# Game Semantics for the Consistency of Martin-Löf Type Theory with Church's Thesis

Norihiro Yamada

yamad041@umn.edu
University of Minnesota

Mathematical Logic Seminar
University of Padova
September 21, 2020

# Table of Contents

# Introduction

# Introduction

The goal of this talk is to sketch my *proof idea* of

# Introduction

The goal of this talk is to sketch my *proof idea* of

Theorem (Consistency of MLTT with CT [Yamada, 2020])

# Introduction

The goal of this talk is to sketch my *proof idea* of

**Theorem (Consistency of MLTT with CT [Yamada, 2020])**

*Intensional Martin-Löf type theory (MLTT) is consistent with formal Church's thesis (CT).*

# Introduction

The goal of this talk is to sketch my *proof idea* of

**Theorem (Consistency of MLTT with CT [Yamada, 2020])**

*Intensional Martin-Löf type theory (MLTT) is consistent with formal Church's thesis (CT). (*1-, 0-, N-, Π-, Σ- *and* Id*-types)*

# Introduction

The goal of this talk is to sketch my *proof idea* of

**Theorem (Consistency of MLTT with CT [Yamada, 2020])**

*Intensional Martin-Löf type theory (MLTT) is consistent with formal Church's thesis (CT). (*1-, 0-, N-, Π-, Σ- *and* Id*-types)*

*MLTT* is a formal system for constructive math,

# Introduction

The goal of this talk is to sketch my *proof idea* of

**Theorem (Consistency of MLTT with CT [Yamada, 2020])**

*Intensional Martin-Löf type theory (MLTT) is consistent with formal Church's thesis (CT). (*1-, 0-, N-, Π-, Σ- *and* Id*-types)*

*MLTT* is a formal system for constructive math, and *CT* is a logical formula expressible in MLTT that states

# Introduction

The goal of this talk is to sketch my *proof idea* of

**Theorem (Consistency of MLTT with CT [Yamada, 2020])**

*Intensional Martin-Löf type theory (MLTT) is consistent with formal Church's thesis (CT). (*1-, 0-, N-, Π-, Σ- *and* Id*-types)*

*MLTT* is a formal system for constructive math, and *CT* is a logical formula expressible in MLTT that states

> *Every total function* $\mathbb{N} \to \mathbb{N}$ *is recursive, and moreover it is effective to calculate a realizer for a given total function* $\mathbb{N} \to \mathbb{N}$.

# Introduction

The goal of this talk is to sketch my *proof idea* of

**Theorem (Consistency of MLTT with CT [Yamada, 2020])**

*Intensional Martin-Löf type theory (MLTT) is consistent with formal Church's thesis (CT). (*1-, 0-, N-, Π-, Σ- *and* Id*-types)*

*MLTT* is a formal system for constructive math, and *CT* is a logical formula expressible in MLTT that states

> *Every total function* $\mathbb{N} \to \mathbb{N}$ *is recursive, and moreover it is effective to calculate a realizer for a given total function* $\mathbb{N} \to \mathbb{N}$.

CT is false in recursion theory,

# Introduction

The goal of this talk is to sketch my *proof idea* of

**Theorem (Consistency of MLTT with CT [Yamada, 2020])**

*Intensional Martin-Löf type theory (MLTT) is consistent with formal Church's thesis (CT). (*1-, 0-, N-, Π-, Σ- *and* Id*-types)*

*MLTT* is a formal system for constructive math, and *CT* is a logical formula expressible in MLTT that states

> *Every total function* $\mathbb{N} \to \mathbb{N}$ *is recursive, and moreover it is effective to calculate a realizer for a given total function* $\mathbb{N} \to \mathbb{N}$.

CT is false in recursion theory, but expected to be consistent with MLTT because objects in MLTT are all *constructive*.

# Why is the consistency problem hard?

# Why is the consistency problem hard?

However, the consistency of MLTT with CT had been open for a while [Maietti and Sambin, 2005, Ishihara et al., 2018].

# Why is the consistency problem hard?

However, the consistency of MLTT with CT had been open for a while [Maietti and Sambin, 2005, Ishihara et al., 2018].

Why is it hard to show that MLTT is consistent with CT?

# Why is the consistency problem hard?

However, the consistency of MLTT with CT had been open for a while [Maietti and Sambin, 2005, Ishihara et al., 2018].

Why is it hard to show that MLTT is consistent with CT?

Answer (in a nutshell)

# Why is the consistency problem hard?

However, the consistency of MLTT with CT had been open for a while [Maietti and Sambin, 2005, Ishihara et al., 2018].

Why is it hard to show that MLTT is consistent with CT?

**Answer (in a nutshell)**

Due to the 'dilemma between *intensionality* and *extensionality*.'

# Why is the consistency problem hard?

However, the consistency of MLTT with CT had been open for a while [Maietti and Sambin, 2005, Ishihara et al., 2018].

Why is it hard to show that MLTT is consistent with CT?

**Answer (in a nutshell)**

Due to the 'dilemma between *intensionality* and *extensionality*.'

**Answer (slightly expositive)**

# Why is the consistency problem hard?

However, the consistency of MLTT with CT had been open for a while [Maietti and Sambin, 2005, Ishihara et al., 2018].

Why is it hard to show that MLTT is consistent with CT?

**Answer (in a nutshell)**

Due to the 'dilemma between *intensionality* and *extensionality*.'

**Answer (slightly expositive)**

A standard consistency proof by *realizability* does not work since it must be intensional for CT but also extensional for MLTT.

# Why is the consistency problem hard?

However, the consistency of MLTT with CT had been open for a while [Maietti and Sambin, 2005, Ishihara et al., 2018].

Why is it hard to show that MLTT is consistent with CT?

**Answer (in a nutshell)**

Due to the 'dilemma between *intensionality* and *extensionality*.'

**Answer (slightly expositive)**

A standard consistency proof by *realizability* does not work since it must be intensional for CT but also extensional for MLTT.

1. Realizability *à la Kleene* models CT, but not MLTT;

# Why is the consistency problem hard?

However, the consistency of MLTT with CT had been open for a while [Maietti and Sambin, 2005, Ishihara et al., 2018].

Why is it hard to show that MLTT is consistent with CT?

**Answer (in a nutshell)**

Due to the 'dilemma between *intensionality* and *extensionality*.'

**Answer (slightly expositive)**

A standard consistency proof by *realizability* does not work since it must be intensional for CT but also extensional for MLTT.

1. Realizability *à la Kleene* models CT, but not MLTT;
2. Realizability *à la assemblies* models MLTT, but not CT.

# Kleene fails to model the $\xi$-rule

# Kleene fails to model the $\xi$-rule

Kleene fails to model MLTT because it does not validate the congruence rule on $\Pi$-types (a.k.a. the *$\xi$-rule*)

$$(\xi)\ \frac{\Gamma, x : A \vdash b = b' : B}{\Gamma \vdash \lambda x^A.\, b = \lambda x^A.\, b' : \Pi_{x:A} B}$$

# Kleene fails to model the $\xi$-rule

Kleene fails to model MLTT because it does not validate the congruence rule on $\Pi$-types (a.k.a. the *$\xi$-rule*)

$$(\xi)\ \frac{\Gamma, x : A \vdash b = b' : B}{\Gamma \vdash \lambda x^A.\, b = \lambda x^A.\, b' : \Pi_{x:A} B}$$

- Kleene models *closed* terms by *realizers*, validating CT.

# Kleene fails to model the $\xi$-rule

Kleene fails to model MLTT because it does not validate the congruence rule on $\Pi$-types (a.k.a. the *$\xi$-rule*)

$$(\xi)\ \frac{\Gamma, x : A \vdash b = b' : B}{\Gamma \vdash \lambda x^A.\, b = \lambda x^A.\, b' : \Pi_{x:A} B}$$

- Kleene models *closed* terms by *realizers*, validating CT.
- It takes *quotients* of realizers for *open* terms.

# Kleene fails to model the $\xi$-rule

Kleene fails to model MLTT because it does not validate the congruence rule on $\Pi$-types (a.k.a. the *$\xi$-rule*)

$$(\xi)\ \frac{\Gamma, x : A \vdash b = b' : B}{\Gamma \vdash \lambda x^A.\, b = \lambda x^A.\, b' : \Pi_{x:A} B}$$

- Kleene models *closed* terms by *realizers*, validating CT.
- It takes *quotients* of realizers for *open* terms.

$$(\text{N-Comp}) \quad x : \mathsf{N} \vdash x = x + 0 : \mathsf{N}.$$

# Kleene fails to model the $\xi$-rule

Kleene fails to model MLTT because it does not validate the congruence rule on $\Pi$-types (a.k.a. the *$\xi$-rule*)

$$(\xi)\ \frac{\Gamma, x : A \vdash b = b' : B}{\Gamma \vdash \lambda x^A.\, b = \lambda x^A.\, b' : \Pi_{x:A} B}$$

- Kleene models *closed* terms by *realizers*, validating CT.
- It takes *quotients* of realizers for *open* terms.

$$(\text{N-Comp}) \quad x : \mathsf{N} \vdash x = x + 0 : \mathsf{N}.$$

- However, the quotient makes the interpretation of open terms more extensional than their $\lambda$-abs., refuting $\xi$.

# Kleene fails to model the $\xi$-rule

Kleene fails to model MLTT because it does not validate the congruence rule on $\Pi$-types (a.k.a. the *$\xi$-rule*)

$$(\xi)\ \frac{\Gamma, x : A \vdash b = b' : B}{\Gamma \vdash \lambda x^A.\, b = \lambda x^A.\, b' : \Pi_{x:A} B}$$

- Kleene models *closed* terms by *realizers*, validating CT.
- It takes *quotients* of realizers for *open* terms.

$$(\text{N-Comp}) \quad x : \mathsf{N} \vdash x = x + 0 : \mathsf{N}.$$

- However, the quotient makes the interpretation of open terms more extensional than their $\lambda$-abs., refuting $\xi$.

$$(\xi)\ \frac{x : \mathsf{N} \vdash x = x + 0 : \mathsf{N}}{\vdash \lambda x^{\mathsf{N}}.\, x = \lambda x^{\mathsf{N}}.\, x + 0 : \mathsf{N} \Rightarrow \mathsf{N}}$$

# Why is the consistency problem hard? (reloaded)

# Why is the consistency problem hard? (reloaded)

**Remark (Realizability à la assemblies)**

If one takes quotients of realizers for closed terms too (for $\xi$), then we would recover assemblies modeling MLTT, yet unable to validate CT.

# Why is the consistency problem hard? (reloaded)

**Remark (Realizability à la assemblies)**

If one takes quotients of realizers for closed terms too (for $\xi$), then we would recover assemblies modeling MLTT, yet unable to validate CT.

We have seen why the consistency is hard to prove.

# Why is the consistency problem hard? (reloaded)

**Remark (Realizability à la assemblies)**

If one takes quotients of realizers for closed terms too (for $\xi$), then we would recover assemblies modeling MLTT, yet unable to validate CT.

We have seen why the consistency is hard to prove. To repeat:

# Why is the consistency problem hard? (reloaded)

**Remark (Realizability à la assemblies)**

If one takes quotients of realizers for closed terms too (for $\xi$), then we would recover assemblies modeling MLTT, yet unable to validate CT.

We have seen why the consistency is hard to prove. To repeat:

**Problem (in a nutshell)**

*The 'dilemma between intensionality and extensionality.'*

# Why is the consistency problem hard? (reloaded)

**Remark (Realizability à la assemblies)**

If one takes quotients of realizers for closed terms too (for $\xi$), then we would recover assemblies modeling MLTT, yet unable to validate CT.

We have seen why the consistency is hard to prove. To repeat:

**Problem (in a nutshell)**

*The 'dilemma between intensionality and extensionality.'*

**Problem (slightly expositive)**

*Realizability can only be either intensional or extensional, but the consistency problem requires it to be both. Impossible!*

# Main idea of the consistency proof: games with oracles

# Main idea of the consistency proof: games with oracles

How can we prove the consistency of MLTT plus CT then?

# Main idea of the consistency proof: games with oracles

How can we prove the consistency of MLTT plus CT then?

## Main idea of the consistency proof

# Main idea of the consistency proof: games with oracles

How can we prove the consistency of MLTT plus CT then?

## Main idea of the consistency proof

To equip recursive maps (or mor. in assemblies) with 'computations with *an oracle*,' a kind of *games*.

# Main idea of the consistency proof: games with oracles

How can we prove the consistency of MLTT plus CT then?

## Main idea of the consistency proof

To equip recursive maps (or mor. in assemblies) with 'computations with *an oracle*,' a kind of *games*. This idea leads to *game semantics*.

# Main idea of the consistency proof: games with oracles

How can we prove the consistency of MLTT plus CT then?

**Main idea of the consistency proof**

To equip recursive maps (or mor. in assemblies) with 'computations with *an oracle*,' a kind of *games*. This idea leads to *game semantics*.

I.e., to lift recursive maps in such a way that they additionally calculate a realizer for an output on the assumption that an oracle provides a specific realizer for each input.

# Main idea of the consistency proof: games with oracles

How can we prove the consistency of MLTT plus CT then?

**Main idea of the consistency proof**

To equip recursive maps (or mor. in assemblies) with 'computations with *an oracle*,' a kind of *games*. This idea leads to *game semantics*.

I.e., to lift recursive maps in such a way that they additionally calculate a realizer for an output on the assumption that an oracle provides a specific realizer for each input.

**Remark (Why the idea would work)**

# Main idea of the consistency proof: games with oracles

How can we prove the consistency of MLTT plus CT then?

**Main idea of the consistency proof**

To equip recursive maps (or mor. in assemblies) with 'computations with *an oracle*,' a kind of *games*. This idea leads to *game semantics*.

I.e., to lift recursive maps in such a way that they additionally calculate a realizer for an output on the assumption that an oracle provides a specific realizer for each input.

**Remark (Why the idea would work)**

- For CT, the calculation of realizers in this game semantics would be 'copy-cat,' like Kleene, trivially validating CT.

# Main idea of the consistency proof: games with oracles

How can we prove the consistency of MLTT plus CT then?

**Main idea of the consistency proof**

To equip recursive maps (or mor. in assemblies) with 'computations with *an oracle*,' a kind of *games*. This idea leads to *game semantics*.

I.e., to lift recursive maps in such a way that they additionally calculate a realizer for an output on the assumption that an oracle provides a specific realizer for each input.

**Remark (Why the idea would work)**

- For CT, the calculation of realizers in this game semantics would be 'copy-cat,' like Kleene, trivially validating CT.
- It would be as extensional as the existing game semantics (n.b., realizers not assigned to mor.), validating $\xi$ (and modeling MLTT).

# Game semantics: a general introduction

# Game semantics: a general introduction

*Game semantics* refers to a particular kind of mathematical (or *denotational*) semantics of type theories that models types and terms by *games* and *strategies*, respectively.

# Game semantics: a general introduction

*Game semantics* refers to a particular kind of mathematical (or *denotational*) semantics of type theories that models types and terms by *games* and *strategies*, respectively.

- Its modern, *categorical* form traces back to *fully complete* game semantics of (some fragments of) linear logic;

# Game semantics: a general introduction

*Game semantics* refers to a particular kind of mathematical (or *denotational*) semantics of type theories that models types and terms by *games* and *strategies*, respectively.

- Its modern, *categorical* form traces back to *fully complete* game semantics of (some fragments of) linear logic;
- Known in TCS by solving *full abstraction of PCF*;

# Game semantics: a general introduction

*Game semantics* refers to a particular kind of mathematical (or *denotational*) semantics of type theories that models types and terms by *games* and *strategies*, respectively.

- Its modern, *categorical* form traces back to *fully complete* game semantics of (some fragments of) linear logic;
- Known in TCS by solving *full abstraction of PCF*;
- Today, various full abstraction results with applications in *model checking* and *program verification*;

# Game semantics: a general introduction

*Game semantics* refers to a particular kind of mathematical (or *denotational*) semantics of type theories that models types and terms by *games* and *strategies*, respectively.

- Its modern, *categorical* form traces back to *fully complete* game semantics of (some fragments of) linear logic;
- Known in TCS by solving *full abstraction of PCF*;
- Today, various full abstraction results with applications in *model checking* and *program verification*;
- Models *effects* systematically;

# Game semantics: a general introduction

*Game semantics* refers to a particular kind of mathematical (or *denotational*) semantics of type theories that models types and terms by *games* and *strategies*, respectively.

- Its modern, *categorical* form traces back to *fully complete* game semantics of (some fragments of) linear logic;
- Known in TCS by solving *full abstraction of PCF*;
- Today, various full abstraction results with applications in *model checking* and *program verification*;
- Models *effects* systematically;
- New directions: I have applied game semantics to recursion theory, constructive math, category theory, categorical logic, etc.

# Game semantics: asymmetry between players

# Game semantics: asymmetry between players

A distinguishing feature of game semantics is its interpretation in terms of interactions between *Player (P)* and *Opponent (O)*.

# Game semantics: asymmetry between players

A distinguishing feature of game semantics is its interpretation in terms of interactions between *Player (P)* and *Opponent (O)*.

Main idea (rephrased)

# Game semantics: asymmetry between players

A distinguishing feature of game semantics is its interpretation in terms of interactions between *Player (P)* and *Opponent (O)*.

## Main idea (rephrased)

To utilize the distinction between P and O in game semantics in such a way that P (resp. O) plays the role of *effective maps* (resp. *oracles*).

# Game semantics: asymmetry between players

A distinguishing feature of game semantics is its interpretation in terms of interactions between *Player (P)* and *Opponent (O)*.

**Main idea (rephrased)**

To utilize the distinction between P and O in game semantics in such a way that P (resp. O) plays the role of *effective maps* (resp. *oracles*).

Roughly, P = extensional (for $\xi$), while O = intensional (for CT).

# Game semantics: asymmetry between players

A distinguishing feature of game semantics is its interpretation in terms of interactions between *Player (P)* and *Opponent (O)*.

**Main idea (rephrased)**

To utilize the distinction between P and O in game semantics in such a way that P (resp. O) plays the role of *effective maps* (resp. *oracles*).

Roughly, P = extensional (for $\xi$), while O = intensional (for CT).
This idea does not work for realizability since it has only one entity for terms, i.e., realizers or their quotients, either intensional or extensional.

# Game semantics: asymmetry between players

A distinguishing feature of game semantics is its interpretation in terms of interactions between *Player (P)* and *Opponent (O)*.

**Main idea (rephrased)**

To utilize the distinction between P and O in game semantics in such a way that P (resp. O) plays the role of *effective maps* (resp. *oracles*).

Roughly, P = extensional (for $\xi$), while O = intensional (for CT).
This idea does not work for realizability since it has only one entity for terms, i.e., realizers or their quotients, either intensional or extensional.

**Remark (Not quite the conventional game semantics)**

# Game semantics: asymmetry between players

A distinguishing feature of game semantics is its interpretation in terms of interactions between *Player (P)* and *Opponent (O)*.

## Main idea (rephrased)

To utilize the distinction between P and O in game semantics in such a way that P (resp. O) plays the role of *effective maps* (resp. *oracles*).

Roughly, P = extensional (for $\xi$), while O = intensional (for CT).
This idea does not work for realizability since it has only one entity for terms, i.e., realizers or their quotients, either intensional or extensional.

## Remark (Not quite the conventional game semantics)

The present work enlarges the asymmetry between P and O in standard games, and so it is novel from the game-semantic point of view as well.

# Game semantics: the variant of games and strategies

# Game semantics: the variant of games and strategies

Specifically, my model of MLTT + CT for the consistency proof is a modification of the game semantics of MLTT [Yamada, 2016],

# Game semantics: the variant of games and strategies

Specifically, my model of MLTT + CT for the consistency proof is a modification of the game semantics of MLTT [Yamada, 2016], which is in turn based on the standard game semantics of simple type theories [Abramsky and McCusker, 1999].

# Game semantics: the variant of games and strategies

Specifically, my model of MLTT + CT for the consistency proof is a modification of the game semantics of MLTT [Yamada, 2016], which is in turn based on the standard game semantics of simple type theories [Abramsky and McCusker, 1999].

Remark (Technical simplification)

# Game semantics: the variant of games and strategies

Specifically, my model of MLTT + CT for the consistency proof is a modification of the game semantics of MLTT [Yamada, 2016], which is in turn based on the standard game semantics of simple type theories [Abramsky and McCusker, 1999].

**Remark (Technical simplification)**

Because the main part of CT has nothing to do with dependent types, we work on [Abramsky and McCusker, 1999] in this talk.

# Game semantics: the variant of games and strategies

Specifically, my model of MLTT + CT for the consistency proof is a modification of the game semantics of MLTT [Yamada, 2016], which is in turn based on the standard game semantics of simple type theories [Abramsky and McCusker, 1999].

**Remark (Technical simplification)**

Because the main part of CT has nothing to do with dependent types, we work on [Abramsky and McCusker, 1999] in this talk.

I.e., we focus on how to model the term $\mathsf{ct} : (\mathsf{N} \Rightarrow \mathsf{N}) \Rightarrow \mathsf{N}$ that outputs a realizer $\mathsf{e} : \mathsf{N}$ for a given input $\mathsf{f} : \mathsf{N} \Rightarrow \mathsf{N}$.

# Games (moderately simplified)

# Games (moderately simplified)

Definition (Games [Abramsky and McCusker, 1999])

# Games (moderately simplified)

**Definition (Games [Abramsky and McCusker, 1999])**

A ***move*** is a finite sequence of natural numbers with the O/P parity embedded.

# Games (moderately simplified)

**Definition (Games [Abramsky and McCusker, 1999])**

A ***move*** is a finite sequence of natural numbers with the O/P parity embedded. A ***game*** is a rooted, directed forest s.t. nodes are moves, and each path from a root has the parity OPOP...

# Games (moderately simplified)

**Definition (Games [Abramsky and McCusker, 1999])**

A ***move*** is a finite sequence of natural numbers with the O/P parity embedded. A ***game*** is a rooted, directed forest s.t. nodes are moves, and each path from a root has the parity OPOP...

We call a path from a root in a game $G$ a ***position*** in $G$.

# Games (moderately simplified)

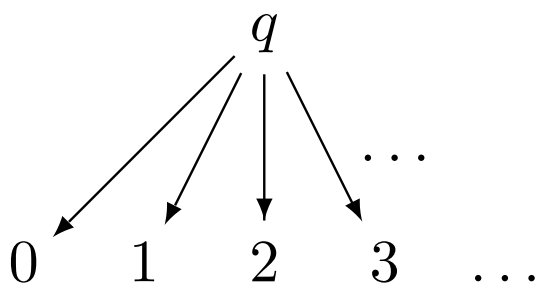
**Definition (Games [Abramsky and McCusker, 1999])**

A ***move*** is a finite sequence of natural numbers with the O/P parity embedded. A ***game*** is a rooted, directed forest s.t. nodes are moves, and each path from a root has the parity OPOP...

We call a path from a root in a game $G$ a ***position*** in $G$.

**Example (Games)**

The game $N$ of natural numbers and the game $\mathbf{0}$ of falsity are




$q$

# Implication between games (part I)

# Implication between games (part I)

Definition (Implication [Abramsky and McCusker, 1999])

# Implication between games (part I)

**Definition (Implication [Abramsky and McCusker, 1999])**

The ***implication*** $A \Rightarrow B$ between games $A$ and $B$ consists of positions $\boldsymbol{s}$ that are 'interleaving mixtures' of positions $\boldsymbol{t}$ in $A$ and $\boldsymbol{u}$ in $B$ (n.b., we take the disjoint union of $A$ and $B$) s.t.

# Implication between games (part I)

**Definition (Implication [Abramsky and McCusker, 1999])**

The ***implication*** $A \Rightarrow B$ between games $A$ and $B$ consists of positions $\boldsymbol{s}$ that are 'interleaving mixtures' of positions $\boldsymbol{t}$ in $A$ and $\boldsymbol{u}$ in $B$ (n.b., we take the disjoint union of $A$ and $B$) s.t.

1. The first move of $\boldsymbol{s}$ is that of $\boldsymbol{u}$ ($B$);

# Implication between games (part I)

**Definition (Implication [Abramsky and McCusker, 1999])**

The ***implication*** $A \Rightarrow B$ between games $A$ and $B$ consists of positions $\boldsymbol{s}$ that are 'interleaving mixtures' of positions $\boldsymbol{t}$ in $A$ and $\boldsymbol{u}$ in $B$ (n.b., we take the disjoint union of $A$ and $B$) s.t.

1. The first move of $\boldsymbol{s}$ is that of $\boldsymbol{u}$ ($B$);
2. A switch between $\boldsymbol{t}$ ($A$) and $\boldsymbol{u}$ ($B$) in $\boldsymbol{s}$ is always made by P, where the O/P parity on $\boldsymbol{t}$ ($A$) is *flipped*.

# Implication between games (part I)

**Definition (Implication [Abramsky and McCusker, 1999])**

The ***implication*** $A \Rightarrow B$ between games $A$ and $B$ consists of positions $\boldsymbol{s}$ that are 'interleaving mixtures' of positions $\boldsymbol{t}$ in $A$ and $\boldsymbol{u}$ in $B$ (n.b., we take the disjoint union of $A$ and $B$) s.t.

1. The first move of $\boldsymbol{s}$ is that of $\boldsymbol{u}$ ($B$);
2. A switch between $\boldsymbol{t}$ ($A$) and $\boldsymbol{u}$ ($B$) in $\boldsymbol{s}$ is always made by P, where the O/P parity on $\boldsymbol{t}$ ($A$) is *flipped*.

**Remark (Linear implication)**

Strictly speaking, the construction $\Rightarrow$ models *linear implication*. However, this simplification is not harmful to the rest of the talk.

# Implication between games (part II)

# Implication between games (part II)

**Example (Implications)**

Typical plays in implications are

# Implication between games (part II)

## Example (Implications)

Typical plays in implications are

$$\begin{array}{ccc} N & \Rightarrow & N \\ \hline & & q(\mathrm{O}) \\ q(\mathrm{P}) & & \\ n(\mathrm{O}) & & \\ & & n'(\mathrm{P}) \end{array}$$

# Implication between games (part II)

## Example (Implications)

Typical plays in implications are

$$\frac{N \quad \Rightarrow \quad N}{q(\mathrm{O})}$$

$q(\mathrm{P})$
$n(\mathrm{O})$
$n'(\mathrm{P})$

$$\frac{(N \quad \Rightarrow \quad N) \quad \Rightarrow \quad N}{q(\mathrm{O})}$$

$q(\mathrm{P})$
$q(\mathrm{O})$
$m(\mathrm{P})$
$m'(\mathrm{O})$
$m''(\mathrm{P})$

# Implication between games (part II)

## Example (Implications)

Typical plays in implications are

$$\begin{array}{ccc} N & \Rightarrow & N \\ \hline & & q(\mathrm{O}) \\ q(\mathrm{P}) & & \\ n(\mathrm{O}) & & \\ & & n'(\mathrm{P}) \end{array} \qquad \begin{array}{ccccc} (N & \Rightarrow & N) & \Rightarrow & N \\ \hline & & & & q(\mathrm{O}) \\ & & q(\mathrm{P}) & & \\ q(\mathrm{O}) & & & & \\ m(\mathrm{P}) & & & & \\ & & m'(\mathrm{O}) & & \\ & & & & m''(\mathrm{P}) \end{array}$$

## Remark

O plays the role of P in the domain $A$ of each implication $A \Rightarrow B$.

# Strategies (moderately simplified)

# Strategies (moderately simplified)

Definition (Strategies [Abramsky and McCusker, 1999])

# Strategies (moderately simplified)

**Definition (Strategies [Abramsky and McCusker, 1999])**

A ***strategy*** $\sigma$ on a game $G$, written $\sigma : G$, is a partial map

$$\sigma : \text{odd-length position } m_1 m_2 \dots m_{2i+1} \text{ in } G \mapsto \text{P-move } m_{2i+2} \text{ in } G$$

s.t. the concatenation $m_1 m_2 \dots m_{2i+1} m_{2i+2}$ is a position in $G$.

# Strategies (moderately simplified)

**Definition (Strategies [Abramsky and McCusker, 1999])**

A ***strategy*** $\sigma$ on a game $G$, written $\sigma : G$, is a partial map

$$\sigma : \text{odd-length position } m_1 m_2 \dots m_{2i+1} \text{ in } G \mapsto \text{P-move } m_{2i+2} \text{ in } G$$

s.t. the concatenation $m_1 m_2 \dots m_{2i+1} m_{2i+2}$ is a position in $G$.

Hence, a strategy $\sigma$ on a game $G$ tells P 'how to play in $G$.'

# Strategies (moderately simplified)

**Definition (Strategies [Abramsky and McCusker, 1999])**

A ***strategy*** $\sigma$ on a game $G$, written $\sigma : G$, is a partial map

$$\sigma : \text{odd-length position } m_1 m_2 \dots m_{2i+1} \text{ in } G \mapsto \text{P-move } m_{2i+2} \text{ in } G$$

s.t. the concatenation $m_1 m_2 \dots m_{2i+1} m_{2i+2}$ is a position in $G$.

Hence, a strategy $\sigma$ on a game $G$ tells P 'how to play in $G$.'
***Total*** (resp. ***recursive***) strategies refer to the obvious ones.

# Strategies (moderately simplified)

**Definition (Strategies [Abramsky and McCusker, 1999])**

A ***strategy*** $\sigma$ on a game $G$, written $\sigma : G$, is a partial map

$$\sigma : \text{odd-length position } m_1 m_2 \dots m_{2i+1} \text{ in } G \mapsto \text{P-move } m_{2i+2} \text{ in } G$$

s.t. the concatenation $m_1 m_2 \dots m_{2i+1} m_{2i+2}$ is a position in $G$.

Hence, a strategy $\sigma$ on a game $G$ tells P 'how to play in $G$.'
***Total*** (resp. ***recursive***) strategies refer to the obvious ones.

**Example (Strategies)**

# Strategies (moderately simplified)

**Definition (Strategies [Abramsky and McCusker, 1999])**

A ***strategy*** $\sigma$ on a game $G$, written $\sigma : G$, is a partial map

$$\sigma : \text{odd-length position } m_1 m_2 \dots m_{2i+1} \text{ in } G \mapsto \text{P-move } m_{2i+2} \text{ in } G$$

s.t. the concatenation $m_1 m_2 \dots m_{2i+1} m_{2i+2}$ is a position in $G$.

Hence, a strategy $\sigma$ on a game $G$ tells P 'how to play in $G$.'
***Total*** (resp. ***recursive***) strategies refer to the obvious ones.

**Example (Strategies)**

The strategy succ : $N \Rightarrow N$ plays as the successor $\mathbb{N} \to \mathbb{N}$ by succ : $q \mapsto q, qqn \mapsto qqn(n+1)$, which is total and recursive.

# Strategies (moderately simplified)

**Definition (Strategies [Abramsky and McCusker, 1999])**

A ***strategy*** $\sigma$ on a game $G$, written $\sigma : G$, is a partial map

$$\sigma : \text{odd-length position } m_1 m_2 \dots m_{2i+1} \text{ in } G \mapsto \text{P-move } m_{2i+2} \text{ in } G$$

s.t. the concatenation $m_1 m_2 \dots m_{2i+1} m_{2i+2}$ is a position in $G$.

Hence, a strategy $\sigma$ on a game $G$ tells P 'how to play in $G$.'
***Total*** (resp. ***recursive***) strategies refer to the obvious ones.

**Example (Strategies)**

The strategy succ : $N \Rightarrow N$ plays as the successor $\mathbb{N} \to \mathbb{N}$ by succ : $q \mapsto q, qqn \mapsto qqn(n+1)$, which is total and recursive.
On the other hand, there is no total strategy on the game $\mathbf{0}$.

# Conventional game semantics does not work

# Conventional game semantics does not work

Finally, since totality is not preserved under composition of strategies, we must lift totality to stronger *winning* (details omitted).

# Conventional game semantics does not work

Finally, since totality is not preserved under composition of strategies, we must lift totality to stronger *winning* (details omitted). We have recalled conventional games and winning, recursive strategies.

# Conventional game semantics does not work

Finally, since totality is not preserved under composition of strategies, we must lift totality to stronger *winning* (details omitted). We have recalled conventional games and winning, recursive strategies. However, this conventional variant cannot validate CT because

# Conventional game semantics does not work

Finally, since totality is not preserved under composition of strategies, we must lift totality to stronger *winning* (details omitted). We have recalled conventional games and winning, recursive strategies. However, this conventional variant cannot validate CT because

**Problem 1**

O may play in an implication $A \Rightarrow B$ as a *non-recursive* strategy $\alpha$ on the domain $A$ (so that there is no realizer for $\alpha$).

# Conventional game semantics does not work

Finally, since totality is not preserved under composition of strategies, we must lift totality to stronger *winning* (details omitted). We have recalled conventional games and winning, recursive strategies. However, this conventional variant cannot validate CT because

**Problem 1**

O may play in an implication $A \Rightarrow B$ as a *non-recursive* strategy $\alpha$ on the domain $A$ (so that there is no realizer for $\alpha$).

Even if we restrict O's plays in the domain $A$ to recursive ones,

# Conventional game semantics does not work

Finally, since totality is not preserved under composition of strategies, we must lift totality to stronger *winning* (details omitted). We have recalled conventional games and winning, recursive strategies. However, this conventional variant cannot validate CT because

**Problem 1**

O may play in an implication $A \Rightarrow B$ as a *non-recursive* strategy $\alpha$ on the domain $A$ (so that there is no realizer for $\alpha$).

Even if we restrict O's plays in the domain $A$ to recursive ones,

**Problem 2**

There is no effective way to calculate a realizer for a given strategy $\alpha$ on the domain $A$ from a *finite* interaction with $\alpha$.

# Sketch of the consistency proof (part I)

# Sketch of the consistency proof (part I)

To overcome these two problems, we have arrived at

# Sketch of the consistency proof (part I)

To overcome these two problems, we have arrived at

Definition (Modified implication [Yamada, 2020])

# Sketch of the consistency proof (part I)

To overcome these two problems, we have arrived at

**Definition (Modified implication [Yamada, 2020])**

A ***modified implication*** $A \multimap B$ between games $A$ and $B$ consists of positions $\boldsymbol{s}_{[e]}$ s.t.

# Sketch of the consistency proof (part I)

To overcome these two problems, we have arrived at

**Definition (Modified implication [Yamada, 2020])**

A ***modified implication*** $A \multimap B$ between games $A$ and $B$ consists of positions $\boldsymbol{s}_{[e]}$ s.t.

1. $\boldsymbol{s}_{[e]}$ is a position $\boldsymbol{s}$ in $A \Rightarrow B$ whose moves are equipped with a realizer $e \in \mathbb{N}$ for a winning, recursive strategy $\alpha : A$;

# Sketch of the consistency proof (part I)

To overcome these two problems, we have arrived at

**Definition (Modified implication [Yamada, 2020])**

A ***modified implication*** $A \mathrel{-\!\!\triangleright} B$ between games $A$ and $B$ consists of positions $\boldsymbol{s}_{[e]}$ s.t.

1. $\boldsymbol{s}_{[e]}$ is a position $\boldsymbol{s}$ in $A \Rightarrow B$ whose moves are equipped with a realizer $e \in \mathbb{N}$ for a winning, recursive strategy $\alpha : A$;
2. O's computation in $\boldsymbol{s}$ for the domain $A$ matches $\alpha$.

# Sketch of the consistency proof (part I)

To overcome these two problems, we have arrived at

**Definition (Modified implication [Yamada, 2020])**

A ***modified implication*** $A \multimap B$ between games $A$ and $B$ consists of positions $\boldsymbol{s}_{[e]}$ s.t.

1. $\boldsymbol{s}_{[e]}$ is a position $\boldsymbol{s}$ in $A \Rightarrow B$ whose moves are equipped with a realizer $e \in \mathbb{N}$ for a winning, recursive strategy $\alpha : A$;
2. O's computation in $\boldsymbol{s}$ for the domain $A$ matches $\alpha$.

This new construction is the core of my proof of

# Sketch of the consistency proof (part I)

To overcome these two problems, we have arrived at

**Definition (Modified implication [Yamada, 2020])**

A ***modified implication*** $A \mathrel{-\!\!\triangleright} B$ between games $A$ and $B$ consists of positions $\boldsymbol{s}_{[e]}$ s.t.

1. $\boldsymbol{s}_{[e]}$ is a position $\boldsymbol{s}$ in $A \Rightarrow B$ whose moves are equipped with a realizer $e \in \mathbb{N}$ for a winning, recursive strategy $\alpha : A$;
2. O's computation in $\boldsymbol{s}$ for the domain $A$ matches $\alpha$.

This new construction is the core of my proof of

**Theorem (Consistency of MLTT + CT)**

*MLTT is consistent with CT.*

# Sketch of the consistency proof (part II)

# Sketch of the consistency proof (part II)

## Proof sketch

# Sketch of the consistency proof (part II)

## Proof sketch

The category $\mathcal{G}$, whose objects are games, and morphisms $A \to B$ are winning, recursive strategies on $A \twoheadrightarrow B$,

# Sketch of the consistency proof (part II)

## Proof sketch

The category $\mathcal{G}$, whose objects are games, and morphisms $A \to B$ are winning, recursive strategies on $A \twoheadrightarrow B$, gives rise to a *category with families*, an algebraic model of MLTT.

# Sketch of the consistency proof (part II)

## Proof sketch

The category $\mathcal{G}$, whose objects are games, and morphisms $A \to B$ are winning, recursive strategies on $A \twoheadrightarrow B$, gives rise to a *category with families*, an algebraic model of MLTT.
This model refutes falsity since strategies in $\mathcal{G}$ are all total.

# Sketch of the consistency proof (part II)

## Proof sketch

The category $\mathcal{G}$, whose objects are games, and morphisms $A \to B$ are winning, recursive strategies on $A \multimap B$, gives rise to a *category with families*, an algebraic model of MLTT.
This model refutes falsity since strategies in $\mathcal{G}$ are all total.
It remains to model the term $\mathsf{ct} : (\mathsf{N} \Rightarrow \mathsf{N}) \Rightarrow \mathsf{N}$ by a morphism $\mathrm{ct} : (N \multimap N) \multimap N$ in $\mathcal{G}$,

# Sketch of the consistency proof (part II)

## Proof sketch

The category $\mathcal{G}$, whose objects are games, and morphisms $A \to B$ are winning, recursive strategies on $A \multimap B$, gives rise to a *category with families*, an algebraic model of MLTT.
This model refutes falsity since strategies in $\mathcal{G}$ are all total.
It remains to model the term $\mathsf{ct} : (\mathsf{N} \Rightarrow \mathsf{N}) \Rightarrow \mathsf{N}$ by a morphism $\mathrm{ct} : (N \multimap N) \multimap N$ in $\mathcal{G}$, which is trivial.

# Sketch of the consistency proof (part II)

## Proof sketch

The category $\mathcal{G}$, whose objects are games, and morphisms $A \to B$ are winning, recursive strategies on $A \multimap B$, gives rise to a *category with families*, an algebraic model of MLTT.
This model refutes falsity since strategies in $\mathcal{G}$ are all total.
It remains to model the term $\mathsf{ct} : (\mathsf{N} \Rightarrow \mathsf{N}) \Rightarrow \mathsf{N}$ by a morphism $\mathrm{ct} : (N \multimap N) \multimap N$ in $\mathcal{G}$, which is trivial. □

# Sketch of the consistency proof (part II)

## Proof sketch

The category $\mathcal{G}$, whose objects are games, and morphisms $A \to B$ are winning, recursive strategies on $A \multimap B$, gives rise to a *category with families*, an algebraic model of MLTT.
This model refutes falsity since strategies in $\mathcal{G}$ are all total.
It remains to model the term $\mathsf{ct} : (\mathsf{N} \Rightarrow \mathsf{N}) \Rightarrow \mathsf{N}$ by a morphism $\mathrm{ct} : (N \multimap N) \multimap N$ in $\mathcal{G}$, which is trivial. □

I.e., ct just copy-cats the realizer given by O at his first move:

# Sketch of the consistency proof (part II)

## Proof sketch

The category $\mathcal{G}$, whose objects are games, and morphisms $A \to B$ are winning, recursive strategies on $A \twoheadrightarrow B$, gives rise to a *category with families*, an algebraic model of MLTT.
This model refutes falsity since strategies in $\mathcal{G}$ are all total.
It remains to model the term $\mathsf{ct} : (\mathsf{N} \Rightarrow \mathsf{N}) \Rightarrow \mathsf{N}$ by a morphism $\mathrm{ct} : (N \twoheadrightarrow N) \twoheadrightarrow N$ in $\mathcal{G}$, which is trivial. □

I.e., ct just copy-cats the realizer given by O at his first move:

$$\frac{(N \twoheadrightarrow N) \overset{\mathrm{ct}}{\twoheadrightarrow} N}{\begin{array}{c} q_{[e]} \\ e_{[e]} \end{array}}$$

# Concluding remarks

# Concluding remarks

Recursion theory: It is not even r.e. to calculate a realizer for a given total recursive function.

# Concluding remarks

Recursion theory: It is not even r.e. to calculate a realizer for a given total recursive function.

## Main idea (further rephrased)

# Concluding remarks

Recursion theory: It is not even r.e. to calculate a realizer for a given total recursive function.

## Main idea (further rephrased)

We impose this computationally infeasible task $\{f : \mathbb{N} \to \mathbb{N}\} \to \{e \in \mathbb{N}\}$ (s.t. $e$ realizes $f$) only on O (= an oracle), but not on P (= a strategy).

# Concluding remarks

Recursion theory: It is not even r.e. to calculate a realizer for a given total recursive function.

## Main idea (further rephrased)

We impose this computationally infeasible task $\{f : \mathbb{N} \to \mathbb{N}\} \to \{e \in \mathbb{N}\}$ (s.t. $e$ realizes $f$) only on O (= an oracle), but not on P (= a strategy).

## Further work

# Concluding remarks

Recursion theory: It is not even r.e. to calculate a realizer for a given total recursive function.

## Main idea (further rephrased)

We impose this computationally infeasible task $\{f : \mathbb{N} \to \mathbb{N}\} \to \{e \in \mathbb{N}\}$ (s.t. $e$ realizes $f$) only on O (= an oracle), but not on P (= a strategy).

## Further work

1. Thanks to the *compositionality* of my method, one would easily apply the present method to extensions of MLTT + CT.

# Concluding remarks

Recursion theory: It is not even r.e. to calculate a realizer for a given total recursive function.

## Main idea (further rephrased)

We impose this computationally infeasible task $\{f : \mathbb{N} \to \mathbb{N}\} \to \{e \in \mathbb{N}\}$ (s.t. $e$ realizes $f$) only on O (= an oracle), but not on P (= a strategy).

## Further work

1. Thanks to the *compositionality* of my method, one would easily apply the present method to extensions of MLTT + CT.
2. Generally, there would be more meta-theoretic problems on MLTT for which game semantics works well.


Abramsky, S. and McCusker, G. (1999).
Game semantics.
In *Computational Logic: Proceedings of the 1997 Marktoberdorf Summer School*, pages 1–55, Berlin, Heidelberg. Springer.


Ishihara, H., Maietti, M. E., Maschio, S., and Streicher, T. (2018).
Consistency of the intensional level of the minimalist foundation with church's thesis and axiom of choice.
*Archive for Mathematical Logic*, 57(7-8):873–888.


Maietti, M. E. and Sambin, G. (2005).
Toward a minimalist foundation for constructive mathematics.
*From Sets and Types to Topology and Analysis: Practicable Foundations for Constructive Mathematics*, 48:91–114.


Yamada, N. (2016).
Game semantics of Martin-Löf type theory, part I: a mathematical theory of predicative games and strategies.
*arXiv preprint arXiv:1610.01669*.


Yamada, N. (2020).
Game semantics of Martin-Löf type theory, part III: its consistency with Church's thesis.
*arXiv preprint arXiv:2007.08094.*