Accessible Sets in Martin-Löf Type Theory with Function Extensionality

Yuta Takahashi[1]

Aomori University, Japan

June 9th, 2025
TYPES 2025

[1]This work is supported by JSPS KAKENHI Grant Number JP21K12822.

CZF and MLTT

- To formalize Bishop's constructive analysis,
  - Aczel [1, 2, 3] introduced a system of constructive set theory called constructive Zermelo-Fraenkel set theory **CZF**
    - Several set-theoretic principles and set-existence axioms were codified on the basis of first-order intuitionistic logic with equality

CZF and MLTT

- To formalize Bishop's constructive analysis,
  - Aczel [1, 2, 3] introduced a system of constructive set theory called constructive Zermelo-Fraenkel set theory **CZF**
    - Several set-theoretic principles and set-existence axioms were codified on the basis of first-order intuitionistic logic with equality
  - Martin-Löf [4] took a different approach: he formulated a framework of constructive type theory called **MLTT**

CZF and MLTT

- To formalize Bishop's constructive analysis,
  - Aczel [1, 2, 3] introduced a system of constructive set theory called constructive Zermelo-Fraenkel set theory **CZF**
    - Several set-theoretic principles and set-existence axioms were codified on the basis of first-order intuitionistic logic with equality
  - Martin-Löf [4] took a different approach: he formulated a framework of constructive type theory called **MLTT**
- Aczel also showed that these two approaches are compatible
  - He defined a cumulative hierarchy $\mathbb{V}$ of sets as a W-type in **MLTT**, and interpreted all axioms of **CZF** in **MLTT**
  - $\mathbb{V}$ is a type with the equivalence relation $\doteq$, which is similar to bisimulation

CZF and MLTT

- To formalize Bishop's constructive analysis,
  - Aczel [1, 2, 3] introduced a system of constructive set theory called constructive Zermelo-Fraenkel set theory **CZF**
    - Several set-theoretic principles and set-existence axioms were codified on the basis of first-order intuitionistic logic with equality
  - Martin-Löf [4] took a different approach: he formulated a framework of constructive type theory called **MLTT**
- Aczel also showed that these two approaches are compatible
  - He defined a cumulative hierarchy $\mathbb{V}$ of sets as a W-type in **MLTT**, and interpreted all axioms of **CZF** in **MLTT**
  - $\mathbb{V}$ is a type with the equivalence relation $\doteq$, which is similar to bisimulation
- Each set $a : \mathbb{V}$ can be considered as $\{\mathsf{pred}\, a\, x \mid x : \mathsf{index}\, a\}$
  - $\mathsf{index}\, a$ is the type of indices for the elements of a
  - $\mathsf{pred}\, a\, x$ with $x : \mathsf{index}\, a$ is the element of a of index x
- The relation $a \in b$ is defined as $a \in b := \Sigma_{(x:\mathsf{index}\, b)}(a \doteq \mathsf{pred}\, b\, x)$

Transitive Closures of Sets

- The transitive closure of a set can be defined in **CZF** (cf. [5])
  - The transitive closure $\mathbf{TC}(a)$ of a set a satisfies the equation

$$\mathbf{TC}(a) = a \cup \bigcup \{\mathbf{TC}(x) \mid x \in a\},$$

  which implies that $\mathbf{TC}(a)$ is a transitive set:

$$\forall x \forall y (y \in x \in \mathbf{TC}(a) \to y \in \mathbf{TC}(a))$$

So $\mathbf{TC}(a)$ contains as its elements all sets below a in the hierarchy

Transitive Closures of Sets

- The transitive closure of a set can be defined in **CZF** (cf. [5])
  - The transitive closure $\mathbf{TC}(a)$ of a set a satisfies the equation

$$\mathbf{TC}(a) = a \cup \bigcup\{\mathbf{TC}(x) \mid x \in a\},$$

  which implies that $\mathbf{TC}(a)$ is a transitive set:

$$\forall x \forall y (y \in x \in \mathbf{TC}(a) \to y \in \mathbf{TC}(a))$$

  So $\mathbf{TC}(a)$ contains as its elements all sets below a in the hierarchy

- Through Aczel's interpretation of **CZF**, one has the corresponding operator $\mathsf{tc} : \mathbb{V} \to \mathbb{V}$ in **MLTT**

Transitive Closures of Sets

- The transitive closure of a set can be defined in **CZF** (cf. [5])
  - The transitive closure $\mathbf{TC}(a)$ of a set a satisfies the equation

$$\mathbf{TC}(a) = a \cup \bigcup\{\mathbf{TC}(x) \mid x \in a\},$$

  which implies that $\mathbf{TC}(a)$ is a transitive set:

$$\forall x \forall y (y \in x \in \mathbf{TC}(a) \to y \in \mathbf{TC}(a))$$

  So $\mathbf{TC}(a)$ contains as its elements all sets below a in the hierarchy
- Through Aczel's interpretation of **CZF**, one has the corresponding operator $\mathsf{tc} : \mathbb{V} \to \mathbb{V}$ in **MLTT**
- By using Dybjer's indexed inductive definition [6], one can then define the accessibility $\mathsf{Acc} : \mathbb{V} \to \mathsf{Set}$ on $\mathbb{V}$

Accessible Sets

- Put $\forall_{(x \in a)} \Phi(x) := (i : \mathsf{index}\, a) \to \Phi(\mathsf{pred}\, a\, i)$
- The type $\mathsf{Acc}\, a$ says that "a set a is constructed from below"
  - the constructor $\mathsf{prog} : (a : \mathbb{V}) \to \forall_{(x \in \mathsf{tc}\, a)} \mathsf{Acc}\, x \to \mathsf{Acc}\, a$

Accessible Sets

- Put $\forall_{(x \in a)} \Phi(x) := (i : \mathsf{index}\, a) \to \Phi(\mathsf{pred}\, a\, i)$
- The type $\mathsf{Acc}\, a$ says that "a set a is constructed from below"
  - the constructor $\mathsf{prog} : (a : \mathbb{V}) \to \forall_{(x \in \mathsf{tc}\, a)} \mathsf{Acc}\, x \to \mathsf{Acc}\, a$
  - the induction principle $\mathsf{ind}_{\mathsf{Acc}}$:

$$
\begin{aligned}
\mathsf{ind}_{\mathsf{Acc}} : (P : (a : \mathbb{V}) \to \mathsf{Acc}\, a \to \mathsf{Set}\, \ell) \to \\
\Big((a : \mathbb{V})(f : \forall_{(x \in \mathsf{tc}\, a)} \mathsf{Acc}\, x) \to \\
((i : \mathsf{index}\, (\mathsf{tc}\, a)) \to P\, (\mathsf{pred}\, (\mathsf{tc}\, a)\, i)\, (f\ i)) \to \\
P\, a\, (\mathsf{prog}\, a\, f)\Big) \to \\
(a : \mathbb{V})(c : \mathsf{Acc}\, a) \to P\, a\, c
\end{aligned}
$$

$$
\begin{aligned}
&\mathsf{ind}_{\mathsf{Acc}}\, P\, h\, a\, (\mathsf{prog}\, a\, f) = \\
&h\, a\, f\, (\lambda i.\mathsf{ind}_{\mathsf{Acc}}\, P\, h\, (\mathsf{pred}\, (\mathsf{tc}\, a)\, i)\, (f\ i))
\end{aligned}
$$

Properties of Acc

- The induction principle for Acc is stronger than $\mathbb{V}$-induction, i.e., the W-induction principle on $\mathbb{V}$
  - The former admits the induction hypothesis not only for each $v \in a$, but also for each $w \in \mathsf{tc}\, a$ (i.e., any set w below a in the hierarchy)

Properties of Acc

- The induction principle for Acc is stronger than $\mathbb{V}$-induction, i.e., the W-induction principle on $\mathbb{V}$
  - The former admits the induction hypothesis not only for each $v \in a$, but also for each $w \in \mathsf{tc}\, a$ (i.e., any set w below a in the hierarchy)
  - E.g., a universe type $\mathsf{U}\, a$ containing as its subuniverses not only $\mathsf{U}\, v$ for any $v \in a$, but also $\mathsf{U}\, w$ for any $w \in \mathsf{tc}\, a$

Properties of Acc

- The induction principle for Acc is stronger than $\mathbb{V}$-induction, i.e., the W-induction principle on $\mathbb{V}$
  - The former admits the induction hypothesis not only for each $v \in a$, but also for each $w \in \mathsf{tc}\, a$ (i.e., any set w below a in the hierarchy)
  - E.g., a universe type $\mathsf{U}\, a$ containing as its subuniverses not only $\mathsf{U}\, v$ for any $v \in a$, but also $\mathsf{U}\, w$ for any $w \in \mathsf{tc}\, a$
- The Acc-induction principle has the simple computation rule
  - In fact, the operator tc is accompanied by a similar induction principle $\mathsf{ind}_{\mathsf{tc}}$, which is stronger than $\mathbb{V}$-induction too
  - But $\mathsf{ind}_{\mathsf{tc}}$ lacks a simple computation rule

Computation of tc-Induction

- One might try to show that $\mathsf{ind}_{\mathsf{tc}}$ has the computation rule below:

$$
\begin{aligned}
\mathsf{ind}_{\mathsf{tc}}\ & P\ \text{[a predicate for induction]} \\
& h\ \text{[an inductive clause]} \\
& a\ \text{[an argument]} \\
& = h\ a\ (\lambda i.\mathsf{ind}_{\mathsf{tc}}\ P\ h\ (\mathsf{pred}\ (\mathsf{tc}\ a)\ i))
\end{aligned}
$$

Computation of tc-Induction

- One might try to show that $\mathsf{ind}_{\mathsf{tc}}$ has the computation rule below:

$$
\begin{aligned}
\mathsf{ind}_{\mathsf{tc}}\, P \, &[\text{a predicate for induction}] \\
h \, &[\text{an inductive clause}] \\
a \, &[\text{an argument}] \\
&= h\, a\, (\lambda i.\mathsf{ind}_{\mathsf{tc}}\, P\, h\, (\mathsf{pred}\, (\mathsf{tc}\, a)\, i))
\end{aligned}
$$

  but this is a non-terminating rule:

$$
\begin{aligned}
\mathsf{ind}_{\mathsf{tc}}\, P\, h\, a &= h\, a\, (\lambda i.\mathsf{ind}_{\mathsf{tc}}\, P\, h\, (\mathsf{pred}\, (\mathsf{tc}\, a)\, i)) \\
&= h\, a\, (\lambda i.h\, (\mathsf{pred}\, (\mathsf{tc}\, a)\, i)(\lambda j.\mathsf{ind}_{\mathsf{tc}}\, P\, h\, (\mathsf{pred}\, (\mathsf{tc}\, (\mathsf{pred}\, (\mathsf{tc}\, a)\, i))\, j))) = \cdots
\end{aligned}
$$

Aim

- We first show
  - In **MLTT** with function extensionality the above computation rule of the tc-induction principle holds propositionally
  - With this propositional computation rule $\mathsf{comp}_{\mathsf{tc}}$, the tc-induction principle provides a useful inductive definition

Aim

- We first show
  - In **MLTT** with function extensionality the above computation rule of the tc-induction principle holds propositionally
  - With this propositional computation rule $\mathsf{comp}_{\mathsf{tc}}$, the tc-induction principle provides a useful inductive definition
- We then verify
  - The accessibility Acc on $\mathbb{V}$ is definable by means of $\mathsf{comp}_{\mathsf{tc}}$ without indexed inductive definition

$$\mathsf{Acc}\, a =_{\mathsf{Set}} \forall_{(x \in \mathsf{tc}\, a)} \mathsf{Acc}\, x$$

  - Here the constructor $\mathsf{prog} : (a : \mathbb{V}) \to \forall_{(x \in \mathsf{tc}\, a)} \mathsf{Acc}\, x \to \mathsf{Acc}\, a$ is defined by transporting from the RHS to LHS
  - The Acc-induction principle is defined by transporting in the opposite direction

Aim

- We first show
  - In **MLTT** with function extensionality the above computation rule of the tc-induction principle holds propositionally
  - With this propositional computation rule $\mathsf{comp}_{\mathsf{tc}}$, the tc-induction principle provides a useful inductive definition
- We then verify
  - The accessibility Acc on $\mathbb{V}$ is definable by means of $\mathsf{comp}_{\mathsf{tc}}$ without indexed inductive definition

$$\mathsf{Acc}\,a =_{\mathsf{Set}} \forall_{(x \in \mathsf{tc}\,a)} \mathsf{Acc}\,x$$

  - Here the constructor $\mathsf{prog} : (a : \mathbb{V}) \to \forall_{(x \in \mathsf{tc}\,a)} \mathsf{Acc}\,x \to \mathsf{Acc}\,a$ is defined by transporting from the RHS to LHS
  - The Acc-induction principle is defined by transporting in the opposite direction
- By using function extensionality again, we show that the type $\mathsf{Acc}\,a$ has a unique inhabitant for any $a : \mathbb{V}$

tc-Induction Principle

- Function extensionality:

$$\mathsf{funext} : (A : \mathsf{Set}\ \ell_1)(B : A \to \mathsf{Set}\ \ell_2) \\ (f\ g : (x : A) \to B\ x) \to ((x : A) \to f\ x =_{B\ x} g\ x) \to \\ f =_{(x:A) \to B\ x} g$$

- Without funext, one can derive the tc-induction principle

$$\mathsf{ind}_{\mathsf{tc}} : (P : \mathbb{V} \to \mathsf{Set}\ \ell) \to ((a : \mathbb{V}) \to \forall_{(x \in \mathsf{tc}\ a)} P\ x \to P\ a) \to \\ (a : \mathbb{V}) \to P\ a$$

tc-Induction Principle

- Function extensionality:

$$\mathsf{funext} : (A : \mathsf{Set}\ \ell_1)(B : A \to \mathsf{Set}\ \ell_2)$$
$$(f\ g : (x : A) \to B\ x) \to ((x : A) \to f\ x =_{B\ x} g\ x) \to$$
$$f =_{(x:A)\to B\ x} g$$

- Without funext, one can derive the tc-induction principle

$$\mathsf{ind}_{\mathsf{tc}} : (P : \mathbb{V} \to \mathsf{Set}\ \ell) \to ((a : \mathbb{V}) \to \forall_{(x \in \mathsf{tc}\ a)} P\ x \to P\ a) \to$$
$$(a : \mathbb{V}) \to P\ a$$

Proposition (with funext)

For any $P : \mathbb{V} \to \mathsf{Set}\ \ell$, $h : (b : \mathbb{V}) \to \forall_{(x \in \mathsf{tc}\ b)} P\ x \to P\ b$ *and* $a : \mathbb{V}$, *we have*

$$\mathsf{comp}_{\mathsf{tc}}\ P\ h\ a : \mathsf{ind}_{\mathsf{tc}}\ P\ h\ a =_{P\ a} h\ a\ (\lambda i.\mathsf{ind}_{\mathsf{tc}}\ P\ h\ (\mathsf{pred}\ (\mathsf{tc}\ a)\ i)).$$

Accessible Sets by $\mathsf{comp}_{\mathsf{tc}}$

- Putting

$$
\begin{aligned}
P \text{ [a predicate for induction] } &:= \lambda a.\mathsf{Set} \\
\mathsf{acc} \text{ [an inductive clause] } &:= \lambda a.\lambda g.(i : \mathsf{index}\,(\mathsf{tc}\,a)) \to g\,i \\
\mathsf{Acc} &:= \mathsf{ind}_{\mathsf{tc}}\,P\,\mathsf{acc}
\end{aligned}
$$

we have

$$
\mathsf{comp}_{\mathsf{tc}}\,P\,\mathsf{acc}\,a : \mathsf{Acc}\,a =_{\mathsf{Set}} \forall_{(x \in \mathsf{tc}\,a)} \mathsf{Acc}\,x
$$

Accessible Sets by $\mathsf{comp_{tc}}$

- Putting

$$\begin{aligned} P \text{ [a predicate for induction]} &:= \lambda a.\mathsf{Set} \\ \mathsf{acc} \text{ [an inductive clause]} &:= \lambda a.\lambda g.(i : \mathsf{index}\,(\mathsf{tc}\,a)) \to g\,i \\ \mathsf{Acc} &:= \mathsf{ind_{tc}}\,P\,\mathsf{acc} \end{aligned}$$

  we have

$$\mathsf{comp_{tc}}\,P\,\mathsf{acc}\,a : \mathsf{Acc}\,a =_{\mathsf{Set}} \forall_{(x \in \mathsf{tc}\,a)} \mathsf{Acc}\,x$$

- By transporting from RHS to LHS, we have

$$\mathsf{prog} : (a : \mathbb{V}) \to \forall_{(x \in \mathsf{tc}\,a)} \mathsf{Acc}\,x \to \mathsf{Acc}\,a$$

Accessible Sets by $\mathsf{comp}_{\mathsf{tc}}$

- Putting

$$
\begin{aligned}
P \text{ [a predicate for induction]} &:= \lambda a.\mathsf{Set} \\
\mathsf{acc} \text{ [an inductive clause]} &:= \lambda a.\lambda g.(i : \mathsf{index}\,(\mathsf{tc}\, a)) \to g\, i \\
\mathsf{Acc} &:= \mathsf{ind}_{\mathsf{tc}}\, P\, \mathsf{acc}
\end{aligned}
$$

we have

$$\mathsf{comp}_{\mathsf{tc}}\, P\, \mathsf{acc}\, a : \mathsf{Acc}\, a =_{\mathsf{Set}} \forall_{(x \in \mathsf{tc}\, a)} \mathsf{Acc}\, x$$

- By transporting from RHS to LHS, we have

$$\mathsf{prog} : (a : \mathbb{V}) \to \forall_{(x \in \mathsf{tc}\, a)} \mathsf{Acc}\, x \to \mathsf{Acc}\, a$$

- In the opposite direction,

$$\mathsf{inv} : (a : \mathbb{V}) \to \mathsf{Acc}\, a \to \forall_{(x \in \mathsf{tc}\, a)} \mathsf{Acc}\, x$$

Derived Acc-Induction Principle

- The Acc-induction principle $\mathsf{ind}_{\mathsf{Acc}}$:

$$\begin{aligned}
\mathsf{ind}_{\mathsf{Acc}} : (P : (a : \mathbb{V}) \to \mathsf{Acc}\, a \to \mathsf{Set}\, \ell) \to \\
\Big((a : \mathbb{V})(f : \forall_{(x \in \mathsf{tc}\, a)} \mathsf{Acc}\, x) \to \\
((i : \mathsf{index}\,(\mathsf{tc}\, a)) \to P\,(\mathsf{pred}\,(\mathsf{tc}\, a)\, i)\,(f\, i)) \to \\
P\, a\,(\mathsf{prog}\, a\, f)\Big) \to (a : \mathbb{V})(c : \mathsf{Acc}\, a) \to P\, a\, c
\end{aligned}$$

- By transporting along $\mathsf{Acc}\, a =_{\mathsf{Set}} \forall_{(x \in \mathsf{tc}\, a)} \mathsf{Acc}\, x$,
  we have $P\, a\,(\mathsf{prog}\, a\,(\mathsf{inv}\, a\, c))$

Derived Acc-Induction Principle

- The Acc-induction principle $\mathsf{ind}_{\mathsf{Acc}}$:

$$
\begin{aligned}
\mathsf{ind}_{\mathsf{Acc}} : (P : (a : \mathbb{V}) \to \mathsf{Acc}\, a \to \mathsf{Set}\, \ell) \to \\
\Big((a : \mathbb{V})(f : \forall_{(x \in \mathsf{tc}\, a)} \mathsf{Acc}\, x) \to \\
((i : \mathsf{index}\, (\mathsf{tc}\, a)) \to P\, (\mathsf{pred}\, (\mathsf{tc}\, a)\, i)\, (f\, i)) \to \\
P\, a\, (\mathsf{prog}\, a\, f) \Big) \to (a : \mathbb{V})(c : \mathsf{Acc}\, a) \to P\, a\, c
\end{aligned}
$$

- By transporting along $\mathsf{Acc}\, a =_{\mathsf{Set}} \forall_{(x \in \mathsf{tc}\, a)} \mathsf{Acc}\, x$, we have $P\, a\, (\mathsf{prog}\, a\, (\mathsf{inv}\, a\, c))$
- From a general fact on $\mathsf{transport}$

$$
\begin{aligned}
(A : \mathsf{Set}\, \ell_1)(P : A \to \mathsf{Set}\, \ell_2)(x\, y : A)(p : x =_A y)(c : P\, x) \\
\to \mathsf{transport}\, P\, (\mathsf{sym}\, p)\, (\mathsf{transport}\, P\, p\, c) =_{P\, x} c,
\end{aligned}
$$

we have $\mathsf{prog}\, a\, (\mathsf{inv}\, a\, c) =_{\mathsf{Acc}\, a} c$, hence $P\, a\, c$ holds

Uniqueness of $\mathsf{Acc}\, a$

- We first verify that for any $a : \mathbb{V}$, $\mathsf{Acc}\, a$ holds

Uniqueness of $\mathsf{Acc}\, a$

- We first verify that for any $a : \mathbb{V}$, $\mathsf{Acc}\, a$ holds
- We then show $(t\ s : \mathsf{Acc}\, a) \to t =_{\mathsf{Acc}\, a} s$ by tc-induction on a: for any $t, s : \mathsf{Acc}\, a$, we have

$$\mathsf{inv}\, a\, t : \forall_{(x \in \mathsf{tc}\, a)} \mathsf{Acc}\, x \qquad \mathsf{inv}\, a\, s : \forall_{(x \in \mathsf{tc}\, a)} \mathsf{Acc}\, x$$

- By IH, $(x : \mathsf{index}\, (\mathsf{tc}\, a)) \to \mathsf{inv}\, a\, t\, x =_{\mathsf{Acc}\, x} \mathsf{inv}\, a\, s\, x$ holds

Uniqueness of $\mathsf{Acc}\, a$

- We first verify that for any $a : \mathbb{V}$, $\mathsf{Acc}\, a$ holds
- We then show $(t\ s : \mathsf{Acc}\, a) \to t =_{\mathsf{Acc}\, a} s$ by tc-induction on a: for any $t, s : \mathsf{Acc}\, a$, we have

$$\mathsf{inv}\, a\, t : \forall_{(x \in \mathsf{tc}\, a)} \mathsf{Acc}\, x \qquad \mathsf{inv}\, a\, s : \forall_{(x \in \mathsf{tc}\, a)} \mathsf{Acc}\, x$$

- By IH, $(x : \mathsf{index}\,(\mathsf{tc}\, a)) \to \mathsf{inv}\, a\, t\, x =_{\mathsf{Acc}\, x} \mathsf{inv}\, a\, s\, x$ holds
- By funext, we then have $\mathsf{inv}\, a\, t =_{\forall_{(x \in \mathsf{tc}\, a)} \mathsf{Acc}\, x} \mathsf{inv}\, a\, s$

Uniqueness of $\mathsf{Acc}\, a$

- We first verify that for any $a : \mathbb{V}$, $\mathsf{Acc}\, a$ holds
- We then show $(t\ s : \mathsf{Acc}\, a) \to t =_{\mathsf{Acc}\, a} s$ by tc-induction on a: for any $t, s : \mathsf{Acc}\, a$, we have

$$\mathsf{inv}\, a\, t : \forall_{(x \in \mathsf{tc}\, a)} \mathsf{Acc}\, x \qquad \mathsf{inv}\, a\, s : \forall_{(x \in \mathsf{tc}\, a)} \mathsf{Acc}\, x$$

- By IH, $(x : \mathsf{index}\, (\mathsf{tc}\, a)) \to \mathsf{inv}\, a\, t\, x =_{\mathsf{Acc}\, x} \mathsf{inv}\, a\, s\, x$ holds
- By funext, we then have $\mathsf{inv}\, a\, t =_{\forall_{(x \in \mathsf{tc}\, a)} \mathsf{Acc}\, x} \mathsf{inv}\, a\, s$
- The congruence with $\mathsf{prog}\, a$ gives

$$\mathsf{prog}\, a\, (\mathsf{inv}\, a\, t) =_{\mathsf{Acc}\, a} \mathsf{prog}\, a\, (\mathsf{inv}\, a\, s)$$

Uniqueness of $\mathsf{Acc}\, a$

- We first verify that for any $a : \mathbb{V}$, $\mathsf{Acc}\, a$ holds
- We then show $(t\ s : \mathsf{Acc}\, a) \to t =_{\mathsf{Acc}\, a} s$ by tc-induction on a: for any $t, s : \mathsf{Acc}\, a$, we have

$$\mathsf{inv}\, a\, t : \forall_{(x \in \mathsf{tc}\, a)} \mathsf{Acc}\, x \qquad \mathsf{inv}\, a\, s : \forall_{(x \in \mathsf{tc}\, a)} \mathsf{Acc}\, x$$

- By IH, $(x : \mathsf{index}\, (\mathsf{tc}\, a)) \to \mathsf{inv}\, a\, t\, x =_{\mathsf{Acc}\, x} \mathsf{inv}\, a\, s\, x$ holds
- By funext, we then have $\mathsf{inv}\, a\, t =_{\forall_{(x \in \mathsf{tc}\, a)} \mathsf{Acc}\, x} \mathsf{inv}\, a\, s$
- The congruence with $\mathsf{prog}\, a$ gives

$$\mathsf{prog}\, a\, (\mathsf{inv}\, a\, t) =_{\mathsf{Acc}\, a} \mathsf{prog}\, a\, (\mathsf{inv}\, a\, s)$$

- Canceling the both sides, we obtain $t =_{\mathsf{Acc}\, a} s$

Future Work

- Aczel's interpretation of **CZF** in **MLTT** was refined in Homotopy type theory (HoTT) [7]
  - The cumulative hierarchy $\mathbb{V}$ of sets is defined not as a W-type but as a higher inductive type
  - The equivalence relation $\doteq$ on $\mathbb{V}$ is replaced with the identity type $=_{\mathbb{V}}$ and $\mathbb{V}$ has the path constructor for $=_{\mathbb{V}}$
  - Other interpretations of **CZF** in HoTT were investigated in, e.g., [8, 9, 10]

Future Work

- Aczel's interpretation of **CZF** in **MLTT** was refined in Homotopy type theory (HoTT) [7]
  - The cumulative hierarchy $\mathbb{V}$ of sets is defined not as a W-type but as a higher inductive type
  - The equivalence relation $\doteq$ on $\mathbb{V}$ is replaced with the identity type $=_{\mathbb{V}}$ and $\mathbb{V}$ has the path constructor for $=_{\mathbb{V}}$
  - Other interpretations of **CZF** in HoTT were investigated in, e.g., [8, 9, 10]
- In the literature of HoTT the accessible part of a binary relation is defined by indexed inductive definition [7, 11]
- We will examine in some HoTT-interpretation of **CZF**
  - whether the tc-induction principle and its propositional computation rule are derivable
  - whether the accessibility Acc on $\mathbb{V}$ is definable without indexed inductive definition

Thank you for your attention!

References I

[1] Peter Aczel.
The type theoretic interpretation of constructive set theory.
In Angus Macintyre, Leszek Pacholski, and Jeff Paris, editors, *Logic Colloquium '77*, volume 96 of *Studies in Logic and the Foundations of Mathematics*, pages 55–66. Elsevier, 1978.

[2] Peter Aczel.
The type theoretic interpretation of constructive set theory: Choice principles.
In A. S. Troelstra and D. van Dalen, editors, *The L.E.J. Brouwer Centenary Symposium*, pages 1–40. North-Holland, 1982.

[3] Peter Aczel.
The type theoretic interpretation of constructive set theory: Inductive definitions.
In R. B. Marcus, G. J. Dorn, and G. J. W. Dorn, editors, *Logic, Methodology, and Philosophy of Science VII*, pages 17–49. North-Holland, 1986.

[4] Per Martin-Löf.
An intuitionistic theory of types.
In G. Sambin and Jan M. Smith, editors, *Twenty-five years of constructive type theory*, volume 36 of *Oxford Logic Guides*, pages 127–172. Clarendon Press, 1998.

[5] Edward R. Griffor and Michael Rathjen.
The strength of some Martin-Löf type theories.
Arch. Math. Log., 33(5):347–385, 1994.

[6] Peter Dybjer.
Inductive families.
Formal Aspects Comput., 6(4):440–465, 1994.

References II

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

[8] Håkon Robbestad Gylterud.
From multisets to sets in homotopy type theory.
J. Symb. Log., 83(3):1132–1146, 2018.

[9] Håkon Robbestad Gylterud.
Multisets in type theory.
Mathematical Proceedings of the Cambridge Philosophical Society, 169(1):1–18, 2020.

[10] Cesare Gallozzi.
Homotopy type-theoretic interpretations of constructive set theories.
Math. Struct. Comput. Sci., 31(1):112–143, 2021.

[11] Tom de Jong, Nicolai Kraus, Fredrik Nordvall Forsberg, and Chuangjie Xu.
Set-theoretic and type-theoretic ordinals coincide.
In *LICS*, pages 1–13, 2023.