

BioTT: a Family of Brouwerian Intuitionistic Theories Open to Classical Reasoning

Mark Bickford, Liron Cohen, Bob Constable, Vincent Rahli

June, 2022

Motivation

Time progressing elements are pervasive

- ▶ mutable references in programming languages
- ▶ choice sequence/forcing condition-like entities:
 - ▶ Forcing
 - ▶ Anti-classical theories: van Dalen (CT); Heyting (real analysis), Bridges & Richman (LEM), Kripke (MP); Coquand & Manna (MP); etc.

Motivation

Time progressing elements are pervasive

- ▶ mutable references in programming languages
- ▶ choice sequence/forcing condition-like entities:
 - ▶ Forcing
 - ▶ Anti-classical theories: van Dalen (CT); Heyting (real analysis), Bridges & Richman (LEM), Kripke (MP); Coquand & Manna (MP); etc.

Are choice sequences anti-classical?

Motivation

Time progressing elements are pervasive

- ▶ mutable references in programming languages
- ▶ choice sequence/forcing condition-like entities:
 - ▶ Forcing
 - ▶ Anti-classical theories: van Dalen (CT); Heyting (real analysis), Bridges & Richman (LEM), Kripke (MP); Coquand & Manna (MP); etc.

Are choice sequences anti-classical?

“Open Bar — a Brouwerian Intuitionistic Logic with a Pinch of Excluded Middle” (CSL'21): a classically inclined type theory with choice sequences

Motivation

Time progressing elements are pervasive

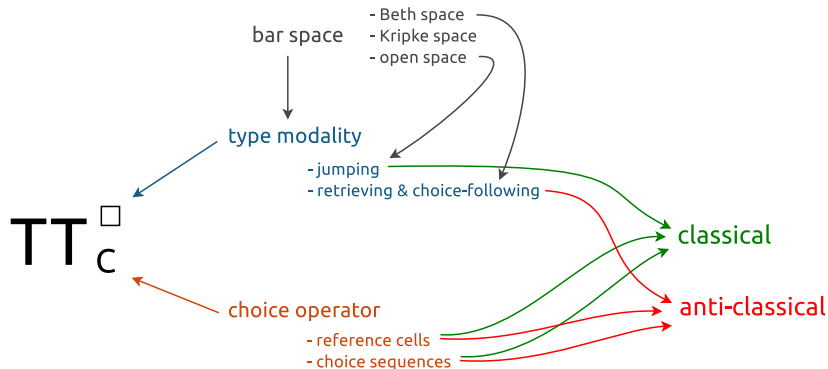
- ▶ mutable references in programming languages
- ▶ choice sequence/forcing condition-like entities:
 - ▶ Forcing
 - ▶ Anti-classical theories: van Dalen (CT); Heyting (real analysis), Bridges & Richman (LEM), Kripke (MP); Coquand & Manna (MP); etc.

Are choice sequences anti-classical?

“Open Bar — a Brouwerian Intuitionistic Logic with a Pinch of Excluded Middle” (CSL'21): a classically inclined type theory with choice sequences

Are choice sequences necessary?

This talk in 1 slide



Draws inspiration from

BITT

An anti-classical Brouwerian Intuitionistic type theory with computable choice sequences and choice sequence axioms validated by a Beth model

Formalized in Coq

OpenTT

A classically-compatible Brouwerian Intuitionistic type theory with computable choice sequences and choice sequence axioms validated by a Beth-like “*open*” model

Formalized in Coq/Agda

BITT & OpenTT: Extensional Type Theories

Untyped call-by-name
lambda-calculus

sequent calculus

realizability semantics

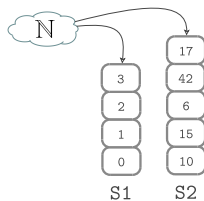
Extensional

Dependent types

BITT & OpenTT: with Choice Sequences

ETT

+

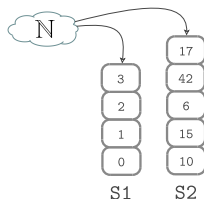


Choice Sequences

BITT & OpenTT: with Choice Sequences

ETT

+



Choice Sequences

Broader sense of computation

BITT & OpenTT: with Choice Sequences

lawless (free choice) sequences: no restrictions on the choices
(except for initial segments)

LS₁ (density) $\forall s. \exists \alpha. \alpha \in s$

LS₂ (discreteness) $\forall \alpha, \beta. (\alpha \equiv \beta \vee \neg \alpha \equiv \beta)$

LS₃ (open data) $A(\alpha) \Rightarrow \exists n. \forall \beta. (\bar{\alpha}n = \bar{\beta}n \Rightarrow A(\beta))$

s finite sequence
 α lawless sequence
 $\alpha \in s$ s is an initial segment of α
 \equiv intensional equality
 $\bar{\alpha}n$ the initial segment of α of length n

BITT & OpenTT: with Choice Sequences

BITT

LS1 $\prod n:\mathbb{N}.\prod f:\mathcal{B}_n.\sum \alpha:\text{Free}.f = \alpha \in \mathcal{B}_n$

LS2 $\prod \alpha, \beta:\text{Free}.\left(\alpha = \beta \in \mathcal{B}\right) + \left(\neg \alpha = \beta \in \mathcal{B}\right)$

LS3 –

\neg LEM $\neg \prod P:\mathbb{P}.\downarrow(P + \neg P)$

\neg MP $\neg \prod P:\mathbb{B}^{\mathbb{N}}.\neg(\prod n:\mathbb{N}.\neg P(n)) \rightarrow \sum n:\mathbb{N}.P(n)$

\neg IP $\neg \prod A:\mathbb{P}.\prod B:\mathbb{P}^{\mathbb{N}}.(A \rightarrow \sum n:\mathbb{N}.B(n))$
 $\rightarrow \sum n:\mathbb{N}.(A \rightarrow B(n))$

\neg LPO $\neg \prod P:\mathbb{B}^{\mathbb{N}}.\left(\sum n:\mathbb{N}.P(n)\right) + \left(\prod n:\mathbb{N}.\neg P(n)\right)$

OpenTT

LS1 $\prod n:\mathbb{N}.\prod f:\mathcal{B}_n.\downarrow \sum \alpha:\text{Free}.f = \alpha \in \mathcal{B}_n$

LS2 $\prod \alpha, \beta:\text{Free}.\left(\alpha = \beta \in \mathcal{B}\right) + \left(\neg \alpha = \beta \in \mathcal{B}\right)$

LS3 $\prod \alpha:\text{Free}.P(\alpha) \rightarrow$

$\sum n:\mathbb{N}.\downarrow \prod \beta:\text{Free}.\left(\alpha = \beta \in \mathcal{B}_{\downarrow n} \rightarrow \downarrow P(\beta)\right)$

LEM $\prod P:\mathbb{P}.\downarrow(P + \neg P)$

(where $\mathcal{B} = \mathbb{N}^{\mathbb{N}}$ and $\mathcal{B}_n = \mathbb{N}^{\mathbb{N}_n}$)
(\downarrow is a “proof erasure” operator)

BITT & OpenTT: Syntax

Syntax:

$$\begin{aligned} T \in \text{Type} ::= & \mathbb{N} \mid \mathbb{U}_i \mid \prod x:t.t \mid \Sigma x:t.t \mid \{x : t \mid t\} \\ & \mid t = t \in t \mid t+t \mid \dots \\ & \mid \text{Free (choice sequence type)} \end{aligned}$$
$$\begin{aligned} v \in \text{Value} ::= & T \mid \star \mid \underline{n} \mid \lambda x.t \mid \langle t, t \rangle \mid \text{inl}(t) \mid \text{inr}(t) \mid \dots \\ & \mid \eta \text{ (choice sequence name)} \end{aligned}$$
$$\begin{aligned} t \in \text{Term} ::= & x \mid v \mid t t \mid \text{fix}(t) \mid \text{let } x := t \text{ in } t \\ & \mid \text{case } t \text{ of } \text{inl}(x) \Rightarrow t \mid \text{inr}(y) \Rightarrow t \\ & \mid \text{let } x, y = t \text{ in } t \mid \text{if } t=t \text{ then } t \text{ else } t \mid \dots \end{aligned}$$

BITT & OpenTT: World-Based Computations

Operational semantics:

$$\begin{array}{lll} w \vdash (\lambda x.t_1) t_2 & \mapsto & t_1[x \setminus t_2] \\ w \vdash \text{let } x_1, x_2 = \langle t_1, t_2 \rangle \text{ in } t & \mapsto & t[x_1 \setminus t_1; x_2 \setminus t_2] \\ w \vdash \text{fix}(v) & \mapsto & v \text{ fix}(v) \end{array}$$

...

where $w \in \mathcal{W}$ (a poset)

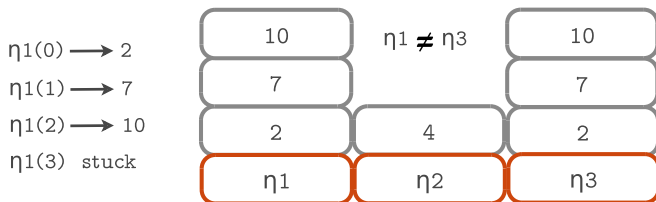
BITT & OpenTT: World-Based Computations

Operational semantics:

$$\begin{aligned} w \vdash (\lambda x. t_1) t_2 &\longmapsto t_1[x \setminus t_2] \\ w \vdash \text{let } x_1, x_2 = \langle t_1, t_2 \rangle \text{ in } t &\longmapsto t[x_1 \setminus t_1; x_2 \setminus t_2] \\ w \vdash \text{fix}(v) &\longmapsto v \text{ fix}(v) \\ \dots & \end{aligned}$$

where $w \in \mathcal{W}$ (a poset)

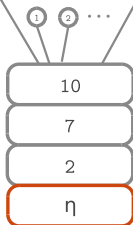
World-dependent operational semantics:



BITT vs. OpenTT

Beth model

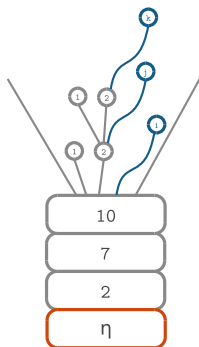
true eventually for all extensions
(at a bar)



$$w \vDash T \iff \exists b \in \text{bar}(w). \forall w_1 \in b. \forall w_2 \sqsupseteq w_1. w_2 \vDash T$$

(a modality: $\Box_B(T)$)

Open Bar model



$$w \vDash T \iff \forall w_1 \sqsupseteq w. \exists w_2 \sqsupseteq w_1. \forall w_3 \sqsupseteq w_2. w_3 \vDash T$$

(a modality: $\Box_O(T)$)

From BITT & OpenTT to TT_c^{\square}

This can all be generalized as follows...

$TT_{\mathcal{C}}^{\square}$: Choice Operator

Components

- ▶ \mathcal{N} : abstract type of choice names
- ▶ \mathcal{C} : abstract type of choices inhabited by two distinct choices κ_0 and κ_1
- ▶ a partial function: $\text{choice?} \in \mathcal{W} \rightarrow \mathcal{N} \rightarrow \mathbb{N} \rightarrow \mathcal{C}$

$TT_{\mathcal{C}}^{\square}$: Choice Operator

Components

- ▶ \mathcal{N} : abstract type of choice names
- ▶ \mathcal{C} : abstract type of choices inhabited by two distinct choices κ_0 and κ_1
- ▶ a partial function: $\text{choice?} \in \mathcal{W} \rightarrow \mathcal{N} \rightarrow \mathbb{N} \rightarrow \mathcal{C}$

Syntax

$$v \in \text{Value} ::= \dots \mid \delta \text{ (choice name)}$$

$TT_{\mathcal{C}}^{\square}$: Choice Operator

Components

- ▶ \mathcal{N} : abstract type of choice names
- ▶ \mathcal{C} : abstract type of choices inhabited by two distinct choices κ_0 and κ_1
- ▶ a partial function: $\text{choice?} \in \mathcal{W} \rightarrow \mathcal{N} \rightarrow \mathbb{N} \rightarrow \mathcal{C}$

Syntax

$v \in \text{Value} ::= \dots \mid \delta \text{ (choice name)}$

Operational Semantics

$w \vdash \delta(\underline{n}) \mapsto \text{choice?}(w, \delta, n)$

TT_C^{\square} : Modality

An abstract modality on (the semantics of) types: \square

TT_c^\square : Modality

An abstract modality on (the semantics of) types: \square

Forcing interpretation: $\square_w(w'.w' \vdash T) \rightarrow w \vDash T$

TT_c^\square : Modality

An abstract modality on (the semantics of) types: \square

Forcing interpretation: $\square_w(w'.w' \vdash T) \rightarrow w \vDash T$

Properties:

monotonicity of \square $\forall(w : \mathcal{W})(P : \mathcal{P}_w).\forall w' \sqsupseteq w.\square_w P \rightarrow \square_{w'} P$

K , distribution axiom $\forall(w : \mathcal{W})(P, Q : \mathcal{P}_w).\square_w(P \rightarrow Q) \rightarrow \square_w P \rightarrow \square_w Q$

$C4$, i.e., $\square\square \rightarrow \square$ $\forall(w : \mathcal{W})(P : \mathcal{P}_w).\square_w(w'.\square_{w'} P) \rightarrow \square_w P$

$\forall \rightarrow \square$ $\forall(w : \mathcal{W})(P : \mathcal{P}_w).\forall_w^\square(P) \rightarrow \square_w P$

T , reflexivity axiom $\forall(w : \mathcal{W})(P : \mathbb{P}).\square_w(w'.P) \rightarrow P$

TT_c^\square : Modality

An abstract modality on (the semantics of) types: \square

Forcing interpretation: $\square_w(w'.w' \vdash T) \rightarrow w \vDash T$

Properties:

monotonicity of \square $\forall(w : \mathcal{W})(P : \mathcal{P}_w).\forall w' \sqsupseteq w.\square_w P \rightarrow \square_{w'} P$

K , distribution axiom $\forall(w : \mathcal{W})(P, Q : \mathcal{P}_w).\square_w(P \rightarrow Q) \rightarrow \square_w P \rightarrow \square_w Q$

$C4$, i.e., $\square\square \rightarrow \square$ $\forall(w : \mathcal{W})(P : \mathcal{P}_w).\square_w(w'.\square_{w'} P) \rightarrow \square_w P$

$\forall \rightarrow \square$ $\forall(w : \mathcal{W})(P : \mathcal{P}_w).\forall_w^\square(P) \rightarrow \square_w P$

T , reflexivity axiom $\forall(w : \mathcal{W})(P : \mathbb{P}).\square_w(w'.P) \rightarrow P$

Enough to prove the standard properties of the type system:
consistency, symmetry, transitivity, etc.

TT_c^\square : Bar Space

Modalities can be derived from *bar spaces*

TT_c^\square : Bar Space

Modalities can be derived from *bar spaces*

Opens: $\mathcal{O} := \mathcal{W} \rightarrow \mathbb{P}$ (predicates on worlds)

$\text{TT}_{\mathcal{C}}^{\square}$: Bar Space

Modalities can be derived from *bar spaces*

Opens: $\mathcal{O} := \mathcal{W} \rightarrow \mathbb{P}$ (predicates on worlds)

Predicates on opens: $\text{BarProp} := \mathcal{W} \rightarrow \mathcal{O} \rightarrow \mathbb{P}$

TT_c^\square : Bar Space

Modalities can be derived from *bar spaces*

Opens: $\mathcal{O} := \mathcal{W} \rightarrow \mathbb{P}$ (predicates on worlds)

Predicates on opens: $\text{BarProp} := \mathcal{W} \rightarrow \mathcal{O} \rightarrow \mathbb{P}$

$B \in \text{BarProp}$ is a *bar space* if:

- ▶ it is closed under binary intersections, unions, & subsets
- ▶ it contains the top element
- ▶ its elements are non-empty

TT_c^\square : Bar Space

Modalities can be derived from *bar spaces*

Opens: $\mathcal{O} := \mathcal{W} \rightarrow \mathbb{P}$ (predicates on worlds)

Predicates on opens: $\text{BarProp} := \mathcal{W} \rightarrow \mathcal{O} \rightarrow \mathbb{P}$

$B \in \text{BarProp}$ is a *bar space* if:

- ▶ it is closed under binary intersections, unions, & subsets
- ▶ it contains the top element
- ▶ its elements are non-empty

Any bar space $B \in \text{BarProp}$ can be turned into a modality \square

$\mathbb{T}\mathbb{T}_c^\square$: Bar Space – Examples

Kripke bar space

$$\text{Kripke} := \lambda w. \lambda o. \forall_w^{\square} (w'. w' \in o)$$

TT_c^\square : Bar Space – Examples

Kripke bar space

$$\text{Kripke} := \lambda w. \lambda o. \forall_w^{\square} (w'. w' \in o)$$

Beth bar space

$$\text{Beth} := \lambda w. \lambda o. \forall (c : \text{chain}(w)). \text{barred}(o, c)$$

TT_c^\square : Bar Space – Examples

Kripke bar space

$$\text{Kripke} := \lambda w. \lambda o. \forall_w^{\square} (w'. w' \in o)$$

Beth bar space

$$\text{Beth} := \lambda w. \lambda o. \forall (c : \text{chain}(w)). \text{barred}(o, c)$$

open bar space

$$\text{Open} := \lambda w. \lambda o. \forall_w^{\square} (w_1. \exists_{w_1}^{\square} (w_2. w_2 \in o))$$

TT_C^{\square} : LEM and \neg LEM requirements

$\text{TT}_{\mathcal{C}}^{\square}$: LEM and \neg LEM requirements

- ▶ assume $w \models \prod P:\mathbb{P}.\downarrow(P+\neg P)$, prove \perp
- ▶ w : a world & δ : a **new choice** w.r.t. w
- ▶ $\Sigma\mathcal{C} := \sum k:\mathbb{N}.\delta(k)=\kappa_1 \in \text{Type}\mathcal{C}$ (an equality type)

$\text{TT}_{\mathcal{C}}^{\square}$: LEM and \neg LEM requirements

- ▶ assume $w \models \prod P:\mathbb{P}.\downarrow(P+\neg P)$, prove \perp
- ▶ w : a world & δ : a **new choice** w.r.t. w
- ▶ $\Sigma\mathcal{C} := \sum k:\mathbb{N}.\delta(k)=\kappa_1 \in \text{Type}\mathcal{C}$ (an equality type)

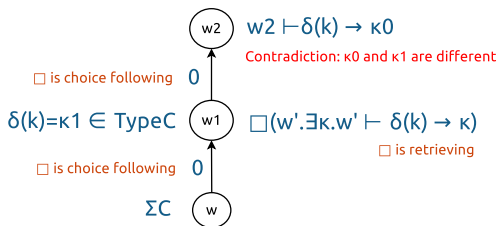
Only δ -choice so far in w is κ_0

$\text{TT}_{\mathcal{C}}^{\square}$: LEM and \neg LEM requirements

- ▶ assume $w \vDash \prod P:\mathbb{P}.\downarrow(P+\neg P)$, prove \perp
- ▶ w : a world & δ : a **new choice** w.r.t. w
- ▶ $\Sigma\mathcal{C} := \sum k:\mathbb{N}.\delta(k)=\kappa_1 \in \text{Type}\mathcal{C}$ (an equality type)

Only δ -choice so far in w is κ_0

- ▶ $\neg w \vDash \Sigma\mathcal{C}$: Assume $\Sigma\mathcal{C}$.

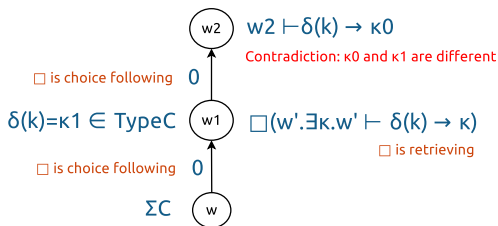


$\text{TT}_{\mathcal{C}}^{\square}$: LEM and \neg LEM requirements

- ▶ assume $w \vDash \prod P:\mathbb{P}.\downarrow(P+\neg P)$, prove \perp
- ▶ w : a world & δ : a **new choice** w.r.t. w
- ▶ $\Sigma\mathcal{C} := \sum k:\mathbb{N}.\delta(k)=\kappa_1 \in \text{Type}\mathcal{C}$ (an equality type)

Only δ -choice so far in w is κ_0

- ▶ $\neg w \vDash \Sigma\mathcal{C}$: Assume $\Sigma\mathcal{C}$.



- ▶ $\neg\forall_{w'}^{\square}(w''.\neg w'' \vDash \Sigma\mathcal{C})$: Instantiate $\forall_{w'}^{\square}(w''.\neg w'' \vDash \Sigma\mathcal{C})$ with a world w_1 where κ_1 is **immutable**, and assume $\neg w_1 \vDash \Sigma\mathcal{C}$.

Contradiction: we can also prove $w_1 \vDash \Sigma\mathcal{C}$

TT_C^{\square} : LEM and \neg LEM requirements

$TT_{\mathcal{C}}^{\square}$: LEM and \neg LEM requirements

\square is *jumping*:

$$\forall (w : \mathcal{W})(P : \mathcal{P}_w). \forall_w^{\square} (w_1. \exists_{w_1}^{\square} (w_2. \square_{w_2} P)) \rightarrow \square_w P$$

TT_C^\square : Summary

\square	C	classical?
<i>Beth</i>	<i>CS</i>	<i>X</i>
<i>Beth</i>	<i>Ref</i>	<i>X</i>
<i>open</i>	<i>CS</i>	<i>✓</i>
<i>open</i>	<i>Ref</i>	<i>✓</i>
<i>Kripke</i>	<i>CS</i>	<i>?</i>
<i>Kripke</i>	<i>Ref</i>	<i>X</i>

TT_C^\square : Summary

\square	C	classical?
<i>Beth</i>	<i>CS</i>	<i>X</i>
<i>Beth</i>	<i>Ref</i>	<i>X</i>
<i>open</i>	<i>CS</i>	✓
<i>open</i>	<i>Ref</i>	✓
<i>Kripke</i>	<i>CS</i>	?
<i>Kripke</i>	<i>Ref</i>	<i>X</i>

Beth is not *jumping*

Open is not *choice-following*

Kripke is not *retrieving* (with CS) and not *jumping*

TT_C^\square : Summary

\square	C	classical?
<i>Beth</i>	<i>CS</i>	<i>X</i>
<i>Beth</i>	<i>Ref</i>	<i>X</i>
<i>open</i>	<i>CS</i>	<i>✓</i>
<i>open</i>	<i>Ref</i>	<i>✓</i>
<i>Kripke</i>	<i>CS</i>	<i>?</i>
<i>Kripke</i>	<i>Ref</i>	<i>X</i>

Beth is not *jumping*

Open is not *choice-following*

Kripke is not *retrieving* (with CS) and not *jumping*

Choice sequences are **not necessary** (references are enough)

Choice sequences (or references) are **not anti-classical**

TT_C^{\square} : Summary

\square	C	classical?
<i>Beth</i>	<i>CS</i>	<i>X</i>
<i>Beth</i>	<i>Ref</i>	<i>X</i>
<i>open</i>	<i>CS</i>	<i>✓</i>
<i>open</i>	<i>Ref</i>	<i>✓</i>
<i>Kripke</i>	<i>CS</i>	<i>?</i>
<i>Kripke</i>	<i>Ref</i>	<i>X</i>

Beth is not *jumping*

Open is not *choice-following*

Kripke is not *retrieving* (with CS) and not *jumping*

Choice sequences are **not necessary** (references are enough)

Choice sequences (or references) are **not anti-classical**

Questions?