Different Notions of Ordinals in Homotopy Type Theory

Fredrik Nordvall Forsberg
University of Strathclyde

Joint work with Nicolai Kraus and Chuangjie Xu

HOTTEST seminar

3 March 2022

What are ordinals?

"Numbers" for ranking/ordering:

$$0,\ 1,\ 2,\ \ldots,\ \omega,\ \omega+1,\ \ldots,\ \omega\cdot 2,\ \omega\cdot 2+1,\ \ldots,\ \omega\cdot 3,\ \ldots$$

$$\omega^2,\ \ldots,\ \omega^2\cdot 3+\omega\cdot 7+13,\ \ldots,\ \omega^\omega,\ \ldots,\ \varepsilon_0=\omega^{\omega^{\omega^{\cdots}}},\ \ldots,\ \varepsilon_{17},\ \ldots$$

What are ordinals?

"Numbers" for ranking/ordering:

$$0,\ 1,\ 2,\ \ldots,\ \omega,\ \omega+1,\ \ldots,\ \omega\cdot 2,\ \omega\cdot 2+1,\ \ldots,\ \omega\cdot 3,\ \ldots$$

$$\omega^2,\ \ldots,\ \omega^2\cdot 3+\omega\cdot 7+13,\ \ldots,\ \omega^\omega,\ \ldots,\ \varepsilon_0=\omega^{\omega^{\omega^{\cdots}}},\ \ldots,\ \varepsilon_{17},\ \ldots$$

Classically: sets with an order $<$, which is

- **transitive**: $(a<b)\to(b<c)\to(a<c)$
- **wellfounded**: every sequence $a_0>a_1>a_2>a_3>\ldots$ terminates
- and **trichotomous**: $(a<b)\vee(a=b)\vee(b<a)$

What are ordinals?

"Numbers" for ranking/ordering:

$$0,\quad 1,\quad 2,\quad \ldots,\quad \omega,\quad \omega+1,\quad \ldots,\quad \omega\cdot 2,\quad \omega\cdot 2+1,\quad \ldots,\quad \omega\cdot 3,\quad \ldots$$

$$\omega^2,\quad \ldots,\quad \omega^2\cdot 3+\omega\cdot 7+13,\quad \ldots,\quad \omega^\omega,\quad \ldots,\quad \varepsilon_0=\omega^{\omega^{\omega^{\cdots}}},\quad \ldots,\quad \varepsilon_{17},\quad \ldots$$

Classically: sets with an order $<$, which is

- **transitive**: $(a<b)\to(b<c)\to(a<c)$
- **wellfounded**: every sequence $a_0>a_1>a_2>a_3>\ldots$ terminates
- and **trichotomous**: $(a<b)\vee(a=b)\vee(b<a)$
- ... or **extensional** (instead of trichotomous):

$$(\forall a. a<b \leftrightarrow a<c)\to b=c$$

What are ordinals?

"Numbers" for ranking/ordering:

$$0, \quad 1, \quad 2, \quad \ldots, \quad \omega, \quad \omega+1, \quad \ldots, \quad \omega\cdot 2, \quad \omega\cdot 2+1, \quad \ldots, \quad \omega\cdot 3, \quad \ldots$$

$$\omega^2, \quad \ldots, \quad \omega^2\cdot 3+\omega\cdot 7+13, \quad \ldots, \quad \omega^\omega, \quad \ldots, \quad \varepsilon_0 = \omega^{\omega^{\omega^{\cdots}}}, \quad \ldots, \quad \varepsilon_{17}, \quad \ldots$$

Classically: sets with an order $<$, which is

- **transitive**: $(a < b) \to (b < c) \to (a < c)$
- **wellfounded**: every sequence $a_0 > a_1 > a_2 > a_3 > \ldots$ terminates
- and **trichotomous**: $(a < b) \vee (a = b) \vee (b < a)$
- . . . or **extensional** (instead of trichotomous):

$$(\forall a. a < b \leftrightarrow a < c) \to b = c$$

Perhaps more importantly: what are they for?

Transfinite iteration

Let $F : \mathsf{Set} \to \mathsf{Set}$ be a finitary functor.

Transfinite iteration

Let F : Set $\to$ Set be a finitary functor.

The initial algebra of F can be constructed as the colimit of the sequence

$$X_0 \longrightarrow X_1 \longrightarrow X_2 \longrightarrow \ldots$$

where

Transfinite iteration

Let F : Set $\to$ Set be a finitary functor.

The initial algebra of F can be constructed as the colimit of the sequence

$$X_0 \longrightarrow X_1 \longrightarrow X_2 \longrightarrow \dots$$

where

$$X_0 = \emptyset$$

Transfinite iteration

Let F : Set $\to$ Set be a finitary functor.

The initial algebra of F can be constructed as the colimit of the sequence

$$X_0 \longrightarrow X_1 \longrightarrow X_2 \longrightarrow \dots$$

where

$$X_0 = \emptyset$$
$$X_{n+1} = F(X_n)$$

Transfinite iteration

Let F : Set $\to$ Set be a finitary functor.

The initial algebra of F can be constructed as the colimit of the sequence

$$X_0 \xrightarrow{!} X_1 \longrightarrow X_2 \longrightarrow \dots$$

where

$$\begin{aligned} X_0 &= \emptyset \\ X_{n+1} &= F(X_n) \end{aligned}$$

Transfinite iteration

Let $F : \mathsf{Set} \to \mathsf{Set}$ be a finitary functor.

The initial algebra of F can be constructed as the colimit of the sequence

$$X_0 \xrightarrow{\;!\;} X_1 \xrightarrow{F(!)} X_2 \longrightarrow \dots$$

where

$$X_0 = \emptyset$$
$$X_{n+1} = F(X_n)$$

Transfinite iteration

Let F : Set $\to$ Set be a finitary functor.

The initial algebra of F can be constructed as the colimit of the sequence

$$X_0 \xrightarrow{!} X_1 \xrightarrow{F(!)} X_2 \xrightarrow{F^2(!)} \dots$$

where

$$X_0 = \emptyset$$
$$X_{n+1} = F(X_n)$$

Transfinite iteration

Let F : Set $\to$ Set be a finitary functor.

The initial algebra of F can be constructed as the colimit of the sequence

$$X_0 \xrightarrow{!} X_1 \xrightarrow{F(!)} X_2 \xrightarrow{F^2(!)} \dots$$

where

$$X_0 = \emptyset$$
$$X_{\alpha+1} = F(X_\alpha)$$

Transfinite iteration

Let F : Set $\to$ Set be a finitary functor.

The initial algebra of F can be constructed as the colimit of the sequence

$$X_0 \xrightarrow{!} X_1 \xrightarrow{F(!)} X_2 \xrightarrow{F^2(!)} \dots \longrightarrow X_\omega$$

where

$$X_0 = \emptyset$$
$$X_{\alpha+1} = F(X_\alpha)$$
$$\mu F = X_\omega = \mathrm{colim}_{\beta<\omega} X_\beta$$

Transfinite iteration

Let F : Set $\to$ Set be a functor preserving κ-colimits.

The initial algebra of F can be constructed as the colimit of the sequence

$$X_0 \xrightarrow{!} X_1 \xrightarrow{F(!)} X_2 \xrightarrow{F^2(!)} \dots \longrightarrow X_\omega \longrightarrow X_{\omega+1} \longrightarrow \dots$$

where

$$\begin{aligned} X_0 &= \emptyset \\ X_{\alpha+1} &= F(X_\alpha) \\ X_\omega &= \mathrm{colim}_{\beta<\omega} X_\beta \end{aligned}$$

Transfinite iteration

Let $F : \mathsf{Set} \to \mathsf{Set}$ be a functor preserving κ-colimits.

The initial algebra of F can be constructed as the colimit of the sequence

$$X_0 \xrightarrow{!} X_1 \xrightarrow{F(!)} X_2 \xrightarrow{F^2(!)} \dots \longrightarrow X_\omega \longrightarrow X_{\omega+1} \longrightarrow \dots$$

where

$$\begin{aligned} X_0 &= \emptyset \\ X_{\alpha+1} &= F(X_\alpha) \\ X_\lambda &= \mathsf{colim}_{\beta<\lambda} X_\beta \end{aligned}$$

Transfinite iteration

Let $F : \mathsf{Set} \to \mathsf{Set}$ be a functor preserving κ-colimits.

The initial algebra of F can be constructed as the colimit of the sequence

$$X_0 \xrightarrow{\;!\;} X_1 \xrightarrow{F(!)} X_2 \xrightarrow{F^2(!)} \dots \longrightarrow X_\omega \longrightarrow X_{\omega+1} \longrightarrow \dots \longrightarrow X_\kappa$$

where

$$\begin{aligned} X_0 &= \emptyset \\ X_{\alpha+1} &= F(X_\alpha) \\ X_\lambda &= \mathsf{colim}_{\beta<\lambda} X_\beta \\ \mu F &= X_\kappa \end{aligned}$$

Transfinite iteration

Let $F : \mathsf{Set} \to \mathsf{Set}$ be a functor preserving κ-colimits.

The initial algebra of F can be constructed as the colimit of the sequence

$$X_0 \xrightarrow{!} X_1 \xrightarrow{F(!)} X_2 \xrightarrow{F^2(!)} \dots \longrightarrow X_\omega \longrightarrow X_{\omega+1} \longrightarrow \dots \longrightarrow X_\kappa$$

where

$$\begin{aligned} X_0 &= \emptyset \\ X_{\alpha+1} &= F(X_\alpha) \\ X_\lambda &= \mathrm{colim}_{\beta<\lambda} X_\beta \\ \mu F &= X_\kappa \end{aligned}$$

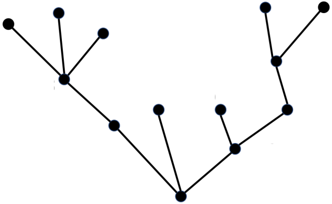
Useful: Definitional principle where ordinals are classified as 0, $\alpha + 1$ or a limit.

Termination of processes

- Programs terminating [Turing 1949]
- Consistency proof e.g. of Peano's axioms [Gentzen 1936]
- Termination of Goodstein sequences [Goodstein 1944], the Hydra game [Kirby&Paris 1982]:

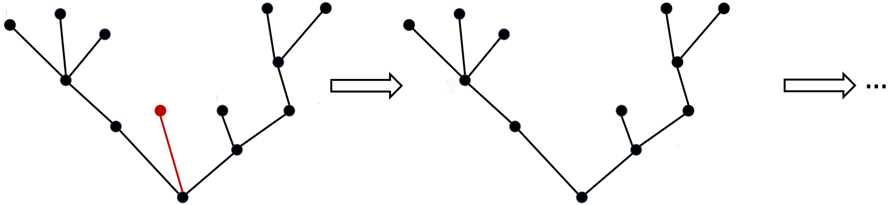
Termination of processes

- Programs terminating [Turing 1949]
- Consistency proof e.g. of Peano's axioms [Gentzen 1936]
- Termination of Goodstein sequences [Goodstein 1944], the Hydra game [Kirby&Paris 1982]:



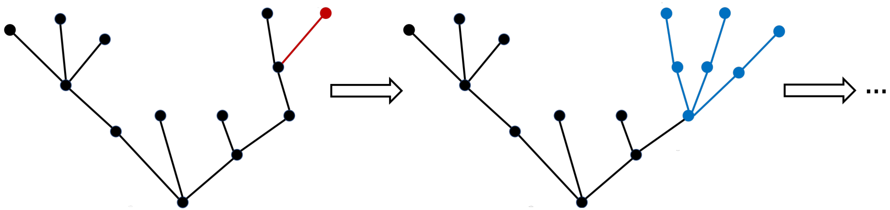
Termination of processes

- Programs terminating [Turing 1949]
- Consistency proof e.g. of Peano's axioms [Gentzen 1936]
- Termination of Goodstein sequences [Goodstein 1944], the Hydra game [Kirby&Paris 1982]:



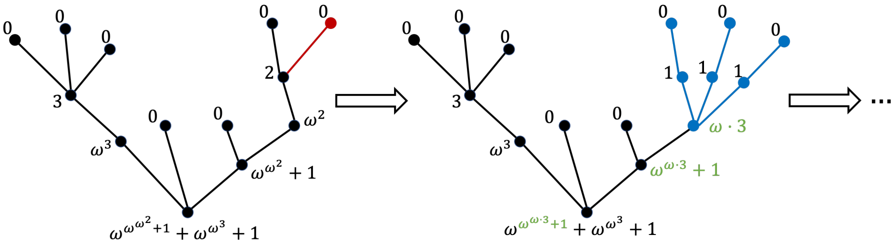
Termination of processes

- Programs terminating [Turing 1949]
- Consistency proof e.g. of Peano's axioms [Gentzen 1936]
- Termination of Goodstein sequences [Goodstein 1944], the Hydra game [Kirby&Paris 1982]:



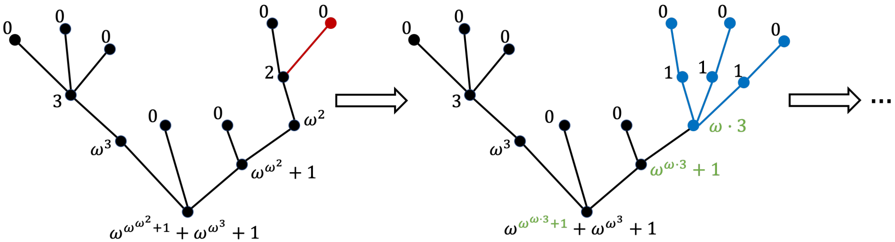
Termination of processes

- Programs terminating [Turing 1949]
- Consistency proof e.g. of Peano's axioms [Gentzen 1936]
- Termination of Goodstein sequences [Goodstein 1944], the Hydra game [Kirby&Paris 1982]:



Termination of processes

- Programs terminating [Turing 1949]
- Consistency proof e.g. of Peano's axioms [Gentzen 1936]
- Termination of Goodstein sequences [Goodstein 1944], the Hydra game [Kirby&Paris 1982]:



Useful: Arithmetic, and every decreasing sequence of ordinals hits 0.

Ordinals in constructive type theory

Problem/feature of a constructive setting: different definitions differ!

Ordinals in constructive type theory

Problem/feature of a constructive setting: different definitions differ!

Classical definition not particularly well suited for either iteration or termination.

Ordinals in constructive type theory

Problem/feature of a constructive setting: different definitions differ!

Classical definition not particularly well suited for either iteration or termination.

Three standard notions of "ordinals" in computer science:

- Cantor normal forms
- Brouwer trees
- Wellfounded, extensional, and transitive orders

How are they connected? Why can we call them "ordinals"?

Need features and concepts of HoTT to give "correct" formulations.

Cantor normal forms

Motivational classical theorem

Every ordinal α can be written uniquely

$$\alpha = \omega^{\beta_1} + \omega^{\beta_2} + \cdots + \omega^{\beta_n}$$

for some $\beta_1 \geq \beta_2 \geq \cdots \geq \beta_n$.

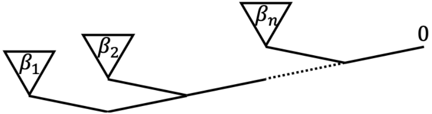
Cantor normal forms

Motivational classical theorem

Every ordinal α can be written uniquely

$$\alpha = \omega^{\beta_1} + \omega^{\beta_2} + \cdots + \omega^{\beta_n}$$

for some $\beta_1 \geq \beta_2 \geq \cdots \geq \beta_n$.

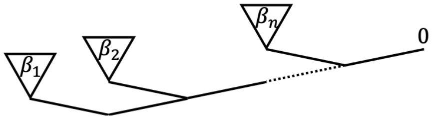
$\alpha =$ 

Cantor normal forms

Motivational classical theorem

Every ordinal α can be written uniquely

$$\alpha = \omega^{\beta_1} + \omega^{\beta_2} + \cdots + \omega^{\beta_n}$$

$\alpha =$ 

for some $\beta_1 \geq \beta_2 \geq \cdots \geq \beta_n$.

Let $\mathcal{T}$ be the type of *unlabeled binary trees*:

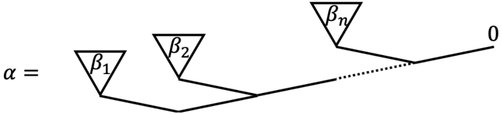
$$\begin{aligned} 0 &: \mathcal{T} \\ \omega^{-} + - &: \mathcal{T} \to \mathcal{T} \to \mathcal{T} \end{aligned}$$

Cantor normal forms

Motivational classical theorem

Every ordinal α can be written uniquely

$$\alpha = \omega^{\beta_1} + \omega^{\beta_2} + \cdots + \omega^{\beta_n}$$



for some $\beta_1 \geq \beta_2 \geq \cdots \geq \beta_n$.

Let $\mathcal{T}$ be the type of *unlabeled binary trees*:

leaf

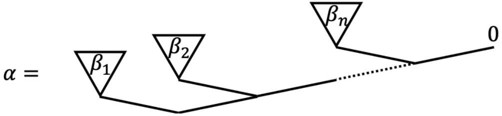
$$\begin{aligned} 0 &: \mathcal{T} \\ \omega^{-} + - &: \mathcal{T} \to \mathcal{T} \to \mathcal{T} \end{aligned}$$

Cantor normal forms

Motivational classical theorem

Every ordinal α can be written uniquely

$$\alpha = \omega^{\beta_1} + \omega^{\beta_2} + \cdots + \omega^{\beta_n}$$



for some $\beta_1 \geq \beta_2 \geq \cdots \geq \beta_n$.

Let $\mathcal{T}$ be the type of *unlabeled binary trees*:

node

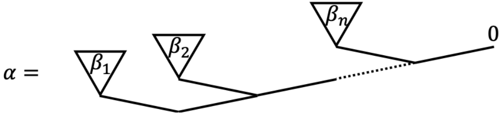
$$\begin{aligned} 0 &: \mathcal{T} \\ \omega^{-} + - &: \mathcal{T} \to \mathcal{T} \to \mathcal{T} \end{aligned}$$

Cantor normal forms

Motivational classical theorem

Every ordinal α can be written uniquely

$$\alpha = \omega^{\beta_1} + \omega^{\beta_2} + \cdots + \omega^{\beta_n}$$



for some $\beta_1 \geq \beta_2 \geq \cdots \geq \beta_n$.

Let $\mathcal{T}$ be the type of *unlabeled binary trees*:

$$\begin{aligned} 0 \quad &: \mathcal{T} \\ \omega^{-} + - \quad &: \mathcal{T} \to \mathcal{T} \to \mathcal{T} \end{aligned}$$

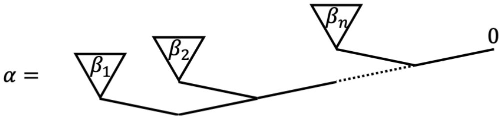
Define isCNF(α) to express $\beta_1 \geq \beta_2 \geq \cdots \geq \beta_n$ (*lexicographical order*).

Cantor normal forms

Motivational classical theorem

Every ordinal α can be written uniquely

$$\alpha = \omega^{\beta_1} + \omega^{\beta_2} + \cdots + \omega^{\beta_n}$$



for some $\beta_1 \geq \beta_2 \geq \cdots \geq \beta_n$.

Let $\mathcal{T}$ be the type of *unlabeled binary trees*:

$$\begin{aligned} 0 \quad &: \mathcal{T} \\ \omega^{-} + - \quad &: \mathcal{T} \to \mathcal{T} \to \mathcal{T} \end{aligned}$$

Define isCNF(α) to express $\beta_1 \geq \beta_2 \geq \cdots \geq \beta_n$ (*lexicographical order*).

We write Cnf $= (\Sigma\alpha : \mathcal{T})$ isCNF(α) for the type of Cantor Normal Forms.

Basic properties of Cantor Normal Forms

Equivalent implementations [Ghani, N.-F., Xu 2020]:

(i) inductive-inductively inlining the isCNF condition (no junk!)

(ii) as finite hereditary multisets.

Basic properties of Cantor Normal Forms

Equivalent implementations [Ghani, N.-F., Xu 2020]:

(i) inductive-inductively inlining the isCNF condition (no junk!)

(ii) as finite hereditary multisets.

Basic properties of Cantor Normal Forms

Equivalent implementations [Ghani, N.-F., Xu 2020]:

(i) inductive-inductively inlining the isCNF condition (no junk!)

(ii) as finite hereditary multisets.

Theorem: $<$ is trichotomous, i.e. have $<$-tri : $(x, y : \mathsf{Cnf}) \to (x < y) \uplus (x \geq y)$.

Basic properties of Cantor Normal Forms

Equivalent implementations [Ghani, N.-F., Xu 2020]:

(i) inductive-inductively inlining the isCNF condition (no junk!)

(ii) as finite hereditary multisets.

Theorem: $<$ is trichotomous, i.e. have $<$-tri : $(x, y : \mathsf{Cnf}) \to (x < y) \uplus (x \geq y)$.

Corollary: Cnf has decidable equality.

Basic properties of Cantor Normal Forms

Equivalent implementations [Ghani, N.-F., Xu 2020]:

(i) inductive-inductively inlining the isCNF condition (no junk!)

(ii) as finite hereditary multisets.

Theorem: $<$ is trichotomous, i.e. have $<$-tri : $(x, y : \mathsf{Cnf}) \to (x < y) \uplus (x \geq y)$.

Corollary: Cnf has decidable equality.

Theorem: Transfinite induction holds for Cnf, i.e. there is a proof

$$\mathsf{TI} : (P : \mathsf{Cnf} \to \mathsf{Type}\,\ell) \to (\forall x.(\forall y < x.P\,y) \to P\,x) \to \forall x.P\,x$$

Basic properties of Cantor Normal Forms

Equivalent implementations [Ghani, N.-F., Xu 2020]:

(i) inductive-inductively inlining the isCNF condition (no junk!)

(ii) as finite hereditary multisets.

Theorem: $<$ is trichotomous, i.e. have $<$-tri : $(x, y : \mathsf{Cnf}) \to (x < y) \uplus (x \geq y)$.

Corollary: Cnf has decidable equality.

Theorem: Transfinite induction holds for Cnf, i.e. there is a proof

$$\mathsf{TI} : (P : \mathsf{Cnf} \to \mathsf{Type}\,\ell) \to (\forall x.(\forall y < x. P\,y) \to P\,x) \to \forall x. P\,x$$

Theorem: Can classify each Cnf as zero, successor or limit, but cannot compute limits (implies WLPO).

Brouwer ordinal trees

Another definition: the usual inductive type $\mathcal{O}$ generated by

$$\mathsf{zero} : \mathcal{O} \qquad \mathsf{succ} : \mathcal{O} \to \mathcal{O} \qquad \mathsf{sup} : (\mathbb{N} \to \mathcal{O}) \to \mathcal{O}$$

Brouwer ordinal trees

Another definition: the usual inductive type $\mathcal{O}$ generated by

$$\mathsf{zero} : \mathcal{O} \qquad \mathsf{succ} : \mathcal{O} \to \mathcal{O} \qquad \mathsf{sup} : (\mathbb{N} \to \mathcal{O}) \to \mathcal{O}$$

Problem:

$$\mathsf{sup}\,(0, 1, 2, 3, \ldots) \neq \mathsf{sup}\,(1, 2, 3, \ldots)$$

Brouwer ordinal trees

Another definition: the usual inductive type $\mathcal{O}$ generated by

$$\mathsf{zero} : \mathcal{O} \qquad \mathsf{succ} : \mathcal{O} \to \mathcal{O} \qquad \mathsf{sup} : (\mathbb{N} \to \mathcal{O}) \to \mathcal{O}$$

Problem:

$$\mathsf{sup}\,(0, 1, 2, 3, \ldots) \neq \mathsf{sup}\,(1, 2, 3, \ldots)$$

Brouwer ordinal trees

Another definition: the usual inductive type $\mathcal{O}$ generated by

$$\mathsf{zero} : \mathcal{O} \qquad \mathsf{succ} : \mathcal{O} \to \mathcal{O} \qquad \mathsf{sup} : (\mathbb{N} \to \mathcal{O}) \to \mathcal{O}$$

Problem:

$$\mathsf{sup}\,(0, 1, 2, 3, \ldots) \neq \mathsf{sup}\,(1, 2, 3, \ldots)$$
$$\mathsf{sup}\,(0, 1, 2, 3, \ldots) \neq \mathsf{sup}\,(1, 0, 2, 3, \ldots)$$

Brouwer ordinal trees

Another definition: the usual inductive type $\mathcal{O}$ generated by

$$\mathsf{zero} : \mathcal{O} \qquad \mathsf{succ} : \mathcal{O} \to \mathcal{O} \qquad \mathsf{sup} : (\mathbb{N} \to \mathcal{O}) \to \mathcal{O}$$

Problem:

$$\mathsf{sup}\,(0, 1, 2, 3, \ldots) \neq \mathsf{sup}\,(1, 2, 3, \ldots)$$
$$\mathsf{sup}\,(0, 1, 2, 3, \ldots) \neq \mathsf{sup}\,(1, 0, 2, 3, \ldots)$$

Brouwer ordinal trees

Another definition: the usual inductive type $\mathcal{O}$ generated by

$$\mathsf{zero} : \mathcal{O} \qquad \mathsf{succ} : \mathcal{O} \to \mathcal{O} \qquad \mathsf{sup} : (\mathbb{N} \to \mathcal{O}) \to \mathcal{O}$$

Problem:

$$\mathsf{sup}\,(0, 1, 2, 3, \ldots) \neq \mathsf{sup}\,(1, 2, 3, \ldots)$$
$$\mathsf{sup}\,(0, 1, 2, 3, \ldots) \neq \mathsf{sup}\,(1, 0, 2, 3, \ldots)$$

How to fix this without losing wellfoundedness, classification, and so on?

Brouwer trees quotient inductive-inductively

```
data Brw : Set where
  zero  : Brw
  succ  : Brw → Brw
  limit : (f : ℕ → Brw) {f↑ : increasing f} → Brw
  bisim : f ≈ g → limit f ≡ limit g

data _≤_ : Brw → Brw → Prop where
  ≤-zero      : zero ≤ x
  ≤-trans     : x ≤ y → y ≤ z → x ≤ z
  ≤-succ-mono : x ≤ y → succ x ≤ succ y
  ≤-cocone    : x ≤ f k → x ≤ limit f
  ≤-limiting  : (∀ k → f k ≤ x) → limit f ≤ x
```

Brouwer trees quotient inductive-inductively

```
data Brw : Set where
  zero  : Brw
  succ  : Brw → Brw
  limit : (f : ℕ → Brw) {f↑ : increasing f} → Brw
  bisim : f ≈ g → limit f ≡ limit g

data _≤_ : Brw → Brw → Prop where
  ≤-zero      : zero ≤ x
  ≤-trans     : x ≤ y → y ≤ z → x ≤ z
  ≤-succ-mono : x ≤ y → succ x ≤ succ y
  ≤-cocone    : x ≤ f k → x ≤ limit f
  ≤-limiting  : (∀ k → f k ≤ x) → limit f ≤ x
```

$f \approx g = (f \lesssim g) \times (g \lesssim f)$, where $f \lesssim g$ if $\forall i. \exists j. f\, i \leq g\, j$.

Brouwer trees quotient inductive-inductively

```
data Brw : Set where
  zero  : Brw
  succ  : Brw → Brw
  limit : (f : ℕ → Brw) {f↑ : increasing f} → Brw
  bisim : f ≈ g → limit f ≡ limit g

data _≤_ : Brw → Brw → Prop where
  ≤-zero      : zero ≤ x
  ≤-trans     : x ≤ y → y ≤ z → x ≤ z
  ≤-succ-mono : x ≤ y → succ x ≤ succ y
  ≤-cocone    : x ≤ f k → x ≤ limit f
  ≤-limiting  : (∀ k → f k ≤ x) → limit f ≤ x
```

$f \approx g = (f \lesssim g) \times (g \lesssim f)$, where $f \lesssim g$ if $\forall i. \exists j. f\, i \leq g\, j$.

$x < y$ if succ $x \leq y$.

Characterising $\leq$ using encode-decode

Characterising $\leq$ using encode-decode

We use an encode-decode method to characterise $x \leq y$: define

$$\mathsf{Code} : \mathsf{Brw} \to \mathsf{Brw} \to \mathsf{Prop}$$

such that $\mathsf{Code}\, x\, y \;\equiv\; (x \leq y)$.

Characterising $\leq$ using encode-decode

We use an encode-decode method to characterise $x \leq y$: define

$$\mathsf{Code} : \mathsf{Brw} \to \mathsf{Brw} \to \mathsf{Prop}$$

such that $\mathsf{Code}\, x\, y \;\equiv\; (x \leq y)$.

For example:

$$\mathsf{Code}\,(\mathsf{succ}\, x)\,(\mathsf{limit}\, f) = (\exists n : \mathbb{N})\,(\mathsf{Code}\,(\mathsf{succ}\, x)\,(f\, n))$$

Characterising $\leq$ using encode-decode

We use an encode-decode method to characterise $x \leq y$: define

$$\mathsf{Code} : \mathsf{Brw} \to \mathsf{Brw} \to \mathsf{Prop}$$

such that $\mathsf{Code}\, x\, y \ \equiv \ (x \leq y)$.

For example:

$$\mathsf{Code}\,(\mathsf{succ}\, x)\,(\mathsf{limit}\, f) = (\exists n : \mathbb{N})\,(\mathsf{Code}\,(\mathsf{succ}\, x)\,(f\, n))$$

Technically involved: need to simultaneously prove transitivity, reflexivity of Code, and $(x \leq y) \to \mathsf{Code}\, x\, y$.

Basic properties of Brouwer trees

Basic properties of Brouwer trees

Theorem: The order $<$ is wellfounded and extensional.

Basic properties of Brouwer trees

Theorem: The order $<$ is wellfounded and extensional.

Theorem: It is decidable if a Brouwer tree is finite, but decidable (even $\neg\neg$-stable) equality in general implies Markov's Principle.

Basic properties of Brouwer trees

Theorem: The order $<$ is wellfounded and extensional.

Theorem: It is decidable if a Brouwer tree is finite, but decidable (even $\neg\neg$-stable) equality in general implies Markov's Principle.

Can prove expected properties such as:

- $n \cdot \omega \equiv \omega$;
- If $a < \omega^b$ then $a + \omega^b \equiv \omega^b$;
- $\epsilon_0 = \text{limit}\,(\omega, \omega^\omega, \omega^{\omega^\omega}, \omega^{\omega^{\omega^\omega}}, \ldots)$ is a fixed point $\omega^{\epsilon_0} = \epsilon_0$;
- and so on.

Extensional wellfounded orders

The type Ord consists of pairs $(X : \text{Type}, \prec : X \to X \to \text{Prop})$ such that:

- $\prec$ is transitive
- $\prec$ is extensional
- $\prec$ is wellfounded

Can be found in the HoTT book, further developed by Escardó; inspired by Taylor.

Extensional wellfounded orders

The type Ord consists of pairs $(X : \text{Type}, \prec : X \to X \to \text{Prop})$ such that:

- $\prec$ is transitive
  - $x \prec y \to y \prec z \to x \prec z$;
- $\prec$ is extensional
- $\prec$ is wellfounded

Can be found in the HoTT book, further developed by Escardó; inspired by Taylor.

Extensional wellfounded orders

The type Ord consists of pairs $(X : \text{Type}, \prec : X \to X \to \text{Prop})$ such that:

- $\prec$ is transitive
  - $x \prec y \to y \prec z \to x \prec z$;
- $\prec$ is extensional
  - elements with the same $\prec$-predecessors are equal;
- $\prec$ is wellfounded

Can be found in the HoTT book, further developed by Escardó; inspired by Taylor.

Extensional wellfounded orders

The type Ord consists of pairs $(X : \mathsf{Type}, \prec : X \to X \to \mathsf{Prop})$ such that:

- $\prec$ is transitive
  - $x \prec y \to y \prec z \to x \prec z$;
- $\prec$ is extensional
  - elements with the same $\prec$-predecessors are equal;
- $\prec$ is wellfounded
  - every element is accessible, where x is accessible if every $y \prec x$ is accessible.

Can be found in the HoTT book, further developed by Escardó; inspired by Taylor.

Extensional wellfounded orders

The type Ord consists of pairs $(X : \text{Type}, \prec : X \to X \to \text{Prop})$ such that:

- $\prec$ is transitive
  - $x \prec y \to y \prec z \to x \prec z$;
- $\prec$ is extensional
  - elements with the same $\prec$-predecessors are equal;
- $\prec$ is wellfounded
  - every element is accessible, where x is accessible if every $y \prec x$ is accessible.

inductive definition

Can be found in the HoTT book, further developed by Escardó; inspired by Taylor.

The order on extensional wellfounded orders

Let $(X, \prec_X)$, $(Y, \prec_Y)$: Ord.

The order on extensional wellfounded orders

Let $(X, \prec_X)$, $(Y, \prec_Y)$: Ord.

$X \leq Y$ is:

- a *monotone* function $f : X \to Y$
- such that: if $y \prec_Y f\,x$, then there is $x_0 \prec_X x$ such that $f\,x_0 = y$.

Such an f is a *simulation*.

The order on extensional wellfounded orders

Let $(X, \prec_X)$, $(Y, \prec_Y)$: Ord.

$X \leq Y$ is:

- a *monotone* function $f : X \to Y$
- such that: if $y \prec_Y f\,x$, then there is $x_0 \prec_X x$ such that $f\,x_0 = y$.

Such an f is a *simulation*.

For $y : Y$, define $Y_{/y} :\equiv \Sigma(y' : Y).y' \prec y$.

The order on extensional wellfounded orders

Let $(X, \prec_X)$, $(Y, \prec_Y)$: Ord.

$X \leq Y$ is:

- a *monotone* function $f : X \to Y$
- such that: if $y \prec_Y f\,x$, then there is $x_0 \prec_X x$ such that $f\,x_0 = y$.

Such an f is a *simulation*.

For $y : Y$, define $Y_{/y} :\equiv \Sigma(y' : Y).y' \prec y$.

$X < Y$ is:

- a simulation $f : X \leq Y$
- such that there is $y : Y$ and f factors through $X \simeq Y_{/y}$.

$f : X < Y$ is a *bounded simulation*.

Basic properties of extensional wellfounded orders

Basic properties of extensional wellfounded orders

Theorem: the order on Ord is extensional and wellfounded.

Basic properties of extensional wellfounded orders

Theorem: the order on Ord is extensional and wellfounded.

Theorem: limits of increasing sequences of Ord can be calculated.

Basic properties of extensional wellfounded orders

Theorem: the order on Ord is extensional and wellfounded.

Theorem: limits of increasing sequences of Ord can be calculated.

Theorem: “nothing” is decidable.

Basic properties of extensional wellfounded orders

Theorem: the order on Ord is extensional and wellfounded.

Theorem: limits of increasing sequences of Ord can be calculated.

Theorem: "nothing" is decidable.

For example, deciding whether an Ord is a successor implies LEM.

Abstract setting

What do Cnf, Brw, Ord have to do with each other?

Why are they "types of ordinals"?

Abstract setting

What do Cnf, Brw, Ord have to do with each other?

Why are they "types of ordinals"?

Assume we have a set A with relations $<, \leq$ such that:

- $<$ is transitive and irreflexive;
- $\leq$ is transitive, reflexive, and antisymmetric;
- $(<) \subseteq (\leq)$;
- $(< \circ \leq) \subseteq (<)$.

Abstract setting

What do Cnf, Brw, Ord have to do with each other?

Why are they "types of ordinals"?

Assume we have a set A with relations $<, \leq$ such that:

- $<$ is transitive and irreflexive;
- $\leq$ is transitive, reflexive, and antisymmetric;
- $(<) \subseteq (\leq)$, i.e. $x < y \to x \leq y$;
- $(< \circ \leq) \subseteq (<)$.

Abstract setting

What do Cnf, Brw, Ord have to do with each other?

Why are they "types of ordinals"?

Assume we have a set A with relations $<, \leq$ such that:

- $<$ is transitive and irreflexive;
- $\leq$ is transitive, reflexive, and antisymmetric;
- $(<) \subseteq (\leq)$, i.e. $x < y \to x \leq y$;
- $(< \circ \leq) \subseteq (<)$, i.e. $x < y \to y \leq z \to x < z$.

Abstract setting

What do Cnf, Brw, Ord have to do with each other?

Why are they "types of ordinals"?

Assume we have a set A with relations $<, \leq$ such that:

- $<$ is transitive and irreflexive;
- $\leq$ is transitive, reflexive, and antisymmetric;
- $(<) \subseteq (\leq)$, i.e. $x < y \to x \leq y$;
- $(< \circ \leq) \subseteq (<)$, i.e. $x < y \to y \leq z \to x < z$.

Note: $(\leq \circ <) \subseteq (<)$ for Ord is equivalent to LEM (cf. Taylor).

Abstract setting: zero, successor, limit classification

Abstract setting: zero, successor, limit classification

$a : A$ is zero if $\forall b. a \leq b$.

Abstract setting: zero, successor, limit classification

$a : A$ is zero if $\forall b. a \leq b$.

a is a successor of b if
$a > b$ and $\forall x > b.\, x \geq a$.

The successor is strong if
$\forall x < a. x \leq b$.

Abstract setting: zero, successor, limit classification

$a : A$ is zero if $\forall b. a \leq b$.

a is a successor of b if $a > b$ and $\forall x > b.\, x \geq a$.

The successor is strong if $\forall x < a. x \leq b$.

a is a supremum of $f : \mathbb{N} \to A$ if $\forall i. f_i \leq a$ and $(\forall i. f_i \leq x) \to a \leq x$.

a is a limit if f increasing.

Abstract setting: zero, successor, limit classification

$a : A$ is zero if $\forall b. a \leq b$.

a is a successor of b if $a > b$ and $\forall x > b.\, x \geq a$.

The successor is strong if $\forall x < a. x \leq b$.

a is a supremum of $f : \mathbb{N} \to A$ if $\forall i. f_i \leq a$ and $(\forall i. f_i \leq x) \to a \leq x$.

a is a limit if f increasing.

"Concrete" results:

- Cnf, Brw, Ord uniquely have zero and strong successor.
- Brw, Ord uniquely have limits; Cnf does not.
- For Cnf, Brw, we can decide in which case we are ("classification"); for Ord, this would imply LEM.

Abstract setting: zero, successor, limit classification

$a : A$ is zero if $\forall b. a \leq b$.

a is a successor of b if $a > b$ and $\forall x > b.\, x \geq a$.

The successor is strong if $\forall x < a. x \leq b$.

a is a supremum of $f : \mathbb{N} \to A$ if $\forall i. f_i \leq a$ and $(\forall i. f_i \leq x) \to a \leq x$.

a is a limit if f increasing.

"Concrete" results:

- Cnf, Brw, Ord uniquely have zero and strong successor.
- Brw, Ord uniquely have limits; Cnf does not.
- For Cnf, Brw, we can decide in which case we are ("classification"); for Ord, this would imply LEM.

"Abstract" result:

- is-zero$(a) \uplus$ is-str-suc$(a) \uplus$ is-limit(a) is a proposition.

Abstract setting: zero, successor, limit classification

$a : A$ is zero if $\forall b. a \leq b$.

a is a successor of b if $a > b$ and $\forall x > b.\, x \geq a$.

The successor is strong if $\forall x < a. x \leq b$.

a is a supremum of $f : \mathbb{N} \to A$ if $\forall i. f_i \leq a$ and $(\forall i. f_i \leq x) \to a \leq x$.

a is a limit if f increasing.

"Concrete" results:

- Cnf, Brw, Ord uniquely have zero and strong successor.
- Brw, Ord uniquely have limits; Cnf does not.
- For Cnf, Brw, we can decide in which case we are ("classification"); for Ord, this would imply LEM.

"Abstract" result:

- is-zero$(a) \uplus$ is-str-suc$(a) \uplus$ is-limit(a) is a proposition.
- Corollary: "Classifiability" induction implies classification. (Conversely classification + wellfounded induction implies classifiability induction.)

Abstract arithmetic: addition

Abstract arithmetic: addition

$(A, <, \leq)$ has addition if there is a function $+ : A \to A \to A$ such that:

$$\begin{aligned}
&\text{is-zero}(a) \to c + a = c \\
&a \text{ is-suc-of } b \to d \text{ is-suc-of } (c + b) \to c + a = d \\
&a \text{ is-lim-of } f \to b \text{ is-sup-of } (\lambda i.c + f_i) \to c + a = b
\end{aligned}$$

$(A, <, \leq)$ has unique addition if there is exactly one function with these properties.

Abstract arithmetic: addition

$(A, <, \leq)$ has addition if there is a function $+ : A \to A \to A$ such that:

$$\text{is-zero}(a) \to c + a = c$$
$$a \text{ is-suc-of } b \to d \text{ is-suc-of } (c + b) \to c + a = d$$
$$a \text{ is-lim-of } f \to b \text{ is-sup-of } (\lambda i.c + f_i) \to c + a = b$$

$(A, <, \leq)$ has unique addition if there is exactly one function with these properties.

Concrete results: Cnf and Brw have unique addition. Ord has addition.

Addition for Cantor Normal Forms

Standard definition:

$$\mathbf{0} + b = b$$
$$a + \mathbf{0} = a$$
$$(\omega\hat{}\ a + c) + (\omega\hat{}\ b + d) \text{ with } \text{<-tri}\ a\ b$$
$$\ldots \mid \text{inl}\ a{<}b = \omega\hat{}\ b + d$$
$$\ldots \mid \text{inr}\ a{\geq}b = \omega\hat{}\ a + (c + \omega\hat{}\ b + d)$$

Addition for Cantor Normal Forms

Standard definition:

$$\mathbf{0} + b = b$$
$$a + \mathbf{0} = a$$
$$(\omega\hat{}\ a + c) + (\omega\hat{}\ b + d) \text{ with } <\text{-tri } a\ b$$
$$\ldots \mid \text{inl } a{<}b = \omega\hat{}\ b + d$$
$$\ldots \mid \text{inr } a{\geq}b = \omega\hat{}\ a + (c + \omega\hat{}\ b + d)$$

Followed by proofs that $+$ preserves isCNF.

Addition for Cantor Normal Forms

Standard definition:

$$\mathbf{0} + b = b$$
$$a + \mathbf{0} = a$$
$$(\omega\hat{}\ a + c) + (\omega\hat{}\ b + d)\ \text{with}\ \text{<-tri}\ a\ b$$
$$\ldots \mid \text{inl}\ a{<}b = \omega\hat{}\ b + d$$
$$\ldots \mid \text{inr}\ a{\geq}b = \omega\hat{}\ a + (c + \omega\hat{}\ b + d)$$

Followed by proofs that $+$ preserves isCNF.

Perhaps less standard: to prove correctness, need to define subtraction.

Abstract arithmetic: multiplication

Assume that $(A, <, \leq)$ has addition.

Abstract arithmetic: multiplication

Assume that $(A, <, \leq)$ has addition.

$(A, <, \leq)$ has multiplication if we have $\cdot : A \to A \to A$ such that:

$$\begin{aligned}
&\text{is-zero}(a) \to c \cdot a = a \\
&a \text{ is-suc-of } b \to c \cdot a = c \cdot b + c \\
&a \text{ is-lim-of } f \to b \text{ is-sup-of } (\lambda i. c \cdot f_i) \to c \cdot a = b
\end{aligned}$$

$(A, <, \leq)$ has unique multiplication if it has unique addition and there is exactly one function with the above properties.

Abstract arithmetic: multiplication

Assume that $(A, <, \leq)$ has addition.

$(A, <, \leq)$ has multiplication if we have $\cdot : A \to A \to A$ such that:

$$
\begin{aligned}
&\text{is-zero}(a) \to c \cdot a = a \\
&a \text{ is-suc-of } b \to c \cdot a = c \cdot b + c \\
&a \text{ is-lim-of } f \to b \text{ is-sup-of } (\lambda i.c \cdot f_i) \to c \cdot a = b
\end{aligned}
$$

$(A, <, \leq)$ has unique multiplication if it has unique addition and there is exactly one function with the above properties.

Concrete results: Cnf and Brw have unique multiplication. Ord has multiplication.

Multiplication for Brouwer trees

Seemingly straightforward definition:

$$
\begin{aligned}
x \cdot \mathsf{zero} &= \mathsf{zero} \\
x \cdot (\mathsf{succ}\, y) &= x \cdot y + x \\
x \cdot (\mathsf{limit}\, f) &= \mathsf{limit}\,(\lambda i.\, x \cdot f_i)
\end{aligned}
$$

Multiplication for Brouwer trees

Seemingly straightforward definition:

$$
\begin{aligned}
x \cdot \mathsf{zero} &= \mathsf{zero} \\
x \cdot (\mathsf{succ}\, y) &= x \cdot y + x \\
x \cdot (\mathsf{limit}\, f) &= \mathsf{limit}\, (\lambda i.\, x \cdot f_i)
\end{aligned}
$$

But! $\lambda i.\, \mathsf{zero} \cdot f_i$ is not increasing even if f is.

Multiplication for Brouwer trees

Seemingly straightforward definition:

$$x \cdot \mathsf{zero} = \mathsf{zero}$$
$$x \cdot (\mathsf{succ}\, y) = x \cdot y + x$$
$$x \cdot (\mathsf{limit}\, f) = \mathsf{limit}\, (\lambda i.\, x \cdot f_i)$$

But! $\lambda i.\, \mathsf{zero} \cdot f_i$ is not increasing even if f is.

Thankfully, we can decide if x is zero or not and act accordingly.

Multiplication for Brouwer trees

Seemingly straightforward definition:

$$
\begin{aligned}
x \cdot \mathsf{zero} &= \mathsf{zero} \\
x \cdot (\mathsf{succ}\, y) &= x \cdot y + x \\
x \cdot (\mathsf{limit}\ f\ \{\textit{incr-f}\})\ &\mathsf{with}\ \mathsf{decZero}\ x \\
\ldots \mid \mathsf{yes}\ \textit{x≡0} &= \mathsf{zero} \\
\ldots \mid \mathsf{no}\ \textit{x≢0} &= \mathsf{limit}\ (\lambda i.\, x \cdot f_i)\ \{\mathsf{x\cdot\text{-}increasing}\ \textit{x≢0}\ \textit{incr-f}\}
\end{aligned}
$$

But! $\lambda i.\, \mathsf{zero} \cdot f_i$ is not increasing even if f is.

Thankfully, we can decide if x is zero or not and act accordingly.

Abstract arithmetic: exponentation

Assume that $(A, <, \leq)$ has addition and multiplication.

Abstract arithmetic: exponentation

Assume that $(A, <, \leq)$ has addition and multiplication.

A has exponentation with base c if there is $\exp(c, -) : A \to A$ such that:

$$\text{is-zero}(b) \to a \text{ is-suc-of } b \to \exp(c, b) = a$$
$$a \text{ is-suc-of } b \to \exp(c, a) = \exp(c, b) \cdot c$$
$$a \text{ is-lim-of } f \to \neg\text{is-zero}(c) \to b \text{ is-sup-of } (\exp(c, f_i)) \to \exp(c, a) = b$$
$$a \text{ is-lim-of } f \to \text{is-zero}(c) \to \exp(c, a) = c$$

A has unique exponentation with base c if it has unique addition and multiplication, and if $\exp(c, -)$ is unique.

Abstract arithmetic: exponentation

Assume that $(A, <, \leq)$ has addition and multiplication.

A has exponentation with base c if there is $\text{exp}(c, -) : A \to A$ such that:

$$\text{is-zero}(b) \to a \text{ is-suc-of } b \to \text{exp}(c, b) = a$$

$$a \text{ is-suc-of } b \to \text{exp}(c, a) = \text{exp}(c, b) \cdot c$$

$$a \text{ is-lim-of } f \to \neg\text{is-zero}(c) \to b \text{ is-sup-of } (\text{exp}(c, f_i)) \to \text{exp}(c, a) = b$$

$$a \text{ is-lim-of } f \to \text{is-zero}(c) \to \text{exp}(c, a) = c$$

A has unique exponentation with base c if it has unique addition and multiplication, and if $\text{exp}(c, -)$ is unique.

Concrete results: Brw and Cnf and have unique exponentation (with base ω).

Connections between the notions

"decidable"	"partially decidable"	"undecidable"
$\boxed{\mathsf{Cnf}}$	$\boxed{\mathsf{Brw}}$	$\boxed{\mathsf{Ord}}$

Connections between the notions

“decidable” | “partially decidable” | “undecidable”

$$\begin{tikzcd} \boxed{\mathsf{Cnf}} \arrow[r, hook, "\mathsf{CtoB}", "(\omega^a + b) \mapsto \omega^{\mathsf{CtoB}(a)} + \mathsf{CtoB}(b)"'] & \boxed{\mathsf{Brw}} & \boxed{\mathsf{Ord}} \end{tikzcd}$$

Connections between the notions

"decidable" "partially decidable" "undecidable"

$$\boxed{\mathsf{Cnf}} \xhookrightarrow{\mathsf{CtoB}} \boxed{\mathsf{Brw}} \qquad \boxed{\mathsf{Ord}}$$

$(\omega^a + b) \mapsto \omega^{\mathsf{CtoB}(a)} + \mathsf{CtoB}(b)$

- injective
- preserves and reflects $<$, $\leq$
- commutes with $+$, $\cdot$, ω^-
- bounded (by ε_0)

Connections between the notions

“decidable” “partially decidable” “undecidable”

$$\begin{tikzcd}[column sep=huge] \boxed{\mathsf{Cnf}} \arrow[r, hook, "\mathsf{CtoB}"', "\mathsf{CtoB}"] & \boxed{\mathsf{Brw}} \arrow[r, hook, "\mathsf{BtoO}"] & \boxed{\mathsf{Ord}} \end{tikzcd}$$

$(\omega^a + b) \mapsto \omega^{\mathsf{CtoB}(a)} + \mathsf{CtoB}(b)$ $\qquad$ $A \mapsto (\Sigma Y : \mathsf{Brw})(Y < A)$

- injective
- preserves and reflects $<$, $\leq$
- commutes with $+$, $\cdot$, ω^{-}
- bounded (by ε_0)

Connections between the notions

"decidable" "partially decidable" "undecidable"

$$\boxed{\mathsf{Cnf}} \xhookrightarrow[(\omega^a + b) \mapsto \omega^{\mathsf{CtoB}(a)} + \mathsf{CtoB}(b)]{\mathsf{CtoB}} \boxed{\mathsf{Brw}} \xhookrightarrow[A \mapsto (\Sigma Y : \mathsf{Brw})(Y < A)]{\mathsf{BtoO}} \boxed{\mathsf{Ord}}$$

- injective
- preserves and reflects $<$, $\leq$
- commutes with $+$, $\cdot$, ω^{-}
- bounded (by ε_0)

- injective
- preserves $<$, $\leq$

Connections between the notions

"decidable" "partially decidable" "undecidable"

$$\begin{tikzcd}[column sep=huge] \boxed{\mathsf{Cnf}} \arrow[r, hook, "\mathsf{CtoB}", "(\omega^a + b) \mapsto \omega^{\mathsf{CtoB}(a)} + \mathsf{CtoB}(b)"'] & \boxed{\mathsf{Brw}} \arrow[r, hook, "\mathsf{BtoO}", "A \mapsto (\Sigma Y : \mathsf{Brw})(Y < A)"'] & \boxed{\mathsf{Ord}} \end{tikzcd}$$

- injective
- preserves and reflects $<$, $\leq$
- commutes with $+$, $\cdot$, ω^{-}
- bounded (by ε_0)

- injective
- preserves $<$, $\leq$
- over-approximates $+$, $\cdot$:
  $\mathsf{BtoO}(x + y) \geq \mathsf{BtoO}(x) + \mathsf{BtoO}(y)$

Connections between the notions

$$\underset{\text{“decidable”}}{\boxed{\mathsf{Cnf}}} \xhookrightarrow[(\omega^a + b) \mapsto \omega^{\mathsf{CtoB}(a)} + \mathsf{CtoB}(b)]{\mathsf{CtoB}} \underset{\text{“partially decidable”}}{\boxed{\mathsf{Brw}}} \xhookrightarrow[A \mapsto (\Sigma Y : \mathsf{Brw})(Y < A)]{\mathsf{BtoO}} \underset{\text{“undecidable”}}{\boxed{\mathsf{Ord}}}$$

CtoB:

- injective
- preserves and reflects $<$, $\leq$
- commutes with $+$, $\cdot$, ω^{-}
- bounded (by ε_0)

BtoO:

- injective
- preserves $<$, $\leq$
- over-approximates $+$, $\cdot$:
  $\mathsf{BtoO}(x + y) \geq \mathsf{BtoO}(x) + \mathsf{BtoO}(y)$
- commutes with limits
  (but not successors)

Connections between the notions

"decidable" "partially decidable" "undecidable"

$$\boxed{\mathsf{Cnf}} \xhookrightarrow{\mathsf{CtoB}} \boxed{\mathsf{Brw}} \xhookrightarrow{\mathsf{BtoO}} \boxed{\mathsf{Ord}}$$

$(\omega^a + b) \mapsto \omega^{\mathsf{CtoB}(a)} + \mathsf{CtoB}(b)$

- injective
- preserves and reflects $<$, $\leq$
- commutes with $+$, $\cdot$, ω^-
- bounded (by ε_0)

$A \mapsto (\Sigma Y : \mathsf{Brw})(Y < A)$

- injective
- preserves $<$, $\leq$
- over-approximates $+$, $\cdot$:
  $\mathsf{BtoO}(x + y) \geq \mathsf{BtoO}(x) + \mathsf{BtoO}(y)$
- commutes with limits
  (but not successors)
- LEM $\Rightarrow$ BtoO is a simulation
- BtoO is a simulation $\Rightarrow$ WLPO

Connections between the notions

"decidable" "partially decidable" "undecidable"

$$\boxed{\mathsf{Cnf}} \xhookrightarrow{\mathsf{CtoB}} \boxed{\mathsf{Brw}} \xhookrightarrow{\mathsf{BtoO}} \boxed{\mathsf{Ord}}$$

$(\omega^a + b) \mapsto \omega^{\mathsf{CtoB}(a)} + \mathsf{CtoB}(b)$

- injective
- preserves and reflects $<$, $\leq$
- commutes with $+$, $\cdot$, ω^{-}
- bounded (by ε_0)

$A \mapsto (\Sigma Y : \mathsf{Brw})(Y < A)$

- injective
- preserves $<$, $\leq$
- over-approximates $+$, $\cdot$:
  $\mathsf{BtoO}(x + y) \geq \mathsf{BtoO}(x) + \mathsf{BtoO}(y)$
- commutes with limits
  (but not successors)
- LEM $\Rightarrow$ BtoO is a simulation
- BtoO is a simulation $\Rightarrow$ WLPO
- bounded (by Brw)

Summary

Constructively, different definitions of ordinals are useful for different purposes.

We have considered three different notions, ranging from “decidable” to “undecidable” in general.

Summary

Constructively, different definitions of ordinals are useful for different purposes.

We have considered three different notions, ranging from "decidable" to "undecidable" in general.

Future work:

- Other notions of ordinals (e.g. based on the Veblen Normal Form, or other types of trees [Jervell 2006])?
- Can we make Brw being "partially decidable" precise using the notion of semi-decidability? [Veltri 2017, Escardó and Knapp 2017]

Summary

Constructively, different definitions of ordinals are useful for different purposes.

We have considered three different notions, ranging from “decidable” to “undecidable” in general.

Future work:

- Other notions of ordinals (e.g. based on the Veblen Normal Form, or other types of trees [Jervell 2006])?
- Can we make Brw being “partially decidable” precise using the notion of semi-decidability? [Veltri 2017, Escardó and Knapp 2017]

More details:

- Connecting Constructive Notions of Ordinals in Homotopy Type Theory, MFCS 2021 (`arxiv:2104.02549`)
- Cubical Agda formalisation: `bitbucket.org/nicolaikraus/constructive-ordinals-in-hott/`

Summary

Constructively, different definitions of ordinals are useful for different purposes.

We have considered "decidable" to
"undecidable" in ger

Thank you!

Future work:

- Other notions Normal Form, or other types of trees [
- Can we make E using the notion of semi-decidabilit 2017]

More details:

- Connecting Co hotopy Type Theory, MFCS 2021 (a
- Cubical Agda formalisation: `bitbucket.org/nicolaikraus/constructive-ordinals-in-hott/`

References

In order of appearance

- Alan Turing. 1949. "Checking a Large Routine". In *Report of a Conference on High Speed Automatic Calculating Machines*. University Mathematical Laboratory, Cambridge, UK, 67–69.
- Gerhard Gentzen. 1936. "Die Widerspruchsfreiheit der reinen Zahlentheorie", Mathematische Annalen, **112**: 493–565.
- Reuben Goodstein. 1944. "On the restricted ordinal theorem", Journal of Symbolic Logic, **9**(2): 33–41.
- Laurie Kirby and Jeff Paris. 1982. "Accessible Independence Results for Peano Arithmetic". Bulletin of the London Mathematical Society. **14**(4): 285–293.
- Fredrik Nordvall Forsberg, Chuangjie Xu, and Neil Ghani. 2020. "Three equivalent ordinal notation systems in cubical Agda". In *the 9th ACM SIGPLAN international conference on Certified Programs and Proofs*, 172–185.
- Martín Escardó. Since 2010. "Compact ordinals, discrete ordinals and their relationships". Available at `https://www.cs.bham.ac.uk/~mhe/TypeTopology/Ordinals.html`.
- Paul Taylor. 1996. "Intuitionistic sets and ordinals". Journal of Symbolic Logic, **61**(3):705–744.
- Herman Ruge Jervell. 2006. "Constructing ordinals". Philosophia Scientiæ. Travaux d'histoire et de philosophie des sciences CS 6: 5–20.
- Niccolò Veltri.2017. "A type-theoretical study of nontermination". PhD thesis, Tallinn University of Technology.
- Martín Escardó, and Cory Knapp. 2017. "Partial elements and recursion via dominances in univalent type theory.". In *the 26th EACSL Annual Conference on Computer Science Logic*. 21:1–21:16.