{OPENTT}(w',T,T',\psi(w'))) \Rightarrow \mathtt{OPENTT}(w,T,T',\wr^{w}\psi)$

[10] See per/nuprl_props.v for proofs of these properties.

Finally, using these properties, it follows that OpenTT is consistent w.r.t. the open bar model.

Theorem 4.11 (Soundness & Consistency) *OpenTT's inference rules are all sound w.r.t. the open bar model, which entails that OpenTT is consistent.* [11]

5 A Theory of Choice Sequences

This section focuses on OpenTT's inference rules that provide an axiomatization of a theory of choice sequence. This theory includes two variants of the Axiom of Open Data (Sec. 5.1.1 and 5.1.2), a density axiom (Sec. 5.2), and a discreteness axiom (Sec. 5.3). These axioms are sometimes called LS3, LS1 and LS2, respectively [17]. We focus our attention on the variants of the Axiom of Open Data that captures a form of continuity which is the core essence of choice sequences, as those where not handled in BITT.

5.1 The Axiom of Open Data (AOD)

The Axiom of Open Data (AOD) is perhaps the seminal axiom in the theory of choice sequences. It is a continuity axiom that states, roughly speaking, that the validity of properties of free choice sequences (with certain side conditions) can only depend on finite initial segments of these sequences. This can be formulated as follows, where we assume that P is a predicate on free choice sequences of numbers (i.e., $P \in \texttt{Free}(0) \to \mathbb{U}_i$, for some universe i), which is free from definitions and choices sequences (i.e., $P\#(\texttt{Free}(0) \to \mathbb{U}_i)$); where $\mathbb{N}_n$ is the type $\{x : \mathbb{N} \mid x < n\}$ of natural number strictly less than n; and where $\mathcal{B}_n = \mathbb{N}_n \to \mathbb{N}$.

$$\mathbf{\Pi}\alpha{:}\texttt{Free}(0).P(\alpha) \to \mathbf{\Sigma}n{:}\mathbb{N}.\mathbf{\Pi}\beta{:}\texttt{Free}(0).(\alpha =_{\mathcal{B}_n} \beta \to P(\beta)) \qquad (\texttt{AOD})$$

Since AOD is a form of continuity principle, and the non-squashed Continuity Principle is incompatible with Nuprl [33; 34], we will only attempt to validate a squashed version of AOD. That is, because we do not have a way to compute the modulus of continuity of P at α, which is preserved over world extensions, as required by the semantics of $\mathbb{N}$, we instead validate versions of AOD where the sum type is squashed. But there are two ways in which it can be squashed, as described below.

There are two additional restrictions we impose in order to validate the squashed variants of AOD. First, to validate the claims we swap α and β in $P(\alpha)$. This has an impact on both the PER of this type, and the world w.r.t. which it is validated. Given an inhabitant t of $P(\alpha)$, we can easily build a proof of $P(\beta)$ by swapping α and β in t. This is however a metatheoretical operation. Therefore, in our variants of AOD the $P(\beta)$ is squashed. Second, note that when swapping one needs to swap α and β in all definitions and choice sequences' choices in the world w.r.t. which it is validated, leading to a different world. Therefore, we require that choice sequences cannot occur in definitions and choice sequences' choices to ensure that swapping α and β in a world w leads to an equivalent world if α and β have the same choices. To see why this is necessary take P to be the predicate $\lambda x.(x =_{\texttt{Free}(0)} \mathrm{Def}())$, and the world w to contain the definition $\mathrm{Def}() \texttt{==} \alpha$. Then, $P(\alpha)$ is equivalent to $\alpha =_{\texttt{Free}(0)} \alpha$ in this world, while $P(\beta)$ is equivalent to $\beta =_{\texttt{Free}(0)} \alpha$ in this world, which are two different types if α and β are two different choice sequences.

Before presenting and validating the variants of AOD, we present a few intermediate results. First, we prove that from $\alpha =_{\mathcal{B}_n} \beta$, we can always construct a world in which α and β contain exactly the same choices. [12]

Lemma 5.1 (Intermediate World) *Let w_1 and w_2 be two worlds such that $w_2 \succeq w_1$, $\texttt{safe}(w_1)$ and $\texttt{sing}(w_1)$ (see Def. 3.2). If η_1 and η_2 are two free choice sequences of numbers that have the same choices up to $|w_1|$ in w_2, then there must exist a world w, such that $w \succeq w_1$, $w_2 \succeq w$, both η_1 and η_2 occur in w, they have the exact same choice in w, and all these choices are numbers.*

Furthermore, we will use the following swapping operator to swap α and β in $P(\alpha)$ to obtain $P(\beta)$. [13]

Definition 5.2 (Swapping) *Let $X{\cdot}(\eta_1|\eta_2)$ be a swapping operation that swaps η_1 and η_2 everywhere in X, where X ranges over all the syntactic forms presented above.*

We can then prove that the various relations introduced in Sec. 4 are preserved by the above swapping operator. For example, crucially, we can prove that the $t_1{\equiv_w}t_2{\in}T$ relation, which expresses that t_1 and t_2 are equal members in T, is preserved by swapping. [14]

Lemma 5.3 (Swapping Preserves PERs) *If $t_1{\equiv_w}t_2{\in}T$ then $t_1{\cdot}(\eta_1|\eta_2){\equiv_{w\cdot(\eta_1|\eta_2)}}t_2{\cdot}(\eta_1|\eta_2){\in}T{\cdot}(\eta_1|\eta_2)$.*

---

[11] See rules.v and per/weak_consistency.v for more details.

[12] See Lemma `to_library_with_equal_cs` in rules/rules_choice_util4.v.

[13] See for example `swap_cs_term` in terms/swap_cs.v, which swaps two choice sequence names in a term.

[14] See Lemma `implies_equality_swap_cs` in rules/rules_choice_util4.v for a formal statement of this lemma and for its proof.

5.1.1 The Space-Squashed Axiom of Open Data ($\mathtt{AOD}_{\downarrow}$)

The first variant of the axiom we validate is the following space-squashed one, which we call $\mathtt{AOD}_{\downarrow}$.

Proposition 5.4 $\mathtt{AOD}_{\downarrow}$ *is valid w.r.t. the open bar model, i.e., the following rule of OpenTT is valid:*

$$\frac{}{H \vdash \mathbf{\Pi}\alpha{:}\mathtt{Free}(0).P(\alpha) \to \downarrow\mathbf{\Sigma}n{:}\mathbb{N}.\mathbf{\Pi}\beta{:}\mathtt{Free}(0).(\alpha =_{\mathcal{B}_n} \beta \to \downarrow P(\beta))}\ \text{[SPACE-SQUASHED-AOD]}$$

Proof. We outline here the proof. See rules/rules_ls3_v0.v for the full proof. Since the sum type is ↓-squashed, a realizer for this formula can simply be $\lambda\alpha, x.\star$. Let P be a predicate on free choice sequences of numbers, α be a free choice sequence of numbers, and instantiate n with $|w|$, the depth of the current world w. From $\alpha =_{\mathcal{B}_n} \beta$, we obtain that α and β have the same choices up to $|w|$ in the extension w' of w, and we have to derive that $P(\beta)$ is true in w'. Using Lem. 5.1 we prove that α and β have exactly the same choices in some world w'' between w and w'. Using Lem. 5.3 we swap α and β in $P(\alpha)$ and w''. Thus, thanks to the constraint that choice sequences cannot occur in definitions and choices, $P(\beta)$ is valid in a world equivalent to w'' and therefore in w'' too.[15] Finally, using monotonicity (see Lem. 4.10), we obtain that $P(\beta)$ is true in w' too. □

5.1.2 The Time-Squashed Axiom of Open Data ($\mathtt{AOD}_{\wr}$)

Next, we discuss a time-squashed version of $\mathtt{AOD}$, where instead of ↓-squashing the sum type we use the $\mathbb{N}_{\wr}$ time-squashed type of numbers introduced in Sec. 3.4, and $\mathcal{QB}_n = \{x : \mathbb{N} \mid x <_{\wr} n\} \to \mathbb{N}$ instead of $\mathcal{B}_n$.[16]

$$\mathbf{\Pi}\alpha{:}\mathtt{Free}(0).P(\alpha) \to \mathbf{\Sigma}n{:}\mathbb{N}_{\wr}.\mathbf{\Pi}\beta{:}\mathtt{Free}(0).(\alpha =_{\mathcal{QB}_n} \beta \to \downarrow P(\beta)) \qquad (\mathtt{AOD}_{\wr})$$

Note that because n is not a member of $\mathbb{N}$ anymore but of $\mathbb{N}_{\wr}$, we use $\mathcal{QB}_n$ instead of $\mathcal{B}_n$ here to state that α and β are equal sequences up to n. If $n \in \mathbb{N}_{\wr}$ then $x < n$, where $x \in \mathbb{N}$, and $\mathcal{B}_n$ are not types anymore: the semantics of $x < n$ requires both x and n to be time-preserved numbers (see Sec. 3.4). Therefore, we use $x <_{\wr} n$ here instead, which does not require numbers to be time-preserved as per its semantics presented in Def. 4.6.

Before diving into the proof of $\mathtt{AOD}_{\wr}$'s validity, we first present a few intermediate results. As mentioned above, $\mathbb{N}$ is a subset of $\mathbb{N}_{\wr}$, which implies that $t_1 <_{\wr} t_2$ is a type even when $t_1 \in \mathbb{N}$ and $t_2 \in \mathbb{N}_{\wr}$. Moreover, as mentioned in Sec. 3.4, the `wDepth` expression is a member of $\mathbb{N}_{\wr}$ (i.e., it is equal to itself in $\mathbb{N}_{\wr}$).

Lemma 5.5 *The $\mathbb{N}$ type is a subtype of $\mathbb{N}_{\wr}$, in the sense that all equal members in $\mathbb{N}$ are also equal members in $\mathbb{N}_{\wr}$, and the* `wDepth` *expression is a member of the $\mathbb{N}_{\wr}$ type.*[17] *I.e. the following rules are valid in OpenTT.*

$$\frac{H \vdash t_1 =_{\mathbb{N}} t_2}{H \vdash t_1 =_{\mathbb{N}_{\wr}} t_2}\ [\mathbb{N}\text{-SUB-}\mathbb{N}_{\wr}] \qquad \frac{}{H \vdash \mathtt{wDepth} =_{\mathbb{N}_{\wr}} \mathtt{wDepth}}\ \text{[DEPTH-IN-QNAT]}$$

For $\mathtt{AOD}_{\downarrow}$, because its $\mathbf{\Sigma}$ type is ↓-squashed, we did not have to provide a witness for the modulus of continuity of P at α. Therefore, we could simply find a suitable metatheoretical number in the proof of its validity, without having to provide an expression from the object theory that computes that number. In the metatheoretical proof, we computed the depth of the current world, which is a metatheoretical number k, and simply used $\underline{k}$, which is a number in the object theory, as an approximation of the modulus of continuity of P at α. The situation is now different in $\mathtt{AOD}_{\wr}$ because the $\mathbf{\Sigma}$ type is not ↓-squashed anymore. We now have to provide an expression from the object theory that computes that modulus of continuity. As mentioned above, we use `wDepth`, which is an expression of *OpenTT*, the object theory. This means that we now have to prove that this expression has the right type, namely, $\mathbb{N}_{\wr}$, which we proved in Lem. 5.5.

Finally, using the above results, we prove that $\mathtt{AOD}_{\wr}$ is valid w.r.t. the semantics presented in Sec. 4.

Proposition 5.6 $\mathtt{AOD}_{\wr}$ *is valid w.r.t. the open bar model, i.e., the following rule of OpenTT is valid:*

$$\frac{}{H \vdash \mathbf{\Pi}\alpha{:}\mathtt{Free}(0).P(\alpha) \to \mathbf{\Sigma}n{:}\mathbb{N}_{\wr}.\mathbf{\Pi}\beta{:}\mathtt{Free}(0).(\alpha =_{\mathcal{QB}_n} \beta \to \downarrow P(\beta))}\ \text{[TIME-SQUASHED-AOD]}$$

Proof. We here outline the proof (which is similar to that of Prop.5.4), while full details are in rules/rules_ls3_v1.v. Since now the sum type is not ↓-squashed, we have to provide a witness for it. The realizer we provide for this formula is: $\lambda\alpha, x.\langle\mathtt{wDepth}, \lambda\beta, y.\star\rangle$. Let P be a predicate on free choice sequences of numbers, and let α be a free choice sequence of numbers. We now have to prove that $\mathtt{wDepth} \in \mathbb{N}_{\wr}$, which follows from Lem. 5.5.

---

[15] See Lemma `member_swapped_css_libs` in rules/rules_choice_util4.v.

[16] Note that as in $\mathtt{AOD}_{\downarrow}$, $P(\beta)$ is also ↓-squashed here. We leave for future work to derive a version where $P(\beta)$ is not squashed.

[17] See Lemma `rule_qnat_subtype_nat_true` in rules/rules_ref.v and Lemma `rule_depth_true` in rules/rules_qnat.v.

Since `wDepth` computes to $|w|$, where w is the current world, we can then use $|w|$ as an approximation of the modulus of continuity of P at α, as in Prop. 5.4's proof. One difference with Prop. 5.4's proof is that we have here that $\alpha =_{\mathcal{QB}_n} \beta$ (which we prove to be a type using Lem. 5.5) instead of $\alpha =_{\mathcal{B}_n} \beta$. This however still suffices to show that α and β have the same choices up to $|w|$ in the extension w' of w. From here, the proof proceeds just as that of Prop. 5.4. ◻

5.2 The Density Axiom (DeA)

Another common free choice sequence axiom, sometimes called the *density* axiom [37], states that for any finite sequence of numbers f, there is a free choice sequence that contains f as initial segment (this is Axiom 2.1 in [26, Sec.2], also sometimes referred to as LS1 [17]). In BITT the following Density Axiom (DeA) was validated: $\mathbf{\Pi} n{:}\mathbb{N}.\mathbf{\Pi} f{:}\mathcal{B}_n.\mathbf{\Sigma}\alpha{:}\mathtt{Free}(0).(f =_{\mathcal{B}_n} \alpha)$ [10]. The proof of its validity was by generating an appropriate choice sequence space that contains the values of the finite sequence f as part of its name. More precisely, given a finite sequence f of n terms in $\mathbb{N}$ from the object theory, BITT includes computations to extract those n numbers, say $\underline{k_1}, \dots, \underline{k_n}$, and finally build a choice sequence name $\langle id, [k_1, \dots, k_n]\rangle$, where the space part is the sequence of metatheoretical numbers $[k_1, \dots, k_n]$, and which is used to witness DeA's sum type. In OpenTT we opted against including such spaces for two reasons. First, in the open bar model it is possible to validate a squashed version of DeA (where the sum type is squashed) without lists of numbers as choice sequence spaces. This is because the open bar model allows for internal choices to be made (see Prop. 5.7 below). Moreover, deterministically generating "fresh" choice sequence names is not preserved by swapping (which would be required for example for Lem. 5.3 to hold). Given a term t that deterministically generates η_1, it might be that swapping η_1 for η_2 turns η_1 into η_2 and leaves t unchanged, while t does not generate η_2.

Therefore, we do not include sequence number lists as possible choice sequence spaces in OpenTT and only validate the following $\downarrow$-squashed version of the Density Axiom, which we call $\mathtt{DeA}_\downarrow$.

Proposition 5.7 $\mathtt{DeA}_\downarrow$ *is valid w.r.t. the open bar model, i.e., the following rule of OpenTT is valid:*

$$\frac{}{H \vdash \mathbf{\Pi} n{:}\mathbb{N}.\mathbf{\Pi} f{:}\mathcal{B}_n.\downarrow\mathbf{\Sigma}\alpha{:}\mathtt{Free}(0).(f =_{\mathcal{B}_n} \alpha)}\ \text{[SQUASHED-DEA]}$$

Proof. To prove the validity of this axiom in some world w, assume that $n \in \mathbb{N}$ and $f \in \mathcal{B}_n$ are true in some extension w' of w. We have to exhibit an extension w'' of w' that contains a free choice sequence that has f as its initial segment. This world w'' can simply be w' augmented with a fresh (w.r.t. w') choice sequence that has f as its initial segment. [18] ◻

Note that when using the Beth model presented in [10], one has to exhibit such free choice sequences at a bar of w. If choice sequence names do not enforce an initial segment, then it could be that the choice sequence picked to witness α in the Density Axiom does not include the appropriate choices in some branches of that bar. This is why BITT features choice sequence names that enforce initial segments. As a side note, Troelstra calls the free choice sequences that can enforce an initial segment *lawless*, while he calls the ones where no initial segment is enforced *proto-lawless* [37, Sec.2.4]. Thanks to the open bar model, OpenTT is able to do without enforcing initial segments within choice sequence names while still featuring a version of the Density Axiom, at the detriment of requiring its sum type be $\downarrow$-squashed.

5.3 The Discreteness Axiom (DiA)

One final common free choice sequence axiom, sometimes called the *discreteness* axiom [31], states that equality between free choice sequences is decidable (it is Axiom 2.2 in [26, Sec.2], sometimes also referred to as LS2 [17]). As for BITT, OpenTT features intensional and extensional versions of the Discreteness Axiom (DiA), which we have proved to be valid w.r.t. the open bar model. [19] Recall that $\simeq$ denotes the theoretical counterpart of the $\sim$ metatheoretical relation, and below $\alpha \simeq \beta$ means that α and β compute to the same choice sequence.

Proposition 5.8 *The following rules of OpenTT are valid w.r.t. the open bar model:*

$$\frac{}{H \vdash \mathbf{\Pi}\alpha,\beta{:}\mathtt{Free}(0).\alpha \simeq \beta + \neg\alpha \simeq \beta}\ \text{[INT-DIA]} \qquad \frac{}{H \vdash \mathbf{\Pi}\alpha,\beta{:}\mathtt{Free}(0).\alpha =_{\mathcal{B}} \beta + \neg\alpha =_{\mathcal{B}} \beta}\ \text{[EXT-DIA]}$$

Both formulas inhabited by the term: $\lambda\alpha,\beta.$**`if`** $\alpha{=}\beta$ **`then tt else ff`**.

[18] See rules/rules_choice1.v for more details.

[19] See rules/rules_choice2.v and rules/rules_choice5.v for further details.

6 The Law of Excluded Middle

This section demonstrates that OpenTT provides a key axiom from classical logic, namely the Law of Excluded Middle (LEM). Even though various other classical principles could be considered here (and will be considered in future work), we focus on LEM as it is considered the central axiom differentiating classical logic from intuitionistic logic. Thus, we show that in addition to capturing the intuitionistic concept of choice sequences, OpenTT also includes a $\downarrow$-squashed version of LEM, that is validated w.r.t. the open bar model. For BITT, even that weak version of the LEM is inconsistent with the theory [10]. More precisely, $\neg\mathbf{\Pi}P{:}\mathbb{U}_i.\downarrow(P+\neg P)$, called $\mathtt{LEM}_\downarrow$ here, is valid w.r.t. the Beth metatheory presented in [10]. Intuitively, this is because $\mathtt{LEM}_\downarrow$'s meaning is that there exists a bar of the current world such that either: (1) P is true at the bar, or (2) it is false in all extensions of the bar. This is false (i.e., the negation is true) because, for example, for $P = (\mathbf{\Sigma}n{:}\mathbb{N}.\mathtt{seq}(\eta)(n) =_{\mathbb{N}} 1)$, where η is a free choice sequence of numbers, (1) is false because η could be the sequence that never chooses 1, and (2) is false because there is an extension of the bar where η chooses 1. Stronger versions of this axiom, such as the non-$\downarrow$-squashed version, are therefore also false. Therefore, as shown below, OpenTT is more amenable to classical logic than theories based on traditional Beth models, such as BITT. As illustrated in Prop. 6.1's proof below, intuitively, this is thanks to the fact that the open bar model implements a notion of time which allows to select futures (i.e., extensions), thereby allowing for some internal choices to be made.

Before we show that $\mathtt{LEM}_\downarrow$ is valid w.r.t. the open bar model, we point out that the counterexample provided above for BITT, namely $P = (\mathbf{\Sigma}n{:}\mathbb{N}.\mathtt{seq}(\eta)(n) =_{\mathbb{N}} 1)$, does not hold for OpenTT because given a world w it is always possible to find an extension where η eventually holds 1.

Proposition 6.1 *The following rule of OpenTT is valid w.r.t. the open bar model (using classical logic in the metatheory).*

$$\frac{}{H \vdash \mathbf{\Pi}P{:}\mathbb{U}_i.\downarrow(P+\neg P)}\ \textit{[SQUASHED-LEM]}$$

Proof. We have to show that for every world w' that extends the current world w, there exists a world w'' that extends w' such that $P+\neg P$ is inhabited in all extensions of w''. Let w' be an extension of w. We need to find a $w'' \succeq w'$ that makes the above true. Using classical logic we assume that $\exists_{\mathtt{EXT}}(w', \lambda w''.\mathtt{inh}(w'', P))$ is either true of false. [20] If it is true, we obtain an extension w'' of w' at which P is inhabited, and we therefore conclude. Otherwise, we use w', which is a trivial extension of w'. We must now show that $P+\neg P$ is inhabited in all extensions of w'. We prove that it is inhabited by $\mathtt{inr}(\star)$ by showing that in all $w'' \succeq w'$, P is not inhabited at w''. Assuming that P is inhabited at w'', we can indeed derive a contradiction to our assumption that $\exists_{\mathtt{EXT}}(w', \lambda w''.\mathtt{inh}(w'', P))$ is false: $\exists_{\mathtt{EXT}}(w', \lambda w''.\mathtt{inh}(w'', P))$ is true because P is inhabited at w''. [21] ◻

7 Conclusion and Related Work

This paper presented OpenTT, a novel intuitionistic type theory, which features both a theory of free choice sequences, and a weak form of the classical Law of Excluded Middle. This was made possible thanks to the open bar model, which internalizes a more relaxed notion of time than traditional Beth models. Thus, OpenTT provides a theoretical framework to study the interplay between intuitionistic and classical logic.

Several forms of choice sequence axioms were studied. As discussed above, some of them are currently squashed using space-squashing or time squashing-operators. We plan on exploring more general versions of these axioms that are "less squashed" in the sense that they have more computational content. In particular, we aim to investigate whether $\mathtt{AOD}_\updownarrow$'s conclusion could be unsquashed by using an object-level swapping operator, as opposed to the current metatheoretical one. In addition, we plan on investigating the compatibility of OpenTT with other standard classical principles, such as the Axiom of Choice.

As mentioned in the Introduction, there is a long line of work on providing intuitionistic counterexamples to classically valid axioms using variants of choice sequences. For example, in [15] Markov's Principle is proved to be false in a Martin-Löf type theory extended with a "generic" element, which essentially behaves as a free choice sequence of Booleans.

The open bar model gives meaning to stateful computations, and as such bears some resemblance with Kripke models, which are often used to model stateful theories. For example, in [29] the Kripke semantics of function types allows the returned values of functions to extend the state at hand. In contrast, in the open bar model all computations are allowed to extend worlds. Other examples include [1; 2] where the model makes use of worlds to interpret reference cells via step-indexing, and [12; 11] where a Kripke semantics is used to interpret a theory with reference cells in which types are interpreted by world-indexed families of logical relations. However, unlike Kripke models, Beth models can interpret formulas that only *eventually* exist.

---

[20] Note that $\mathtt{inh}(w, T)$ straightforwardly adapts $\mathtt{inh}(T)$ to take worlds into consideration.

[21] See rules/rules_classical.v for more details.

References

[1] Amal J. Ahmed, Andrew W. Appel, and Roberto Virga. "A Stratified Semantics of General References Embeddable in Higher-Order Logic". In: *LICS*. IEEE Computer Society, 2002, p. 75.

[2] Amal Jamil Ahmed. "Semantics of Types for Mutable State". PhD thesis. Princeton University, 2004.

[3] Stuart F. Allen. "A Non-Type-Theoretic Definition of Martin-Löf's Types". In: *LICS*. IEEE Computer Society, 1987, pp. 215–221.

[4] Stuart F. Allen. "A Non-Type-Theoretic Semantics for Type-Theoretic Language". PhD thesis. Cornell University, 1987.

[5] Stuart F. Allen, Mark Bickford, Robert L. Constable, Richard Eaton, Christoph Kreitz, Lori Lorigo, and Evan Moran. "Innovations in computational type theory using Nuprl". In: *J. Applied Logic* 4.4 (2006). http://www.nuprl.org/, pp. 428–469.

[6] Abhishek Anand and Vincent Rahli. "Towards a Formally Verified Proof Assistant". In: *ITP 2014*. Vol. 8558. LNCS. Springer, 2014, pp. 27–44.

[7] Mark van Atten. *On Brouwer*. Wadsworth Philosophers. Cengage Learning, 2004.

[8] Mark van Atten and Dirk van Dalen. "Arguments for the continuity principle". In: *Bulletin of Symbolic Logic* 8.3 (2002), pp. 329–347.

[9] Evert Willem Beth. *The foundations of mathematics: A study in the philosophy of science*. Harper and Row, 1966.

[10] Mark Bickford, Liron Cohen, Robert L. Constable, and Vincent Rahli. "Computability Beyond Church-Turing via Choice Sequences". In: *LICS 2018*. ACM, 2018, pp. 245–254.

[11] Lars Birkedal, Bernhard Reus, Jan Schwinghammer, Kristian Støvring, Jacob Thamsborg, and Hongseok Yang. "Step-indexed kripke models over recursive worlds". In: *POPL*. ACM, 2011, pp. 119–132.

[12] Lars Birkedal, Kristian Støvring, and Jacob Thamsborg. "Realisability semantics of parametric polymorphism, general references and recursive types". In: *Mathematical Structures in Computer Science* 20.4 (2010), pp. 655–703.

[13] Douglas Bridges and Fred Richman. *Varieties of Constructive Mathematics*. London Mathematical Society Lecture Notes Series. Cambridge University Press, 1987.

[14] Robert L. Constable, Stuart F. Allen, Mark Bromley, Rance Cleaveland, J. F. Cremer, Robert W. Harper, Douglas J. Howe, Todd B. Knoblock, Nax P. Mendler, Prakash Panangaden, James T. Sasaki, and Scott F. Smith. *Implementing mathematics with the Nuprl proof development system*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1986.

[15] Thierry Coquand and Bassel Mannaa. "The Independence of Markov's Principle in Type Theory". In: *FSCD 2016*. Vol. 52. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016, 17:1–17:18.

[16] Karl Crary. "Type-Theoretic Methodology for Practical Programming Languages". PhD thesis. Ithaca, NY: Cornell University, Aug. 1998.

[17] Dirk van Dalen. "An interpretation of intuitionistic analysis". In: *Annals of mathematical logic* 13.1 (1978), pp. 1–43.

[18] Jacques Dubucs and Michel Bourdeau. *Constructivity and Computability in Historical and Philosophical Perspective*. Vol. 34. Jan. 2014.

[19] Michael A. E. Dummett. *Elements of Intuitionism*. Second. Clarendon Press, 2000.

[20] VH Dyson and Georg Kreisel. *Analysis of Beth's semantic construction of intuitionistic logic*. Stanford University. Applied Mathematics and Statistics Laboratories, 1961.

[21] Arend Heyting. *Intuitionism: an introduction*. North-Holland Pub. Co., 1956.

[22] Douglas J. Howe. "Equality in Lazy Computation Systems". In: *LICS 1989*. IEEE Computer Society, 1989, pp. 198–203.

[23] Stephen C. Kleene and Richard E. Vesley. *The Foundations of Intuitionistic Mathematics, especially in relation to recursive functions*. North-Holland Publishing Company, 1965.

[24] Alexei Kopylov. "Type Theoretical Foundations for Data Structures, Classes, and Objects". PhD thesis. Ithaca, NY: Cornell University, 2004.

[25] Alexei Kopylov and Aleksey Nogin. "Markov's Principle for Propositional Type Theory". In: *CSL 2001*. Vol. 2142. LNCS. Springer, 2001, pp. 570–584.

[26] Georg Kreisel. "Lawless sequences of natural numbers". In: *Compositio Mathematica* 20 (1968), pp. 222–248.

[27] Georg Kreisel and Anne S. Troelstra. "Formal systems for some branches of intuitionistic analysis". In: *Annals of Mathematical Logic* 1.3 (1970), pp. 229–387.

[28] Saul A. Kripke. "Semantical Analysis of Intuitionistic Logic I". In: *Formal Systems and Recursive Functions*. Vol. 40. Studies in Logic and the Foundations of Mathematics. Elsevier, 1965, pp. 92–130.

[29] Paul Blain Levy. "Possible World Semantics for General Storage in Call-By-Value". In: *CSL 2002*. Vol. 2471. LNCS. Springer, 2002, pp. 232–246.

[30] Joan R. Moschovakis. "An intuitionistic theory of lawlike, choice and lawless sequences". In: *Logic Colloquium'90: ASL Summer Meeting in Helsinki*. Association for Symbolic Logic. 1993, pp. 191–209.

[31] Joan Rand Moschovakis. *Choice Sequences and Their Uses*. 2015.

[32] *Nuprl in Coq*. https://github.com/vrahli/NuprlInCoq.

[33] Vincent Rahli and Mark Bickford. "A nominal exploration of intuitionism". In: *CPP 2016*. Extended version: http://www.nuprl.org/html/Nuprl2Coq/continuity-long.pdf. ACM, 2016, pp. 130–141.

[34] Vincent Rahli and Mark Bickford. "Validating Brouwer's continuity principle for numbers using named exceptions". In: *Mathematical Structures in Computer Science* (2017), pp. 1–49.

[35] Vincent Rahli, Mark Bickford, and Robert L. Constable. “Bar induction: The good, the bad, and the ugly”. In: *LICS 2017*. Extended version avaible at https://vrahli.github.io/articles/bar-induction-lics-long.pdf. IEEE Computer Society, 2017, pp. 1–12.

[36] Vincent Rahli, Liron Cohen, and Mark Bickford. “A Verified Theorem Prover Backend Supported by a Monotonic Library”. In: *LPAR-22*. Vol. 57. EPiC Series in Computing. EasyChair, 2018, pp. 564–582.

[37] A. S. Troelstra. “Analysing choice sequences”. In: *J. Philosophical Logic* 12.2 (1983), pp. 197–260.

[38] A.S. Troelstra. *Choice Sequences: A Chapter of Intuitionistic Mathematics*. Clarendon Press, 1977.

[39] Anne S. Troelstra. “Choice Sequences and Informal Rigour”. In: *Synthese* 62.2 (1985), pp. 217–227.

[40] Anne S. Troelstra. *Choice sequences: a chapter of intuitionistic mathematics*. Clarendon Press Oxford, 1977.

[41] The Univalent Foundations Program. *Homotopy Type Theory: Univalent Foundations of Mathematics*. Institute for Advanced Study: http://homotopytypetheory.org/book, 2013.

[42] Wim Veldman. “Understanding and Using Brouwer’s Continuity Principle”. In: *Reuniting the Antipodes — Constructive and Nonstandard Views of the Continuum*. Vol. 306. Synthese Library. Springer Netherlands, 2001, pp. 285–302.

[43] Beth E. W. “Semantic Construction of Intuitionistic Logic”. In: *Journal of Symbolic Logic* 22.4 (1957), pp. 363–365.