# Ground Stratified Induction for the Logic of Definitions

Nathan Guermond & Gopalan Nadathur

University of Minnesota Twin-Cities

December 12, 2025

# Outline

**Context:** The *logic of definitions* is

- A simply typed first order intuitionistic logic + definitions

# Outline

**Context:** The *logic of definitions* is

- A simply typed first order intuitionistic logic + definitions with
- a *stratification* condition to ensure consistency

# Outline

**Context:** The *logic of definitions* is

- A simply typed first order intuitionistic logic + definitions with
- a *stratification* condition to ensure consistency

**Previous work:**

- Tiu [2012] showed *ground stratification* is sufficient for consistency

# Outline

**Context:** The *logic of definitions* is

- A simply typed first order intuitionistic logic + definitions with
- a *stratification* condition to ensure consistency

**Previous work:**

- Tiu [2012] showed *ground stratification* is sufficient for consistency
- We showed [2025] that we can mix *ground stratified definitions* with *inductive definitions*

# Outline

**Context:** The *logic of definitions* is

- A simply typed first order intuitionistic logic + definitions with
- a *stratification* condition to ensure consistency

**Previous work:**

- Tiu [2012] showed *ground stratification* is sufficient for consistency
- We showed [2025] that we can mix *ground stratified definitions* with *inductive definitions*
- **Question:** Can we have ground stratified *inductive* definitions?

# Outline

**Context:** The *logic of definitions* is

- A simply typed first order intuitionistic logic + definitions with
- a *stratification* condition to ensure consistency

**Previous work:**

- Tiu [2012] showed *ground stratification* is sufficient for consistency
- We showed [2025] that we can mix *ground stratified definitions* with *inductive definitions*
- **Question:** Can we have ground stratified *inductive* definitions?

**This work:**

- We show that ground stratified inductive definitions are consistent with a new induction rule

# Outline

**Context:** The *logic of definitions* is

- A simply typed first order intuitionistic logic + definitions with
- a *stratification* condition to ensure consistency

**Previous work:**

- Tiu [2012] showed *ground stratification* is sufficient for consistency
- We showed [2025] that we can mix *ground stratified definitions* with *inductive definitions*
- **Question:** Can we have ground stratified *inductive* definitions?

**This work:**

- We show that ground stratified inductive definitions are consistent with a new induction rule
- This allows us to encode **inductive** logical relations defined by **recursion** on some argument

# Definitions

**Definition (Definition)**

A *definition* for a predicate $p$ is a set of clauses of the form

$$p\ t_1\ \ldots\ t_n \triangleq B$$

for terms $t_1, \ldots, t_n$ and a formula $B$.

# Definitions

### Definition (Definition)

A *definition* for a predicate $p$ is a set of clauses of the form

$$p\ t_1\ \ldots\ t_n \triangleq B$$

for terms $t_1, \ldots, t_n$ and a formula $B$.

Eg. Given a signature for naturals:

$$\texttt{z} : tm \qquad\qquad \texttt{suc} : tm \to tm$$

# Definitions

**Definition (Definition)**

A *definition* for a predicate $p$ is a set of clauses of the form

$$p\ t_1\ \dots\ t_n \triangleq B$$

for terms $t_1, \dots, t_n$ and a formula $B$.

Eg. Given a signature for naturals:

$$\mathtt{z} : tm \qquad\qquad \mathtt{suc} : tm \to tm$$

We define the predicates $nat : tm \to \mathbf{prop}$

$$nat\ \mathtt{z} \triangleq \top \qquad\qquad nat\ (\mathtt{suc}\ X) \triangleq nat\ X$$

# Definitions

**Definition (Definition)**

A *definition* for a predicate $p$ is a set of clauses of the form

$$p\ t_1\ \ldots\ t_n \triangleq B$$

for terms $t_1, \ldots, t_n$ and a formula $B$.

Eg. Given a signature for naturals:

$$\mathtt{z} : tm \qquad\qquad \mathtt{suc} : tm \to tm$$

We define the predicates $nat : tm \to \mathbf{prop}$ and $even : tm \to \mathbf{prop}$

$$nat\ \mathtt{z} \triangleq \top \qquad\qquad nat\ (\mathtt{suc}\ X) \triangleq nat\ X$$

$$even\ \mathtt{z} \triangleq \top \qquad\qquad even\ (\mathtt{suc}\ (\mathtt{suc}\ X)) \triangleq even\ X$$

# Unfolding definitions

The logic comes with *unfolding rules* for definitions.

# Unfolding definitions

The logic comes with *unfolding rules* for definitions.
Unfolding a goal formula corresponds to **pattern matching**:

$$\Longrightarrow \ \textit{even}\ (\texttt{suc}\ (\texttt{suc}\ \texttt{z}))$$

# Unfolding definitions

The logic comes with *unfolding rules* for definitions.
Unfolding a goal formula corresponds to **pattern matching**:

$$\Longrightarrow \; \mathit{even}\ (\mathtt{suc}\ (\mathtt{suc}\ \mathtt{z}))$$

Unfolding a hypothesis corresponds to **case analysis**:

$$\mathit{nat}\ x \;\Longrightarrow\; (x = \mathtt{z}) \vee \exists y.(x = \mathtt{suc}\ y) \wedge (\mathit{nat}\ y)$$

# Unfolding definitions

The logic comes with *unfolding rules* for definitions.
Unfolding a goal formula corresponds to **pattern matching**:

$$\Longrightarrow \; \mathit{even}\ (\mathtt{suc}\ (\mathtt{suc}\ \mathtt{z}))$$

Unfolding a hypothesis corresponds to **case analysis**:

$$\mathit{nat}\ x \;\Longrightarrow\; (x = \mathtt{z}) \lor \exists y.(x = \mathtt{suc}\ y) \land (\mathit{nat}\ y)$$

$$\mathit{even}\ (\mathtt{suc}\ z) \;\Longrightarrow\; \bot$$

# Unfolding definitions

The logic comes with *unfolding rules* for definitions.
Unfolding a goal formula corresponds to **pattern matching**:

$$\Longrightarrow \; \mathit{even}\ (\mathtt{suc}\ (\mathtt{suc}\ \mathtt{z}))$$

Unfolding a hypothesis corresponds to **case analysis**:

$$\mathit{nat}\ x \;\Longrightarrow\; (x = \mathtt{z}) \vee \exists y.(x = \mathtt{suc}\ y) \wedge (\mathit{nat}\ y)$$

$$\mathit{even}\ (\mathtt{suc}\ z) \;\Longrightarrow\; \bot$$

We **cannot** prove

$$\mathit{nat}\ x \;\Longrightarrow\; (\mathit{even}\ x) \vee (\mathit{even}\ (\mathtt{suc}\ x))$$

# Unfolding definitions

The logic comes with *unfolding rules* for definitions.
Unfolding a goal formula corresponds to **pattern matching**:

$$\Longrightarrow \; \mathit{even}\ (\mathtt{suc}\ (\mathtt{suc}\ \mathtt{z}))$$

Unfolding a hypothesis corresponds to **case analysis**:

$$\mathit{nat}\ x \;\Longrightarrow\; (x = \mathtt{z}) \lor \exists y.(x = \mathtt{suc}\ y) \land (\mathit{nat}\ y)$$

$$\mathit{even}\ (\mathtt{suc}\ z) \;\Longrightarrow\; \bot$$

We **cannot** prove

$$\boxed{\mathit{nat}\ x} \;\Longrightarrow\; (\mathit{even}\ x) \lor (\mathit{even}\ (\mathtt{suc}\ x))$$

# Unfolding definitions

The logic comes with *unfolding rules* for definitions.
Unfolding a goal formula corresponds to **pattern matching**:

$$\Longrightarrow \ \mathit{even}\ (\mathtt{suc}\ (\mathtt{suc}\ \mathtt{z}))$$

Unfolding a hypothesis corresponds to **case analysis**:

$$\mathit{nat}\ x \ \Longrightarrow\ (x = \mathtt{z}) \lor \exists y.(x = \mathtt{suc}\ y) \land (\mathit{nat}\ y)$$

$$\mathit{even}\ (\mathtt{suc}\ z) \ \Longrightarrow\ \bot$$

We **cannot** prove

$$\begin{cases} \quad\quad \Longrightarrow & \boxed{(\mathit{even}\ \mathtt{z}) \lor (\mathit{even}\ (\mathtt{suc}\ \mathtt{z}))} \\ \mathit{nat}\ y \ \Longrightarrow & (\mathit{even}\ (\mathtt{suc}\ y)) \lor (\mathit{even}\ (\mathtt{suc}\ (\mathtt{suc}\ y))) \end{cases}$$

# Unfolding definitions

The logic comes with *unfolding rules* for definitions.
Unfolding a goal formula corresponds to **pattern matching**:

$$\Longrightarrow \; \mathit{even}\ (\mathtt{suc}\ (\mathtt{suc}\ \mathtt{z}))$$

Unfolding a hypothesis corresponds to **case analysis**:

$$\mathit{nat}\ x \;\Longrightarrow\; (x = \mathtt{z}) \lor \exists y.(x = \mathtt{suc}\ y) \land (\mathit{nat}\ y)$$

$$\mathit{even}\ (\mathtt{suc}\ z) \;\Longrightarrow\; \bot$$

We **cannot** prove

$$\begin{cases} & \Longrightarrow \; \boxed{(\mathit{even}\ \mathtt{z})} \\ \mathit{nat}\ y & \Longrightarrow \; (\mathit{even}\ (\mathtt{suc}\ y)) \lor (\mathit{even}\ (\mathtt{suc}\ (\mathtt{suc}\ y))) \end{cases}$$

# Unfolding definitions

The logic comes with *unfolding rules* for definitions.
Unfolding a goal formula corresponds to **pattern matching**:

$$\Longrightarrow \; \mathit{even}\ (\mathtt{suc}\ (\mathtt{suc}\ \mathtt{z}))$$

Unfolding a hypothesis corresponds to **case analysis**:

$$\mathit{nat}\ x \;\Longrightarrow\; (x = \mathtt{z}) \lor \exists y.(x = \mathtt{suc}\ y) \land (\mathit{nat}\ y)$$

$$\mathit{even}\ (\mathtt{suc}\ z) \;\Longrightarrow\; \bot$$

We **cannot** prove

$$\begin{cases} & \Longrightarrow\; \top \\ \mathit{nat}\ y & \Longrightarrow\; (\mathit{even}\ (\mathtt{suc}\ y)) \lor (\mathit{even}\ (\mathtt{suc}\ (\mathtt{suc}\ y))) \end{cases}$$

# Unfolding definitions

The logic comes with *unfolding rules* for definitions.
Unfolding a goal formula corresponds to **pattern matching**:

$$\Longrightarrow \ \mathit{even}\ (\mathtt{suc}\ (\mathtt{suc}\ \mathtt{z}))$$

Unfolding a hypothesis corresponds to **case analysis**:

$$\mathit{nat}\ x \ \Longrightarrow\ (x = \mathtt{z}) \lor \exists y.(x = \mathtt{suc}\ y) \land (\mathit{nat}\ y)$$

$$\mathit{even}\ (\mathtt{suc}\ z)\ \Longrightarrow\ \bot$$

We **cannot** prove

$$\begin{cases} & \Longrightarrow\ \top \quad ✅ \\ \mathit{nat}\ y & \Longrightarrow\ (\mathit{even}\ (\mathtt{suc}\ y)) \lor (\mathit{even}\ (\mathtt{suc}\ (\mathtt{suc}\ y))) \end{cases}$$

# Unfolding definitions

The logic comes with *unfolding rules* for definitions.
Unfolding a goal formula corresponds to **pattern matching**:

$$\Longrightarrow \; \mathit{even}\ (\mathtt{suc}\ (\mathtt{suc}\ \mathtt{z}))$$

Unfolding a hypothesis corresponds to **case analysis**:

$$\mathit{nat}\ x \Longrightarrow (x = \mathtt{z}) \vee \exists y.(x = \mathtt{suc}\ y) \wedge (\mathit{nat}\ y)$$

$$\mathit{even}\ (\mathtt{suc}\ z) \Longrightarrow \bot$$

We **cannot** prove

$$\begin{cases} & \Longrightarrow \top \quad \text{✅} \\ \mathit{nat}\ y & \Longrightarrow (\mathit{even}\ (\mathtt{suc}\ y)) \vee \boxed{(\mathit{even}\ (\mathtt{suc}\ (\mathtt{suc}\ y)))} \end{cases}$$

# Unfolding definitions

The logic comes with *unfolding rules* for definitions.
Unfolding a goal formula corresponds to **pattern matching**:

$$\Longrightarrow \quad \mathit{even}\ (\mathtt{suc}\ (\mathtt{suc}\ \mathtt{z}))$$

Unfolding a hypothesis corresponds to **case analysis**:

$$\mathit{nat}\ x \Longrightarrow (x = \mathtt{z}) \vee \exists y.(x = \mathtt{suc}\ y) \wedge (\mathit{nat}\ y)$$

$$\mathit{even}\ (\mathtt{suc}\ z) \Longrightarrow \bot$$

We **cannot** prove

$$\begin{cases} & \Longrightarrow \top \quad ✅ \\ \mathit{nat}\ y & \Longrightarrow (\mathit{even}\ (\mathtt{suc}\ y)) \vee (\mathit{even}\ y) \end{cases}$$

# Unfolding definitions

The logic comes with *unfolding rules* for definitions.
Unfolding a goal formula corresponds to **pattern matching**:

$$\Longrightarrow \; \textit{even}\;(\texttt{suc}\;(\texttt{suc}\;\texttt{z}))$$

Unfolding a hypothesis corresponds to **case analysis**:

$$\textit{nat}\; x \;\Longrightarrow\; (x = \texttt{z}) \vee \exists y.(x = \texttt{suc}\; y) \wedge (\textit{nat}\;\; y)$$

$$\textit{even}\;(\texttt{suc}\; z) \;\Longrightarrow\; \bot$$

We **cannot** prove

$$\begin{cases} & \Longrightarrow \; \top \quad ✅ \\ \textit{nat}\;\; y & \Longrightarrow \; (\textit{even}\;(\texttt{suc}\;\; y)) \vee (\textit{even}\;\; y) \quad ❌ \end{cases}$$

To do this, we need *nat* to be an *inductive definition*!

# Inductive definitions

A predicate $S$ is an *inductive invariant* for $nat$ if it satisfies

$$\Longrightarrow S\ \mathtt{z}$$

$$S\ x \Longrightarrow S\ (\mathtt{suc}\ x)$$

# Inductive definitions

A predicate $S$ is an *inductive invariant* for $nat$ if it satisfies

$$\Longrightarrow S\ \mathtt{z}$$

$$S\ x \Longrightarrow S\ (\mathtt{suc}\ x)$$

The *inductive rule* for $nat$ allows us to conclude

$$nat\ x \Longrightarrow S\ x$$

# Inductive definitions

A predicate $S$ is an *inductive invariant* for $nat$ if it satisfies

$$\implies S\ \mathtt{z}$$

$$S\ x \implies S\ (\mathtt{suc}\ x)$$

The *inductive rule* for $nat$ allows us to conclude

$$nat\ x \implies S\ x$$

We can show the following is an invariant for $nat$

$$S := \lambda x.(even\ x) \vee (even\ (\mathtt{suc}\ x))$$

# Inductive definitions

A predicate $S$ is an *inductive invariant* for $nat$ if it satisfies

$$\Longrightarrow S\ \mathtt{z}$$

$$S\ x \Longrightarrow S\ (\mathtt{suc}\ x)$$

The *inductive rule* for $nat$ allows us to conclude

$$nat\ x \Longrightarrow (even\ x) \lor (even\ (\mathtt{suc}\ x))$$

We can show the following is an invariant for $nat$

$$S := \lambda x.(even\ x) \lor (even\ (\mathtt{suc}\ x))$$

# Inductive definitions

An *inductive* definition for $p$ is a set of clauses

$$p\ \vec{t}_1 \stackrel{\mu}{=} B_1 \qquad \ldots \qquad p\ \vec{t}_n \stackrel{\mu}{=} B_n$$

# Inductive definitions

An *inductive* definition for $p$ is a set of clauses

$$p\ \vec{t}_1 \stackrel{\mu}{=} B_1 \qquad \dots \qquad p\ \vec{t}_n \stackrel{\mu}{=} B_n$$

An *inductive invariant* $S$ for $p$ is a predicate satisfying

$$B_i[S/p] \implies S\ \vec{t}_i$$

for each clause $p\ \vec{t}_i \stackrel{\mu}{=} B_i$.

# Inductive definitions

An *inductive* definition for $p$ is a set of clauses

$$p\ \vec{t}_1 \stackrel{\mu}{=} B_1 \qquad \dots \qquad p\ \vec{t}_n \stackrel{\mu}{=} B_n$$

An *inductive invariant* $S$ for $p$ is a predicate satisfying

$$B_i[S/p] \implies S\ \vec{t}_i$$

for each clause $p\ \vec{t}_i \stackrel{\mu}{=} B_i$.

The *induction rule* says that whenever

$$\Gamma, S\ \vec{t} \implies C$$

then

$$\Gamma, p\ \vec{t} \implies C$$

# Stratification

If we allow *any* definition, eg.

$$p \triangleq p \supset \bot$$

we can prove $\vdash \bot$, which makes the logic **inconsistent**!

# Stratification

If we allow *any* definition, eg.

$$p \triangleq p \supset \bot$$

we can prove $\vdash \bot$, which makes the logic **inconsistent**!

### Definition (Stratification)

A collection of definitions is *stratified* if predicates can be ordered such that those appearing to the left of an implication in the body $B$ of a definition $p\ \vec{t} \triangleq B$ have already been defined.

# Stratification

If we allow *any* definition, eg.

$$p \triangleq p \supset \bot$$

we can prove $\vdash \bot$, which makes the logic **inconsistent**!

**Definition (Stratification)**

A collection of definitions is *stratified* if predicates can be ordered such that those appearing to the left of an implication in the body $B$ of a definition $p\ \vec{t} \triangleq B$ have already been defined.

**Eg.** The following can be stratified:

$$q \triangleq q \qquad\qquad p \triangleq q \supset \bot$$

# Ground stratification

Suppose we've defined a *reduction relation* on *simply-typed* $\lambda$-terms:

$$\mathit{step}\ T\ U$$

# Ground stratification

Suppose we've defined a *reduction relation* on *simply-typed* $\lambda$-terms:

$$\mathit{step}\ \ T\ \ U$$

We define *strong normalizability*:

$$\mathit{sn}\ \ T \stackrel{\mu}{=} \forall u.(\mathit{step}\ \ T\ \ u) \supset (\mathit{sn}\ \ u)$$

# Ground stratification

Suppose we've defined a *reduction relation* on *simply-typed* $\lambda$-terms:

$$step\ T\ U$$

We define *strong normalizability*:

$$sn\ T \stackrel{\mu}{=} \forall u.(step\ T\ u) \supset (sn\ u)$$

Now define *reducibility*:

$$red\ \mathtt{unit}\ t \stackrel{\Delta}{=} sn\ t$$
$$red\ (A \Rightarrow B)\ t \stackrel{\Delta}{=} \forall u.(red\ A\ u) \supset (red\ B\ (\mathtt{app}\ t\ u))$$

# Ground stratification

Suppose we've defined a *reduction relation* on *simply-typed* $\lambda$-terms:

$$\mathit{step}\ T\ U$$

We define *strong normalizability*:

$$\mathit{sn}\ T \stackrel{\mu}{=} \forall u.(\mathit{step}\ T\ u) \supset (\mathit{sn}\ u)$$

Now define *reducibility*:

$$\mathit{red}\ \mathtt{unit}\ t \triangleq \mathit{sn}\ t$$
$$\mathit{red}\ (A \Rightarrow B)\ t \triangleq \forall u.(\mathit{red}\ A\ u) \supset (\mathit{red}\ B\ (\mathtt{app}\ t\ u))$$

**Remark:** *red* is *not* stratified

# Ground stratification

Suppose we've defined a *reduction relation* on *simply-typed* $\lambda$-terms:

$$\mathit{step}\ T\ U$$

We define *strong normalizability*:

$$\mathit{sn}\ T \stackrel{\mu}{=} \forall u.(\mathit{step}\ T\ u) \supset (\mathit{sn}\ u)$$

Now define *reducibility*:

$$\mathit{red}\ \mathtt{unit}\ t \stackrel{\Delta}{=} \mathit{sn}\ t$$
$$\mathit{red}\ (A \Rightarrow B)\ t \stackrel{\Delta}{=} \forall u.(\mathit{red}\ A\ u) \supset (\mathit{red}\ B\ (\mathtt{app}\ t\ u))$$

**Remark:** *red* is *not* stratified; it is defined by *recursion* on the type argument. Tiu [2012] called this **ground stratification** and showed that such definitions are consistent.

# Can inductive definitions be ground stratified?

What if we want *red* to be inductive?

$$\mathit{red}\ \mathtt{unit}\ t \triangleq \mathit{sn}\ t$$
$$\mathit{red}\ (A \Rightarrow B)\ t \triangleq \forall u.(\mathit{red}\ A\ u) \supset (\mathit{red}\ B\ (\mathtt{app}\ t\ u))$$

# Can inductive definitions be ground stratified?

What if we want *red* to be inductive?

$$\mathit{red}\ \mathtt{unit}\ t \ \boxed{\stackrel{\mu}{=}}\ \mathit{sn}\ t$$
$$\mathit{red}\ (A \Rightarrow B)\ t \ \boxed{\stackrel{\mu}{=}}\ \forall u.(\mathit{red}\ A\ u) \supset (\mathit{red}\ B\ (\mathtt{app}\ t\ u))$$

# Can inductive definitions be ground stratified?

What if we want *red* to be inductive?

$$\mathit{red}\ \mathtt{unit}\ t \boxed{\stackrel{\mu}{=}} \mathit{sn}\ t$$
$$\mathit{red}\ (A \Rightarrow B)\ t \boxed{\stackrel{\mu}{=}} \forall u.(\mathit{red}\ A\ u) \supset (\mathit{red}\ B\ (\mathtt{app}\ t\ u))$$

With the current induction rule, ground stratified inductive definitions may lead to inconsistencies [GN 2025][1]!

[1] *Ground stratification for a logic of definitions with induction*

# Can inductive definitions be ground stratified?

What if we want *red* to be inductive?

$$\begin{aligned} red\ \mathtt{unit}\ t &\ \boxed{\stackrel{\mu}{=}}\ sn\ t \\ red\ (A \Rightarrow B)\ t &\ \boxed{\stackrel{\mu}{=}}\ \forall u.(red\ A\ u) \supset (red\ B\ (\mathtt{app}\ t\ u)) \end{aligned}$$

With the current induction rule, ground stratified inductive definitions may lead to inconsistencies [GN 2025][1]!
**We must therefore modify the induction rule!**

[1] *Ground stratification for a logic of definitions with induction*

# Ground stratified inductive definitions

An *inductive invariant* $S$ for $p$ is a predicate such that for each clause,

$$B_i[S/p] \implies S\ \vec{t}_i$$

# Ground stratified inductive definitions

**New definition:**
An *inductive invariant* $S$ for $p$ is a predicate such that for each clause,

$$B_i^+ \; S \implies S \; \vec{t}_i$$

where $B^+$ is obtained from a formula $B$ by abstracting all *strictly positive* occurrences of $p$, eg.

$$((p \; \vec{t}_1 \supset p \; \vec{t}_2) \supset p \; \vec{t}_3)^+ := \lambda S.((p \; \vec{t}_1 \supset p \; \vec{t}_2) \supset S \; \vec{t}_3)$$

# Ground stratified inductive definitions

**New definition:**
An *inductive invariant* $S$ for $p$ is a predicate such that for each clause,

$$B_i^+ \ S \implies S \ \vec{t}_i$$

where $B^+$ is obtained from a formula $B$ by abstracting all *strictly positive* occurrences of $p$, eg.

$$((p \ \vec{t}_1 \supset p \ \vec{t}_2) \supset p \ \vec{t}_3)^+ := \lambda S.((p \ \vec{t}_1 \supset p \ \vec{t}_2) \supset S \ \vec{t}_3)$$

If $p$ is stratified, then $B[S/p] = B^+ \ S$, so this subsumes previous notion of inductive invariant.

# Consistency

**Theorem (Consistency)**

*The logic of definitions with ground stratified definitions and ground stratified inductive definitions (with the new induction rule) is consistent.*

# Consistency

**Theorem (Consistency)**

*The logic of definitions with ground stratified definitions and ground stratified inductive definitions (with the new induction rule) is consistent.*

The key lemma needed is:

**Lemma (Unfolding lemma)**

*For any formula $C$, inductive definition $p$, and inductive invariant $S$, the following holds*

$$C \implies C^{+}\ S.$$

# Example Application

**Application:** Strong normalization for *System T:*

Simply typed $\lambda$-calculus $+$ natural number recursion

# Example Application

**Application:** Strong normalization for *System T:*

Simply typed $\lambda$-calculus $+$ natural number recursion

We adapt Martin-Löf's [1971] *computability predicate* to System T:

$$
\begin{aligned}
\mathit{red}\ \mathtt{nat}\ \mathtt{z} &\stackrel{\mu}{=} \top \\
\mathit{red}\ \mathtt{nat}\ (\mathtt{suc}\ X) &\stackrel{\mu}{=} \mathit{red}\ \mathtt{nat}\ X \\
\mathit{red}\ (A \Rightarrow B)\ (\mathtt{lam}\ S) &\stackrel{\mu}{=} \forall u.(\mathit{red}\ A\ u) \supset (\mathit{red}\ B\ (S\ u)) \\
\mathit{red}\ B\ T &\stackrel{\mu}{=} (\mathit{neutral}\ T) \wedge \forall u.(\mathit{step}\ T\ u) \supset \mathit{red}\ B\ u
\end{aligned}
$$

# Example Application

**Application:** Strong normalization for *System T:*

Simply typed $\lambda$-calculus $+$ natural number recursion

We adapt Martin-Löf's [1971] *computability predicate* to System T:

$$
\begin{aligned}
\mathit{red}\ \mathtt{nat}\ \mathtt{z} &\stackrel{\mu}{=} \top \\
\mathit{red}\ \mathtt{nat}\ (\mathtt{suc}\ X) &\stackrel{\mu}{=} \mathit{red}\ \mathtt{nat}\ X \\
\mathit{red}\ (A \Rightarrow B)\ (\mathtt{lam}\ S) &\stackrel{\mu}{=} \forall u.(\mathit{red}\ A\ u) \supset (\mathit{red}\ B\ (S\ u)) \\
\mathit{red}\ B\ T &\stackrel{\mu}{=} (\mathit{neutral}\ T) \wedge \forall u.(\mathit{step}\ T\ u) \supset \mathit{red}\ B\ u
\end{aligned}
$$

Strong normalization requires induction on reducibility

# Conclusion, related, and future work

**Conclusion:** The logic of definitions with ground stratified definitions is compatible with ground stratified inductive definitions.

# Conclusion, related, and future work

**Conclusion:** The logic of definitions with ground stratified definitions is compatible with ground stratified inductive definitions.

**Future directions:**

1. Show that ground stratified induction is compatible with other features of the *Abella* proof assistant

# Conclusion, related, and future work

**Conclusion:** The logic of definitions with ground stratified definitions is compatible with ground stratified inductive definitions.

**Future directions:**

1. Show that ground stratified induction is compatible with other features of the *Abella* proof assistant

2. Relate ground stratified inductive definitions in the logic of definitions with *inductive-recursive* definitions in type theory

# Conclusion, related, and future work

**Conclusion:** The logic of definitions with ground stratified definitions is compatible with ground stratified inductive definitions.

**Future directions:**

1. Show that ground stratified induction is compatible with other features of the *Abella* proof assistant

2. Relate ground stratified inductive definitions in the logic of definitions with *inductive-recursive* definitions in type theory

3. Generalize the induction rule to work with all (non-strictly) *positive* inductive invariants (ie. $B^+$ should abstract all positive occurrences of $p$)

# Conclusion, related, and future work

**Conclusion:** The logic of definitions with ground stratified definitions is compatible with ground stratified inductive definitions.

**Future directions:**

1. Show that ground stratified induction is compatible with other features of the *Abella* proof assistant

2. Relate ground stratified inductive definitions in the logic of definitions with *inductive-recursive* definitions in type theory

3. Generalize the induction rule to work with all (non-strictly) *positive* inductive invariants (ie. $B^+$ should abstract all positive occurrences of $p$)

**Thank you!**