Noetherianity and Well Quasi-Orders constructively

Gabriele Buriola[1], Peter Schuster[1], Ingo Blechschmidt[2]

[1]University of Verona, Italy
[2]University of Augsburg, Germany

15/09/2023

Ascending chain condition

Classical logic := Excluded Middle (LEM) + Axiom of Choice (AC).

Classical Noetherianity for Rings

- FBP (Finite Basis Property): every ideal is finitely generated;
- ACC: every ascending chain of ideals *stabilizes*

$$I_0 \subseteq I_1 \subseteq I_2 \subseteq \ldots \quad \Rightarrow \quad \exists n : I_n = I_{n+1} = I_{n+2} = \ldots;$$

- Classically: FBP $\leftrightarrow$ ACC.

Problem:

FBP and ACC are not constructively meaningful!

E.g. the 2 element field $\mathbb{F}_2$ is neither constructively FBP nor ACC.

ACC_0^{fg}: every ascending chain of finitely generated ideals *stalls*

$$I_0 \subseteq I_1 \subseteq I_2 \subseteq \ldots \quad \Rightarrow \quad \exists n : I_n = I_{n+1};$$

Notation: RS-Noetherian:= ACC_0^{fg} (d'après Richman and Seidenberg).

Ascending chain condition

Classical logic := Excluded Middle (LEM) + Axiom of Choice (AC).

Classical Noetherianity for Rings

- FBP (Finite Basis Property): every ideal is finitely generated;
- ACC: every ascending chain of ideals *stabilizes*

$$I_0 \subseteq I_1 \subseteq I_2 \subseteq \ldots \quad \Rightarrow \quad \exists n : I_n = I_{n+1} = I_{n+2} = \ldots;$$

- Classically: FBP $\leftrightarrow$ ACC.

Problem:

FBP and ACC are not constructively meaningful!

E.g. the 2 element field $\mathbb{F}_2$ is neither constructively FBP nor ACC.

ACC_0^{fg}: every ascending chain of finitely generated ideals *stalls*

$$I_0 \subseteq I_1 \subseteq I_2 \subseteq \ldots \quad \Rightarrow \quad \exists n : I_n = I_{n+1};$$

Notation: RS-Noetherian:= ACC_0^{fg} (d'après Richman and Seidenberg).

Ascending chain condition

Classical logic := Excluded Middle (LEM) + Axiom of Choice (AC).

Classical Noetherianity for Rings

- FBP (Finite Basis Property): every ideal is finitely generated;
- ACC: every ascending chain of ideals *stabilizes*

$$I_0 \subseteq I_1 \subseteq I_2 \subseteq \ldots \quad \Rightarrow \quad \exists n : I_n = I_{n+1} = I_{n+2} = \ldots;$$

- Classically: FBP $\leftrightarrow$ ACC.

Problem:

FBP and ACC are not constructively meaningful!

E.g. the 2 element field $\mathbb{F}_2$ is neither constructively FBP nor ACC.

ACC_0^{fg}: every ascending chain of finitely generated ideals *stalls*

$$I_0 \subseteq I_1 \subseteq I_2 \subseteq \ldots \quad \Rightarrow \quad \exists n : I_n = I_{n+1};$$

Notation: RS-Noetherian:= ACC_0^{fg} (d'après Richman and Seidenberg).

Ascending chain condition

Classical logic := Excluded Middle (LEM) + Axiom of Choice (AC).

Classical Noetherianity for Rings

- FBP (Finite Basis Property): every ideal is finitely generated;
- ACC: every ascending chain of ideals *stabilizes*

$$I_0 \subseteq I_1 \subseteq I_2 \subseteq \ldots \quad \Rightarrow \quad \exists n : I_n = I_{n+1} = I_{n+2} = \ldots;$$

- Classically: FBP $\leftrightarrow$ ACC.

Problem:

FBP and ACC are not constructively meaningful!

E.g. the 2 element field $\mathbb{F}_2$ is neither constructively FBP nor ACC.

ACC_0^{fg}: every ascending chain of finitely generated ideals *stalls*

$$I_0 \subseteq I_1 \subseteq I_2 \subseteq \ldots \quad \Rightarrow \quad \exists n : I_n = I_{n+1};$$

Notation: RS-Noetherian:= ACC_0^{fg} (d'après Richman and Seidenberg).

Ascending chain condition

Classical logic := Excluded Middle (LEM) + Axiom of Choice (AC).

Classical Noetherianity for Rings

- FBP (Finite Basis Property): every ideal is finitely generated;
- ACC: every ascending chain of ideals *stabilizes*

$$I_0 \subseteq I_1 \subseteq I_2 \subseteq \ldots \quad \Rightarrow \quad \exists n : I_n = I_{n+1} = I_{n+2} = \ldots;$$

- Classically: FBP $\leftrightarrow$ ACC.

Problem:

FBP and ACC are not constructively meaningful!

E.g. the 2 element field $\mathbb{F}_2$ is neither constructively FBP nor ACC.

ACC_0^{fg}: every ascending chain of finitely generated ideals *stalls*

$$I_0 \subseteq I_1 \subseteq I_2 \subseteq \ldots \quad \Rightarrow \quad \exists n : I_n = I_{n+1};$$

Notation: RS-Noetherian:= ACC_0^{fg} (d'après Richman and Seidenberg).

Ascending chain condition

Classical logic := Excluded Middle (LEM) + Axiom of Choice (AC).

Classical Noetherianity for Rings

- FBP (Finite Basis Property): every ideal is finitely generated;
- ACC: every ascending chain of ideals *stabilizes*

$$I_0 \subseteq I_1 \subseteq I_2 \subseteq \ldots \quad \Rightarrow \quad \exists n : I_n = I_{n+1} = I_{n+2} = \ldots;$$

- Classically: FBP $\leftrightarrow$ ACC.

Problem:

FBP and ACC are not constructively meaningful!

E.g. the 2 element field $\mathbb{F}_2$ is neither constructively FBP nor ACC.

ACC_0^{fg}: every ascending chain of finitely generated ideals *stalls*

$$I_0 \subseteq I_1 \subseteq I_2 \subseteq \ldots \quad \Rightarrow \quad \exists n : I_n = I_{n+1};$$

Notation: RS-Noetherian:= ACC_0^{fg} (d'après Richman and Seidenberg).

Ascending chain condition

Classical logic := Excluded Middle (LEM) + Axiom of Choice (AC).

Classical Noetherianity for Rings

- FBP (Finite Basis Property): every ideal is finitely generated;
- ACC: every ascending chain of ideals *stabilizes*

$$I_0 \subseteq I_1 \subseteq I_2 \subseteq \ldots \quad \Rightarrow \quad \exists n : I_n = I_{n+1} = I_{n+2} = \ldots;$$

- Classically: FBP $\leftrightarrow$ ACC.

Problem:

FBP and ACC are not constructively meaningful!

E.g. the 2 element field $\mathbb{F}_2$ is neither constructively FBP nor ACC.

ACC_0^{fg}: every ascending chain of finitely generated ideals *stalls*

$$I_0 \subseteq I_1 \subseteq I_2 \subseteq \ldots \quad \Rightarrow \quad \exists n : I_n = I_{n+1};$$

Notation: RS-Noetherian:= ACC_0^{fg} (d'après Richman and Seidenberg).

Ascending chain condition

Classical logic := Excluded Middle (LEM) + Axiom of Choice (AC).

Classical Noetherianity for Rings

- FBP (Finite Basis Property): every ideal is finitely generated;
- ACC: every ascending chain of ideals *stabilizes*

$$I_0 \subseteq I_1 \subseteq I_2 \subseteq \ldots \quad \Rightarrow \quad \exists n : I_n = I_{n+1} = I_{n+2} = \ldots;$$

- Classically: FBP $\leftrightarrow$ ACC.

Problem:

FBP and ACC are not constructively meaningful!

E.g. the 2 element field $\mathbb{F}_2$ is neither constructively FBP nor ACC.

ACC_0^{fg}: every ascending chain of finitely generated ideals *stalls*

$$I_0 \subseteq I_1 \subseteq I_2 \subseteq \ldots \quad \Rightarrow \quad \exists n : I_n = I_{n+1};$$

Notation: RS-Noetherian:= ACC_0^{fg} (d'après Richman and Seidenberg).

Ascending chain condition

Classical logic := Excluded Middle (LEM) + Axiom of Choice (AC).

Classical Noetherianity for Rings

- FBP (Finite Basis Property): every ideal is finitely generated;
- ACC: every ascending chain of ideals *stabilizes*

$$I_0 \subseteq I_1 \subseteq I_2 \subseteq \ldots \quad \Rightarrow \quad \exists n : I_n = I_{n+1} = I_{n+2} = \ldots;$$

- Classically: FBP $\leftrightarrow$ ACC.

Problem:

FBP and ACC are not constructively meaningful!

E.g. the 2 element field $\mathbb{F}_2$ is neither constructively FBP nor ACC.

ACC_0^{fg}: every ascending chain of finitely generated ideals *stalls*

$$I_0 \subseteq I_1 \subseteq I_2 \subseteq \ldots \quad \Rightarrow \quad \exists n : I_n = I_{n+1};$$

Notation: RS-Noetherian:= ACC_0^{fg} (d'après Richman and Seidenberg).

Related Properties

Let $(E, \leqslant)$ be a partial order with $x < y \equiv x \leqslant y \wedge x \neq y$:

Hereditary conditions

- $H \subseteq E$ is hereditary if $\forall x(\{y \mid y < x\} \subseteq H \Rightarrow x \in H)$;
- E is hereditary well-founded, hwf, if $H \subseteq E$ hereditary $\Rightarrow H = E$;
- E is well ordered if it is hereditary well-founded and linear.

Ascending trees (Richman'03)

An ascending tree in E is a family $(x_i)_{i \in I} \subseteq E$ where

- I is a tree;
- $i < j \Rightarrow x_i \leqslant x_j$.

An ascending tree *stalls* if $\exists\, i < j : x_i = x_j$.

Inductive definition of "P bars σ"

For a predicate P on ascending finite lists on E, we define $P|\sigma$:

- if $P(\sigma)$ then $P|\sigma$;
- if $P|\sigma x$ for all $x \geqslant \sigma$, then $P|\sigma$.

Related Properties

Let $(E, \leqslant)$ be a partial order with $x < y \equiv x \leqslant y \wedge x \neq y$:

Hereditary conditions

- $H \subseteq E$ is hereditary if $\forall x(\{y \mid y < x\} \subseteq H \Rightarrow x \in H)$;
- E is hereditary well-founded, hwf, if $H \subseteq E$ hereditary $\Rightarrow H = E$;
- E is well ordered if it is hereditary well-founded and linear.

Ascending trees (Richman'03)

An ascending tree in E is a family $(x_i)_{i \in I} \subseteq E$ where

- I is a tree;
- $i < j \Rightarrow x_i \leqslant x_j$.

An ascending tree *stalls* if $\exists\, i < j : x_i = x_j$.

Inductive definition of "P bars σ"

For a predicate P on ascending finite lists on E, we define $P|\sigma$:

- if $P(\sigma)$ then $P|\sigma$;
- if $P|\sigma x$ for all $x \geqslant \sigma$, then $P|\sigma$.

Related Properties

Let $(E, \leqslant)$ be a partial order with $x < y \equiv x \leqslant y \wedge x \neq y$:

Hereditary conditions

- $H \subseteq E$ is hereditary if $\forall x(\{y \mid y < x\} \subseteq H \Rightarrow x \in H)$;
- E is hereditary well-founded, hwf, if $H \subseteq E$ hereditary $\Rightarrow H = E$;
- E is well ordered if it is hereditary well-founded and linear.

Ascending trees (Richman'03)

An ascending tree in E is a family $(x_i)_{i \in I} \subseteq E$ where

- I is a tree;
- $i < j \Rightarrow x_i \leqslant x_j$.

An ascending tree *stalls* if $\exists\, i < j : x_i = x_j$.

Inductive definition of "P bars σ"

For a predicate P on ascending finite lists on E, we define $P|\sigma$:

- if $P(\sigma)$ then $P|\sigma$;
- if $P|\sigma x$ for all $x \geqslant \sigma$, then $P|\sigma$.

Related Properties

Let $(E, \leqslant)$ be a partial order with $x < y \equiv x \leqslant y \wedge x \neq y$:

Hereditary conditions

- $H \subseteq E$ is hereditary if $\forall x(\{y \mid y < x\} \subseteq H \Rightarrow x \in H)$;
- E is hereditary well-founded, hwf, if $H \subseteq E$ hereditary $\Rightarrow H = E$;
- E is well ordered if it is hereditary well-founded and linear.

Ascending trees (Richman'03)

An ascending tree in E is a family $(x_i)_{i \in I} \subseteq E$ where

- I is a tree;
- $i < j \Rightarrow x_i \leqslant x_j$.

An ascending tree *stalls* if $\exists\, i < j : x_i = x_j$.

Inductive definition of "P bars σ"

For a predicate P on ascending finite lists on E, we define $P|\sigma$:

- if $P(\sigma)$ then $P|\sigma$;
- if $P|\sigma x$ for all $x \geqslant \sigma$, then $P|\sigma$.

Related Properties

Let $(E, \leqslant)$ be a partial order with $x < y \equiv x \leqslant y \wedge x \neq y$:

Hereditary conditions

- $H \subseteq E$ is hereditary if $\forall x(\{y \mid y < x\} \subseteq H \Rightarrow x \in H)$;
- E is hereditary well-founded, hwf, if $H \subseteq E$ hereditary $\Rightarrow H = E$;
- E is well ordered if it is hereditary well-founded and linear.

Ascending trees (Richman'03)

An ascending tree in E is a family $(x_i)_{i \in I} \subseteq E$ where

- I is a tree;
- $i < j \Rightarrow x_i \leqslant x_j$.

An ascending tree *stalls* if $\exists\, i < j : x_i = x_j$.

Inductive definition of "P bars σ"

For a predicate P on ascending finite lists on E, we define $P|\sigma$:

- if $P(\sigma)$ then $P|\sigma$;
- if $P|\sigma x$ for all $x \geqslant \sigma$, then $P|\sigma$.

Related Properties

Let $(E, \leqslant)$ be a partial order with $x < y \equiv x \leqslant y \wedge x \neq y$:

Hereditary conditions

- $H \subseteq E$ is hereditary if $\forall x(\{y \mid y < x\} \subseteq H \Rightarrow x \in H)$;
- E is hereditary well-founded, hwf, if $H \subseteq E$ hereditary $\Rightarrow H = E$;
- E is well ordered if it is hereditary well-founded and linear.

Ascending trees (Richman'03)

An ascending tree in E is a family $(x_i)_{i \in I} \subseteq E$ where

- I is a tree;
- $i < j \Rightarrow x_i \leqslant x_j$.

An ascending tree *stalls* if $\exists\, i < j : x_i = x_j$.

Inductive definition of "P bars σ"

For a predicate P on ascending finite lists on E, we define $P|\sigma$:

- if $P(\sigma)$ then $P|\sigma$;
- if $P|\sigma x$ for all $x \geqslant \sigma$, then $P|\sigma$.

Related Properties

Let $(E, \leqslant)$ be a partial order with $x < y \equiv x \leqslant y \wedge x \neq y$:

Hereditary conditions

- $H \subseteq E$ is hereditary if $\forall x(\{y \mid y < x\} \subseteq H \Rightarrow x \in H)$;
- E is hereditary well-founded, hwf, if $H \subseteq E$ hereditary $\Rightarrow H = E$;
- E is well ordered if it is hereditary well-founded and linear.

Ascending trees (Richman'03)

An ascending tree in E is a family $(x_i)_{i \in I} \subseteq E$ where

- I is a tree;
- $i < j \Rightarrow x_i \leqslant x_j$.

An ascending tree *stalls* if $\exists\, i < j : x_i = x_j$.

Inductive definition of "P bars σ"

For a predicate P on ascending finite lists on E, we define $P|\sigma$:

- if $P(\sigma)$ then $P|\sigma$;
- if $P|\sigma x$ for all $x \geqslant \sigma$, then $P|\sigma$.

Related Properties

Let $(E, \leqslant)$ be a partial order with $x < y \equiv x \leqslant y \wedge x \neq y$:

Hereditary conditions

- $H \subseteq E$ is hereditary if $\forall x(\{y \mid y < x\} \subseteq H \Rightarrow x \in H)$;
- E is hereditary well-founded, hwf, if $H \subseteq E$ hereditary $\Rightarrow H = E$;
- E is well ordered if it is hereditary well-founded and linear.

Ascending trees (Richman'03)

An ascending tree in E is a family $(x_i)_{i \in I} \subseteq E$ where

- I is a tree;
- $i < j \Rightarrow x_i \leqslant x_j$.

An ascending tree *stalls* if $\exists\, i < j : x_i = x_j$.

Inductive definition of "P bars σ"

For a predicate P on ascending finite lists on E, we define $P|\sigma$:

- if $P(\sigma)$ then $P|\sigma$;
- if $P|\sigma x$ for all $x \geqslant \sigma$, then $P|\sigma$.

Intuitionistic Noetherian properties and their relations

A partial order $(E, \leqslant)$ is

- RS-Noetherian if for $e_1 \leqslant e_2 \leqslant \dots$ there is n with $e_n = e_{n+1}$;
- ML-Noetherian if the reverse order $(E, \geqslant)$ is hwf;
- strongly Noetherian if there is a well-order W and a strictly descending map $\varphi : E \to W$, i.e. $e < f \Rightarrow \varphi(e) > \varphi(f)$;
- tree Noetherian if every ascending tree in E stalls;
- inductively Noetherian if $\mathrm{Stall} \,|\, []$, where $\mathrm{Stall}(\sigma)$="σ is an ascending finite list with repeated terms".

Def: given a ring R, $\mathcal{I}_f(R)$ is the set of finitely generated ideals of R.
Def: a ring R is * Noetherian if $(\mathcal{I}_f(R), \subseteq)$ is * Noetherian.

Constructive implications for a decidable poset $(E, \leqslant)$

$$\begin{array}{ccccccc} \text{strong} & \longrightarrow & \text{ind} & \longrightarrow & \text{tree} & \longrightarrow & \text{RS} \\ & & \updownarrow & & & & \\ & & \text{ML} & & & & \end{array}$$

Intuitionistic Noetherian properties and their relations

A partial order $(E, \leqslant)$ is

- RS-Noetherian if for $e_1 \leqslant e_2 \leqslant \dots$ there is n with $e_n = e_{n+1}$;
- ML-Noetherian if the reverse order $(E, \geqslant)$ is hwf;
- strongly Noetherian if there is a well-order W and a strictly descending map $\varphi : E \to W$, i.e. $e < f \Rightarrow \varphi(e) > \varphi(f)$;
- tree Noetherian if every ascending tree in E stalls;
- inductively Noetherian if Stall $\mid []$, where Stall(σ)="σ is an ascending finite list with repeated terms".

Def: given a ring R, $\mathcal{I}_f(R)$ is the set of finitely generated ideals of R.
Def: a ring R is * Noetherian if $(\mathcal{I}_f(R), \subseteq)$ is * Noetherian.

Constructive implications for a decidable poset $(E, \leqslant)$

Intuitionistic Noetherian properties and their relations

A partial order $(E, \leqslant)$ is

- RS-Noetherian if for $e_1 \leqslant e_2 \leqslant \dots$ there is n with $e_n = e_{n+1}$;
- ML-Noetherian if the reverse order $(E, \geqslant)$ is hwf;
- strongly Noetherian if there is a well-order W and a strictly descending map $\varphi : E \to W$, i.e. $e < f \Rightarrow \varphi(e) > \varphi(f)$;
- tree Noetherian if every ascending tree in E stalls;
- inductively Noetherian if $\mathrm{Stall} \,|\, []$, where $\mathrm{Stall}(\sigma)$="σ is an ascending finite list with repeated terms".

Def: given a ring R, $\mathcal{I}_f(R)$ is the set of finitely generated ideals of R.
Def: a ring R is * Noetherian if $(\mathcal{I}_f(R), \subseteq)$ is * Noetherian.

Constructive implications for a decidable poset $(E, \leqslant)$

$$\begin{array}{ccccccc} \text{strong} & \longrightarrow & \text{ind} & \longrightarrow & \text{tree} & \longrightarrow & \text{RS} \\ & & \updownarrow & & & & \\ & & \text{ML} & & & & \end{array}$$

Intuitionistic Noetherian properties and their relations

A partial order $(E, \leqslant)$ is

- RS-Noetherian if for $e_1 \leqslant e_2 \leqslant \dots$ there is n with $e_n = e_{n+1}$;
- ML-Noetherian if the reverse order $(E, \geqslant)$ is hwf;
- strongly Noetherian if there is a well-order W and a strictly descending map $\varphi\colon E \to W$, i.e. $e < f \Rightarrow \varphi(e) > \varphi(f)$;
- tree Noetherian if every ascending tree in E stalls;
- inductively Noetherian if $\mathrm{Stall} \mid []$, where $\mathrm{Stall}(\sigma)$="σ is an ascending finite list with repeated terms".

Def: given a ring R, $\mathcal{I}_f(R)$ is the set of finitely generated ideals of R.
Def: a ring R is * Noetherian if $(\mathcal{I}_f(R), \subseteq)$ is * Noetherian.

Constructive implications for a decidable poset $(E, \leqslant)$

$$\begin{array}{ccccccc} \text{strong} & \longrightarrow & \text{ind} & \longrightarrow & \text{tree} & \longrightarrow & \text{RS} \\ & & \updownarrow & & & & \\ & & \text{ML} & & & & \end{array}$$

Intuitionistic Noetherian properties and their relations

A partial order $(E, \leqslant)$ is

- RS-Noetherian if for $e_1 \leqslant e_2 \leqslant \dots$ there is n with $e_n = e_{n+1}$;
- ML-Noetherian if the reverse order $(E, \geqslant)$ is hwf;
- strongly Noetherian if there is a well-order W and a strictly descending map $\varphi\colon E \to W$, i.e. $e < f \Rightarrow \varphi(e) > \varphi(f)$;
- tree Noetherian if every ascending tree in E stalls;
- inductively Noetherian if $\mathrm{Stall} \mid []$, where $\mathrm{Stall}(\sigma)$="σ is an ascending finite list with repeated terms".

Def: given a ring R, $\mathcal{I}_f(R)$ is the set of finitely generated ideals of R.
Def: a ring R is * Noetherian if $(\mathcal{I}_f(R), \subseteq)$ is * Noetherian.

Constructive implications for a decidable poset $(E, \leqslant)$

$$\begin{array}{ccccccc} \text{strong} & \longrightarrow & \text{ind} & \longrightarrow & \text{tree} & \longrightarrow & \text{RS} \\ & & \updownarrow & & & & \\ & & \text{ML} & & & & \end{array}$$

Intuitionistic Noetherian properties and their relations

A partial order $(E, \leqslant)$ is

- RS-Noetherian if for $e_1 \leqslant e_2 \leqslant \dots$ there is n with $e_n = e_{n+1}$;
- ML-Noetherian if the reverse order $(E, \geqslant)$ is hwf;
- strongly Noetherian if there is a well-order W and a strictly descending map $\varphi\colon E \to W$, i.e. $e < f \Rightarrow \varphi(e) > \varphi(f)$;
- tree Noetherian if every ascending tree in E stalls;
- inductively Noetherian if $\mathrm{Stall} \mid []$, where $\mathrm{Stall}(\sigma)$="σ is an ascending finite list with repeated terms".

Def: given a ring R, $\mathcal{I}_f(R)$ is the set of finitely generated ideals of R.
Def: a ring R is * Noetherian if $(\mathcal{I}_f(R), \subseteq)$ is * Noetherian.

Constructive implications for a decidable poset $(E, \leqslant)$

Intuitionistic Noetherian properties and their relations

A partial order $(E, \leqslant)$ is

- RS-Noetherian if for $e_1 \leqslant e_2 \leqslant \dots$ there is n with $e_n = e_{n+1}$;
- ML-Noetherian if the reverse order $(E, \geqslant)$ is hwf;
- strongly Noetherian if there is a well-order W and a strictly descending map $\varphi\colon E \to W$, i.e. $e < f \Rightarrow \varphi(e) > \varphi(f)$;
- tree Noetherian if every ascending tree in E stalls;
- inductively Noetherian if Stall $\mid []$, where Stall(σ)="σ is an ascending finite list with repeated terms".

Def: given a ring R, $\mathcal{I}_f(R)$ is the set of finitely generated ideals of R.
Def: a ring R is * Noetherian if $(\mathcal{I}_f(R), \subseteq)$ is * Noetherian.

Constructive implications for a decidable poset $(E, \leqslant)$

strong $\longrightarrow$ ind $\longrightarrow$ tree $\longrightarrow$ RS

ind $\updownarrow$ ML

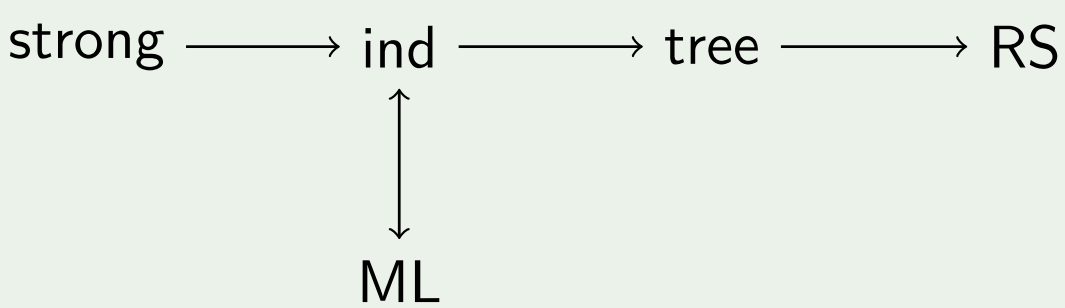
Intuitionistic Noetherian properties and their relations

A partial order $(E, \leqslant)$ is

- RS-Noetherian if for $e_1 \leqslant e_2 \leqslant \dots$ there is n with $e_n = e_{n+1}$;
- ML-Noetherian if the reverse order $(E, \geqslant)$ is hwf;
- strongly Noetherian if there is a well-order W and a strictly descending map $\varphi\colon E \to W$, i.e. $e < f \Rightarrow \varphi(e) > \varphi(f)$;
- tree Noetherian if every ascending tree in E stalls;
- inductively Noetherian if Stall $\mid []$, where Stall(σ)="σ is an ascending finite list with repeated terms".

Def: given a ring R, $\mathcal{I}_f(R)$ is the set of finitely generated ideals of R.
Def: a ring R is * Noetherian if $(\mathcal{I}_f(R), \subseteq)$ is * Noetherian.

Constructive implications for a decidable poset $(E, \leqslant)$



Basic definitions for quasi-orders

Quasi-order

A qo $(Q, \leqslant)$ is a set Q with a transitive and reflexive relation $\leqslant$.

Notation

- $p < q \equiv p \leqslant q \wedge q \not\leqslant p$;
- $p \perp q \equiv p \not\leqslant q \wedge q \not\leqslant p$;
- $p \sim q \equiv p \leqslant q \wedge q \leqslant p$.

Auxiliary definitions

For every qo $(Q, \leqslant)$:

- the closure of $B \subseteq Q$ is $\uparrow B := \{q \in Q \mid \exists b \in B\ b \leqslant q\}$;
- B is closed if $B = \uparrow B$ and finitely generated if $B = \uparrow\{b_1, \ldots, b_n\}$;
- a sequence $(q_k)_k$ in Q is a total function from $\mathbb{N}$ to Q;
- an antichain is a sequence $(q_k)_k$ such that $q_i \perp q_j$ if $i \neq j$;
- an extension of $(Q, \leqslant)$ is a qo $\preccurlyeq$ on Q extending $\leqslant$, i.e., $p \leqslant q \Rightarrow p \preccurlyeq q$ and $p \preccurlyeq q \wedge q \preccurlyeq p \Rightarrow p \sim q$.

Basic definitions for quasi-orders

Quasi-order

A qo $(Q, \leqslant)$ is a set Q with a transitive and reflexive relation $\leqslant$.

Notation

- $p < q \equiv p \leqslant q \wedge q \not\leqslant p$;
- $p \perp q \equiv p \not\leqslant q \wedge q \not\leqslant p$;
- $p \sim q \equiv p \leqslant q \wedge q \leqslant p$.

Auxiliary definitions

For every qo $(Q, \leqslant)$:

- the closure of $B \subseteq Q$ is $\uparrow B := \{q \in Q \mid \exists b \in B\ b \leqslant q\}$;
- B is closed if $B = \uparrow B$ and finitely generated if $B = \uparrow\{b_1, \ldots, b_n\}$;
- a sequence $(q_k)_k$ in Q is a total function from $\mathbb{N}$ to Q;
- an antichain is a sequence $(q_k)_k$ such that $q_i \perp q_j$ if $i \neq j$;
- an extension of $(Q, \leqslant)$ is a qo $\preccurlyeq$ on Q extending $\leqslant$, i.e., $p \leqslant q \Rightarrow p \preccurlyeq q$ and $p \preccurlyeq q \wedge q \preccurlyeq p \Rightarrow p \sim q$.

Basic definitions for quasi-orders

Quasi-order

A qo $(Q, \leqslant)$ is a set Q with a transitive and reflexive relation $\leqslant$.

Notation

- $p < q \equiv p \leqslant q \wedge q \not\leqslant p$;
- $p \perp q \equiv p \not\leqslant q \wedge q \not\leqslant p$;
- $p \sim q \equiv p \leqslant q \wedge q \leqslant p$.

Auxiliary definitions

For every qo $(Q, \leqslant)$:

- the closure of $B \subseteq Q$ is $\uparrow B := \{q \in Q \mid \exists b \in B \; b \leqslant q\}$;
- B is closed if $B = \uparrow B$ and finitely generated if $B = \uparrow\{b_1, \ldots, b_n\}$;
- a sequence $(q_k)_k$ in Q is a total function from $\mathbb{N}$ to Q;
- an antichain is a sequence $(q_k)_k$ such that $q_i \perp q_j$ if $i \neq j$;
- an extension of $(Q, \leqslant)$ is a qo $\preccurlyeq$ on Q extending $\leqslant$, i.e., $p \leqslant q \Rightarrow p \preccurlyeq q$ and $p \preccurlyeq q \wedge q \preccurlyeq p \Rightarrow p \sim q$.

Basic definitions for quasi-orders

Quasi-order

A qo $(Q, \leqslant)$ is a set Q with a transitive and reflexive relation $\leqslant$.

Notation

- $p < q \equiv p \leqslant q \wedge q \not\leqslant p$;
- $p \perp q \equiv p \not\leqslant q \wedge q \not\leqslant p$;
- $p \sim q \equiv p \leqslant q \wedge q \leqslant p$.

Auxiliary definitions

For every qo $(Q, \leqslant)$:

- the closure of $B \subseteq Q$ is $\uparrow B := \{q \in Q \mid \exists b \in B\ b \leqslant q\}$;
- B is closed if $B = \uparrow B$ and finitely generated if $B = \uparrow\{b_1, \ldots, b_n\}$;
- a sequence $(q_k)_k$ in Q is a total function from $\mathbb{N}$ to Q;
- an antichain is a sequence $(q_k)_k$ such that $q_i \perp q_j$ if $i \neq j$;
- an extension of $(Q, \leqslant)$ is a qo $\preccurlyeq$ on Q extending $\leqslant$, i.e., $p \leqslant q \Rightarrow p \preccurlyeq q$ and $p \preccurlyeq q \wedge q \preccurlyeq p \Rightarrow p \sim q$.

Basic definitions for quasi-orders

Quasi-order

A qo $(Q, \leqslant)$ is a set Q with a transitive and reflexive relation $\leqslant$.

Notation

- $p < q \equiv p \leqslant q \wedge q \not\leqslant p$;
- $p \perp q \equiv p \not\leqslant q \wedge q \not\leqslant p$;
- $p \sim q \equiv p \leqslant q \wedge q \leqslant p$.

Auxiliary definitions

For every qo $(Q, \leqslant)$:

- the closure of $B \subseteq Q$ is $\uparrow B := \{q \in Q \mid \exists b \in B\ b \leqslant q\}$;
- B is closed if $B = \uparrow B$ and finitely generated if $B = \uparrow\{b_1, \ldots, b_n\}$;
- a sequence $(q_k)_k$ in Q is a total function from $\mathbb{N}$ to Q;
- an antichain is a sequence $(q_k)_k$ such that $q_i \perp q_j$ if $i \neq j$;
- an extension of $(Q, \leqslant)$ is a qo $\preccurlyeq$ on Q extending $\leqslant$, i.e., $p \leqslant q \Rightarrow p \preccurlyeq q$ and $p \preccurlyeq q \wedge q \preccurlyeq p \Rightarrow p \sim q$.

Basic definitions for quasi-orders

Quasi-order

A qo $(Q, \leqslant)$ is a set Q with a transitive and reflexive relation $\leqslant$.

Notation

- $p < q \equiv p \leqslant q \wedge q \not\leqslant p$;
- $p \perp q \equiv p \not\leqslant q \wedge q \not\leqslant p$;
- $p \sim q \equiv p \leqslant q \wedge q \leqslant p$.

Auxiliary definitions

For every qo $(Q, \leqslant)$:

- the closure of $B \subseteq Q$ is $\uparrow B := \{q \in Q \mid \exists b \in B \ b \leqslant q\}$;
- B is closed if $B = \uparrow B$ and finitely generated if $B = \uparrow\{b_1, \ldots, b_n\}$;
- a sequence $(q_k)_k$ in Q is a total function from $\mathbb{N}$ to Q;
- an antichain is a sequence $(q_k)_k$ such that $q_i \perp q_j$ if $i \neq j$;
- an extension of $(Q, \leqslant)$ is a qo $\preccurlyeq$ on Q extending $\leqslant$, i.e., $p \leqslant q \Rightarrow p \preccurlyeq q$ and $p \preccurlyeq q \wedge q \preccurlyeq p \Rightarrow p \sim q$.

Well quasi-orders definitions

A qo $(Q, \leqslant)$ is

- well-founded if for $q_1 \geqslant q_2 \geqslant \dots$ there is n such that $q_n = q_{n+1}$;
- wqo if for any sequence $(q_k)_k$ in Q there exist $i < j$ with $q_i \leqslant q_j$;
- wqo(set) if every sequence $(q_k)_k$ in Q has an infinite ascending subsequence: there are $k_0 < k_1 < \dots$ such that $q_{k_0} \leqslant q_{k_1} \leqslant \dots$;
- wqo(anti) if it is well-founded and every antichain is finite;
- wqo(ext) if every linear extension of Q is well-founded;
- wqo(fbp) if every closed subset is finitely generated;
- wqo(acc) if the set of closed subsets is Noetherian;
- wqo(*) if the set of finitely generated closed subsets is *Noetherian.

Remark: all the wqo definitions are classically equivalent.

Well quasi-orders definitions

A qo $(Q, \leqslant)$ is

- well-founded if for $q_1 \geqslant q_2 \geqslant \dots$ there is n such that $q_n = q_{n+1}$;
- wqo if for any sequence $(q_k)_k$ in Q there exist $i < j$ with $q_i \leqslant q_j$;
- wqo(set) if every sequence $(q_k)_k$ in Q has an infinite ascending subsequence: there are $k_0 < k_1 < \dots$ such that $q_{k_0} \leqslant q_{k_1} \leqslant \dots$;
- wqo(anti) if it is well-founded and every antichain is finite;
- wqo(ext) if every linear extension of Q is well-founded;
- wqo(fbp) if every closed subset is finitely generated;
- wqo(acc) if the set of closed subsets is Noetherian;
- wqo(*) if the set of finitely generated closed subsets is *Noetherian.

Remark: all the wqo definitions are classically equivalent.

Well quasi-orders definitions

A qo $(Q, \leqslant)$ is

- well-founded if for $q_1 \geqslant q_2 \geqslant \dots$ there is n such that $q_n = q_{n+1}$;
- wqo if for any sequence $(q_k)_k$ in Q there exist $i < j$ with $q_i \leqslant q_j$;
- wqo(set) if every sequence $(q_k)_k$ in Q has an infinite ascending subsequence: there are $k_0 < k_1 < \dots$ such that $q_{k_0} \leqslant q_{k_1} \leqslant \dots$;
- wqo(anti) if it is well-founded and every antichain is finite;
- wqo(ext) if every linear extension of Q is well-founded;
- wqo(fbp) if every closed subset is finitely generated;
- wqo(acc) if the set of closed subsets is Noetherian;
- wqo(*) if the set of finitely generated closed subsets is *Noetherian.

Remark: all the wqo definitions are classically equivalent.

Well quasi-orders definitions

A qo $(Q, \leqslant)$ is

- well-founded if for $q_1 \geqslant q_2 \geqslant \dots$ there is n such that $q_n = q_{n+1}$;
- wqo if for any sequence $(q_k)_k$ in Q there exist $i < j$ with $q_i \leqslant q_j$;
- wqo(set) if every sequence $(q_k)_k$ in Q has an infinite ascending subsequence: there are $k_0 < k_1 < \dots$ such that $q_{k_0} \leqslant q_{k_1} \leqslant \dots$;
- wqo(anti) if it is well-founded and every antichain is finite;
- wqo(ext) if every linear extension of Q is well-founded;
- wqo(fbp) if every closed subset is finitely generated;
- wqo(acc) if the set of closed subsets is Noetherian;
- wqo(*) if the set of finitely generated closed subsets is *Noetherian.

Remark: all the wqo definitions are classically equivalent.

Constructive relations between wqo definitions

Theorem

The conditions wqo(set), wqo(fbp) and wqo(acc) are not constructively meaningful.

Implications between constructive wqo definitions

Constructive relations between wqo definitions

Theorem

The conditions wqo(set), wqo(fbp) and wqo(acc) are not constructively meaningful.

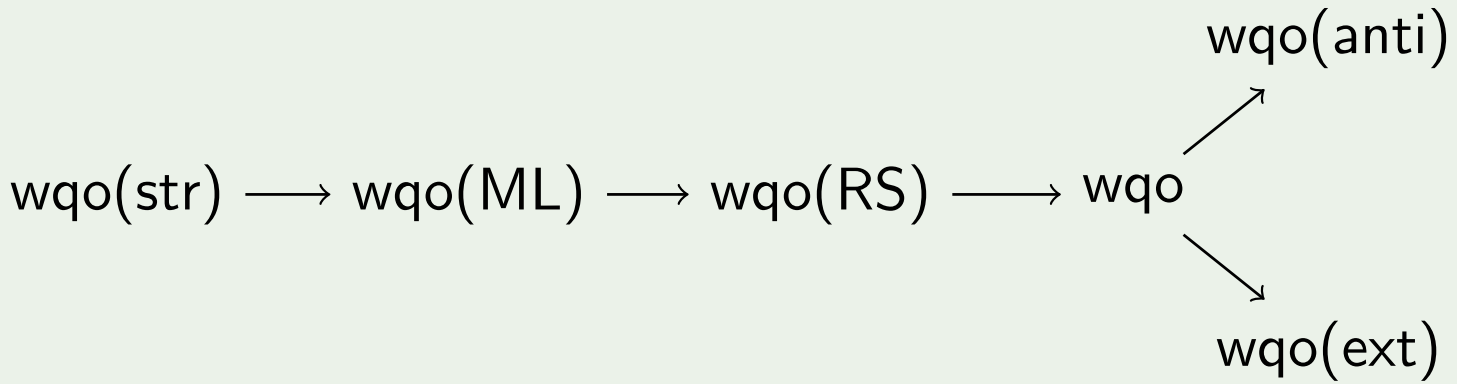
Implications between constructive wqo definitions

Constructive relations between wqo definitions

Theorem

The conditions wqo(set), wqo(fbp) and wqo(acc) are not constructively meaningful.

Implications between constructive wqo definitions



Future work

Reverse implications

Which of the following implications can be reversed?

- strongly Noetherian $\Rightarrow$ ML-Noetherian;
- wqo(RS) $\Rightarrow$ wqo;
- wqo $\Rightarrow$ wqo(anti);
- ...

For now, Tree Noetherian $\not\Rightarrow$ ML-Noetherian by A. Blass.

Closure properties

If P and Q have property $\mathcal{P} \in \{wqo, wqo(anti), \dots\}$, does

- $P \dot{\cup} Q$ constructively have property $\mathcal{P}$?
- $P \times Q$ constructively have property $\mathcal{P}$?

Future work

Reverse implications

Which of the following implications can be reversed?

- strongly Noetherian $\Rightarrow$ ML-Noetherian;
- wqo(RS) $\Rightarrow$ wqo;
- wqo $\Rightarrow$ wqo(anti);
- ...

For now, Tree Noetherian $\not\Rightarrow$ ML-Noetherian by A. Blass.

Closure properties

If P and Q have property $\mathcal{P} \in \{wqo, wqo(anti), \dots\}$, does

- $P \dot{\cup} Q$ constructively have property $\mathcal{P}$?
- $P \times Q$ constructively have property $\mathcal{P}$?

Future work

Reverse implications

Which of the following implications can be reversed?

- strongly Noetherian $\Rightarrow$ ML-Noetherian;
- wqo(RS) $\Rightarrow$ wqo;
- wqo $\Rightarrow$ wqo(anti);
- ...

For now, Tree Noetherian $\not\Rightarrow$ ML-Noetherian by A. Blass.

Closure properties

If P and Q have property $\mathcal{P} \in \{wqo, wqo(anti), \dots\}$, does

- $P \dot{\cup} Q$ constructively have property $\mathcal{P}$?
- $P \times Q$ constructively have property $\mathcal{P}$?

Future work

Reverse implications

Which of the following implications can be reversed?

- strongly Noetherian $\Rightarrow$ ML-Noetherian;
- wqo(RS) $\Rightarrow$ wqo;
- wqo $\Rightarrow$ wqo(anti);
- ...

For now, Tree Noetherian $\not\Rightarrow$ ML-Noetherian by A. Blass.

Closure properties

If P and Q have property $\mathcal{P} \in \{wqo, wqo(anti), \dots\}$, does

- $P \dot{\cup} Q$ constructively have property $\mathcal{P}$?
- $P \times Q$ constructively have property $\mathcal{P}$?

Future work

Reverse implications

Which of the following implications can be reversed?

- strongly Noetherian $\Rightarrow$ ML-Noetherian;
- wqo(RS) $\Rightarrow$ wqo;
- wqo $\Rightarrow$ wqo(anti);
- ...

For now, Tree Noetherian $\not\Rightarrow$ ML-Noetherian by A. Blass.

Closure properties

If P and Q have property $\mathcal{P} \in \{wqo, wqo(anti), \dots\}$, does

- $P \dot{\cup} Q$ constructively have property $\mathcal{P}$?
- $P \times Q$ constructively have property $\mathcal{P}$?

Future work

Reverse implications

Which of the following implications can be reversed?

- strongly Noetherian $\Rightarrow$ ML-Noetherian;
- wqo(RS) $\Rightarrow$ wqo;
- wqo $\Rightarrow$ wqo(anti);
- ...

For now, Tree Noetherian $\not\Rightarrow$ ML-Noetherian by A. Blass.

Closure properties

If P and Q have property $\mathcal{P} \in \{wqo, wqo(anti), \dots\}$, does

- $P \dot{\cup} Q$ constructively have property $\mathcal{P}$?
- $P \times Q$ constructively have property $\mathcal{P}$?

Thank you!

References:


Higman, G.: *Ordering by divisibility in abstract algebras*. Proc. London Math. Soc. 3, 2:326-336, (1952).


Richman, F.: *Constructive Aspects of Noetherian Rings*. Proceedings of the American Mathematical Society 44(2), 436-441, (1974).


Seidenberg, A.: *What is Noetherian?* Rendiconti del Seminario matematico e fisico di Milano 44(1), 55-61 (1974).


Jacobsson C., Löfwall C.: *Standard bases for general coefficient rings and a new constructive proof of Hilbert's basis theorem*. J. Symbolic Comput. 12(3), 337-371 (1991)


Richman, F.: *The ascending tree condition: constructive algebra without countable choice*. Commun. Algebra 31(4), 1993-2002 (2003)


Perdry, H.: *Strongly Noetherian rings and constructive ideal theory*. Journal of symbolic computation 37(4), 511-535 (2004).


Veldman, W.: *An Intuitionistic Proof of Kruskal's Theorem*. Archive for Mathematical Logic 43(2), 215-264 (2004).


Cholak P., Marcone A., and Solomon R.: *Reverse mathematics and the equivalence of definitions for well and better quasi-orders*. The Journal of symbolic logic, 66(1):683-55, (2004).

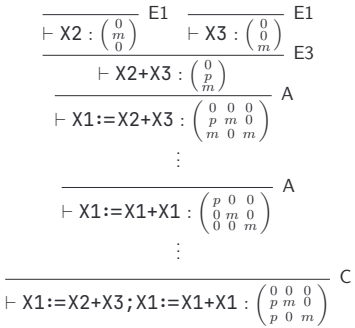
```
void main(int X1, int X2, int X3){
    // initial values  ↑

    // various commands involving
    // variables X1, X2, X3

    // X1', X2', X3' (final values)
}
```

$\forall n$, is $\mathsf{X}_n \rightsquigarrow \mathsf{X}'_n$ polynomially bounded in inputs?

```
void main(int X1, int X2, int X3){
    X1 = X2 + X3;
    X1 = X1 + X1;
}
```



```
void main(int X1; int X2){
    X1 = 1;
    loop X2 {
        X1 = X1 + X1;
    }
}
```

$$\dfrac{\dfrac{}{\vdash \texttt{X1:=1} : \binom{m}{0}}\ \text{E1}}{\vdots}$$
$$\dfrac{}{\vdash \texttt{X1:=X1+X1} : \begin{pmatrix} p & 0 \\ 0 & m \end{pmatrix}}\ \text{A}$$
$$\vdots$$
$$\times$$

Implicit
Computational
Complexity

$\longrightarrow$

theory and beyond!

Static program analysis

 `statycc/pymwp`

Program transformation

Formally verified complexity analysis 

Strong negation in TCF

Nils Köpp

September 15, 2023

LMU München

Strong negation

As opposed to weak (uniform) negation $\neg A := A \to \bot$, define strong negation $\cdot^{\mathbf{N}}$ as a mapping **Form** $\to$ **Form**, defined by recursion on formulas.

$$(A \wedge B)^{\mathbf{N}} := A^{\mathbf{N}} \vee A^{\mathbf{N}}$$
$$(A \vee B)^{\mathbf{N}} := A^{\mathbf{N}} \wedge B^{\mathbf{N}}$$
$$(\forall_x A)^{\mathbf{N}} := \exists_x A^{\mathbf{N}}$$
$$(\exists_x A)^{\mathbf{N}} := \forall_x A^{\mathbf{N}}$$
$$(A \to B)^{\mathbf{N}} := A \wedge B^{\mathbf{N}}$$
$$(R\vec{t})^{\mathbf{N}} :=?$$

Theory of Computable Functionals (TCF)

- independant types; algebras as base-types e.g., $\mathbb{N}$

$$0 : \mathbb{N} \,|\, \mathrm{S} : \mathbb{N} \to \mathbb{N}$$

- Terms including constants defined by computation rules
- Formulas and Predicates

$$\textbf{Form} ::= P\vec{x} \,\Big|\, \forall_x A \,\Big|\, \exists_x A \,\Big|\, A \to B \,\Big|\, A \wedge B \,\Big|\, A \vee B$$

$$\textbf{Pred} ::= X \,\Big|\, \{\vec{x} \,|\, A\} \,|\, \mathbf{I} \,|\, {}^{co}\mathbf{I}$$

- Derivations by minimal logic with axioms for lfp,gfp

Example: stongly uneven

Even numbers may be inductively defined as the least predicate on natural numbers closed under the rules

$$0 \in \mathbf{Ev}, \qquad \forall_n(n \in \mathbf{Ev} \to (\mathtt{SS}n \in \mathbf{Ev})).$$

$n : \mathbb{N}$ is strongly not even if we have evidence that the two rules fail for this n

$$n \not\equiv 0 \wedge \forall_m(n \equiv \mathtt{SS}m \to m \in \mathbf{Ev}^{\mathbf{N}}) \to n \in \mathbf{Ev}^{\mathbf{N}},$$

Define $\mathbf{Ev}^{\mathbf{N}}$ as the greatest fixed point

$$\mathbf{Ev}^{\mathbf{N}} := \nu_X\big(\forall_n(n \not\equiv 0 \wedge \forall_m(n \equiv \mathtt{SS}m \to m \in X) \to n \in X)\big)$$

Then for all $n : \mathbb{N}$, we get $(2n+1) \in Ev^{\mathbf{N}}$ and $\infty \in Ev^{\mathbf{N}}$.

Example: stongly uneven

Even numbers may be inductively defined as the least predicate on natural numbers closed under the rules

$$0 \in \mathbf{Ev}, \qquad \forall_n(n \in \mathbf{Ev} \to (\mathtt{SS}n \in \mathbf{Ev})).$$

$n : \mathbb{N}$ is strongly not even if we have evidence that the two rules fail for this n

$$n \not\equiv 0 \wedge \forall_m(n \equiv \mathtt{SS}m \to m \in \mathbf{Ev}^{\mathbf{N}}) \to n \in \mathbf{Ev}^{\mathbf{N}},$$

Define $\mathbf{Ev}^{\mathbf{N}}$ as the greatest fixed point

$$\mathbf{Ev}^{\mathbf{N}} := \nu_X\left(\forall_n(n \not\equiv 0 \wedge \forall_m(n \equiv \mathtt{SS}m \to m \in X) \to n \in X)\right)$$

Then for all $n : \mathbb{N}$, we get $(2n+1) \in Ev^{\mathbf{N}}$ and $\infty \in Ev^{\mathbf{N}}$.

Example: stongly uneven

Even numbers may be inductively defined as the least predicate on natural numbers closed under the rules

$$0 \in \mathbf{Ev}, \qquad \forall_n(n \in \mathbf{Ev} \to (\mathtt{SS}n \in \mathbf{Ev})).$$

$n : \mathbb{N}$ is strongly not even if we have evidence that the two rules fail for this n

$$n \not\equiv 0 \wedge \forall_m(n \equiv \mathtt{SS}m \to m \in \mathbf{Ev}^{\mathbf{N}}) \to n \in \mathbf{Ev}^{\mathbf{N}},$$

Define $\mathbf{Ev}^{\mathbf{N}}$ as the greatest fixed point

$$\mathbf{Ev}^{\mathbf{N}} := \nu_X\big(\forall_n(n \not\equiv 0 \wedge \forall_m(n \equiv \mathtt{SS}m \to m \in X) \to n \in X)\big)$$

Then for all $n : \mathbb{N}$, we get $(2n+1) \in Ev^{\mathbf{N}}$ and $\infty \in Ev^{\mathbf{N}}$.

$\mathbf{I}$ least-fix-point of clauses $\mathbf{I} := \mu_X\left(\forall_{\vec{x}_i}(\vec{A}_i(X) \to X\vec{t}_i)\right)_{i<k}$

$$\vec{y} \in \mathbf{I} \leftrightarrow \bigvee_{i<k} \exists_{x_i}(\vec{y} \equiv \vec{t}_i \bigwedge \vec{A}_i)$$

How to define strong negation?

$$\vec{y} \in \mathbf{I}^{\mathbf{N}} \leftrightarrow \bigwedge_{i<k} \forall_{x_i}(\vec{y} \equiv \vec{t}_i \to \bigvee \vec{A}_i^{\mathbf{N}})$$

$\mathbf{I}^{\mathbf{N}}$ is the gfp given by

$$\mathbf{I}^{\mathbf{N}} := \nu_{X'}\left(\forall_{\vec{y}}\left((\forall_{\vec{x}_i}(\vec{y} \equiv \vec{t}_i \to \bigvee_{\nu<n_i} A_{i\nu}^{\mathbf{N}}))_{i<k} \to X'\vec{y}\right)\right),$$

and $\bigvee_{\nu<0} := \mathtt{t} \equiv \mathtt{f}$.

$\mathbf{I}$ least-fix-point of clauses $\mathbf{I} := \mu_X \left(\forall_{\vec{x}_i}(\vec{A}_i(X) \to X\vec{t}_i)\right)_{i<k}$

$$\vec{y} \in \mathbf{I} \leftrightarrow \bigvee_{i<k} \exists_{x_i}(\vec{y} \equiv \vec{t}_i \bigwedge \vec{A}_i)$$

How to define strong negation?

$$\vec{y} \in \mathbf{I}^{\mathbf{N}} \leftrightarrow \bigwedge_{i<k} \forall_{x_i}(\vec{y} \equiv \vec{t}_i \to \bigvee \vec{A}_i^{\mathbf{N}})$$

$\mathbf{I}^{\mathbf{N}}$ is the gfp given by

$$\mathbf{I}^{\mathbf{N}} := \nu_{X'}\left(\forall_{\vec{y}}\left((\forall_{\vec{x}_i}(\vec{y} \equiv \vec{t}_i \to \bigvee_{\nu<n_i} A_{i\nu}^{\mathbf{N}}))_{i<k} \to X'\vec{y}\right)\right),$$

and $\bigvee_{\nu<0} := \mathtt{t} \equiv \mathtt{f}$.

$\mathbf{I}$ least-fix-point of clauses $\mathbf{I} := \mu_X\left(\forall_{\vec{x}_i}(\vec{A}_i(X) \to X\vec{t}_i)\right)_{i<k}$

$$\vec{y} \in \mathbf{I} \leftrightarrow \bigvee_{i<k} \exists_{x_i}(\vec{y} \equiv \vec{t}_i \bigwedge \vec{A}_i)$$

How to define strong negation?

$$\vec{y} \in \mathbf{I}^{\mathbf{N}} \leftrightarrow \bigwedge_{i<k} \forall_{x_i}(\vec{y} \equiv \vec{t}_i \to \bigvee \vec{A}_i^{\mathbf{N}})$$

$\mathbf{I}^{\mathbf{N}}$ is the gfp given by

$$\mathbf{I}^{\mathbf{N}} := \nu_{X'}\Big(\forall_{\vec{y}}\big((\forall_{\vec{x}_i}(\vec{y} \equiv \vec{t}_i \to \bigvee_{\nu<n_i} A_{i\nu}^{\mathbf{N}}))_{i<k} \to X'\vec{y}\big)\Big),$$

and $\bigvee_{\nu<0} := \mathtt{t} \equiv \mathtt{f}$.

Properties

Lemma ($\mathbf{EF^N}$)

For all formulas A and B we have that

$$\left\{X' \to X \to B \,|\, X \in \mathbf{OP}(A)\right\} \vdash A^{\mathbf{N}} \to A \to B.$$

Theorem ($\mathbf{DNE^N}$)

For all formulas A we have that

$$\left.\begin{array}{l}\left\{Y' \to \neg Y \,|\, Y \in \mathbf{OP}(B) \text{ and } B \lhd^{-} A \text{ maximal}\right\} \\ \left\{X'' \to X \,|\, X \in \mathbf{OP}(A)\right\}\end{array}\right\} \vdash A^{\mathbf{NN}} \to A,$$

But $\nvdash A \to A^{\mathbf{NN}}$.

Example: Accessible part of a relation

If $\prec$ is a binary relation, the accessible (from below) part of $\prec$ is inductively defined with the introduction rule

$$\forall_x\big(\forall_y(y \prec x \to y \in \mathbf{Acc}_\prec) \to x \in \mathbf{Acc}_\prec\big)$$

i.e. $x \in \mathbf{Acc}_\prec$ if any chain $x_0 \prec \cdots \prec x$ is finite.

Its strong negation is coinductively defined with the closure rule

$$x \in \mathbf{Acc}^{\mathbf{N}}_\prec \to \exists_y\big(y \prec x \land y \in \mathbf{Acc}^{\mathbf{N}}_\prec\big),$$

and it corresponds exactly to the inaccessible part of $\prec$, namely $x \in \mathbf{Acc}^{\mathbf{N}}_\prec$ if there is an infinite chain $\cdots \prec y_1 \prec y_0 \prec x$.

Example: Accessible part of a relation

If $\prec$ is a binary relation, the accessible (from below) part of $\prec$ is inductively defined with the introduction rule

$$\forall_x\big(\forall_y(y \prec x \to y \in \mathbf{Acc}_\prec) \to x \in \mathbf{Acc}_\prec\big)$$

i.e. $x \in \mathbf{Acc}_\prec$ if any chain $x_0 \prec \cdots \prec x$ is finite.
Its strong negation is coinductively defined with the closure rule

$$x \in \mathbf{Acc}^{\mathbf{N}}_\prec \to \exists_y\big(y \prec x \land y \in \mathbf{Acc}^{\mathbf{N}}_\prec\big),$$

and it corresponds exactly to the inaccessible part of $\prec$, namely $x \in \mathbf{Acc}^{\mathbf{N}}_\prec$ if there is an infinite chain $\cdots \prec y_1 \prec y_0 \prec x$.

For more details, see:

`https://arxiv.org/abs/2210.05491`

Strong negation in the theory of computable functionals TCF

Nils Köpp, Iosif Petrakis

Sequences ordered with gap-condition

Autumn school “Proof and Computation” 2023

Patrick Uftring

September 15, 2023

University of Würzburg
Institute of Mathematics - Mathematical Logic

Higman's Lemma

Consider sequences of natural numbers:

$$\underset{s_0}{\langle 1, 1, 2 \rangle} \qquad \underset{s_1}{\langle 0, 0, 1, 0 \rangle} \qquad \underset{s_2}{\langle 1, 0, 2, 3 \rangle}$$

Higman's Lemma

Consider sequences of natural numbers:

$$\underset{s_0}{\langle 1, 1, 2 \rangle} \qquad \underset{s_1}{\langle 0, 0, 1, 0 \rangle} \qquad \underset{s_2}{\langle 1, 0, 2, 3 \rangle}$$

We have $s_0 \leq s_2$:

$$\begin{array}{ccccc} 1 & & 1 & 2 \\ \leq & & \leq & \leq \\ 1 & 0 & 2 & 3 \end{array}$$

Higman's Lemma

Consider sequences of natural numbers:

$$\underset{s_0}{\langle 1,1,2\rangle} \qquad \underset{s_1}{\langle 0,0,1,0\rangle} \qquad \underset{s_2}{\langle 1,0,2,3\rangle}$$

We have $s_0 \leq s_2$:

$$\begin{array}{cccc} 1 & & 1 & 2 \\ \leq & & \leq & \leq \\ 1 & 0 & 2 & 3 \end{array}$$

Higman's Lemma proves the existence of $i < j$ with $s_i \leq s_j$ for any list of sequences $(s_n)_{n\in\mathbb{N}}$.

Higman's Lemma

Consider sequences of natural numbers:

$$\underset{s_0}{\langle 1,1,2\rangle} \qquad \underset{s_1}{\langle 0,0,1,0\rangle} \qquad \underset{s_2}{\langle 1,0,2,3\rangle}$$

We have $s_0 \leq s_2$:

$$\begin{array}{cccc} 1 & & 1 & 2 \\ \leq & & \leq & \leq \\ 1 & 0 & 2 & 3 \end{array}$$

Higman's Lemma proves the existence of $i < j$ with $s_i \leq s_j$ for any list of sequences $(s_n)_{n\in\mathbb{N}}$.

Provable in ACA_0.

Gap embeddability

Any element in a gap must be greater or equal to the next mapped element

$$\begin{array}{ccccc} 1 & & 1 & 2 \\ \leq & & \leq & \leq \\ 1 & 0 & 2 & 3 \end{array}$$

Gap embeddability

Any element in a gap must be greater or equal to the next mapped element

$$\begin{array}{ccccc} 1 & & 1 & 2 \\ \leq & & \leq & \leq \\ 1 & 0 & 2 & 3 \end{array}$$

We can embed ε_0 into such sequences $\Rightarrow$ Not provable in ACA_0.

Gap embeddability

Any element in a gap must be greater or equal to the next mapped element

$$\begin{array}{ccccc} 1 & & 1 & & 2 \\ \leq & & \leq & & \leq \\ 1 & 0 & 2 & & 3 \end{array}$$

We can embed ε_0 into such sequences $\Rightarrow$ Not provable in ACA_0.

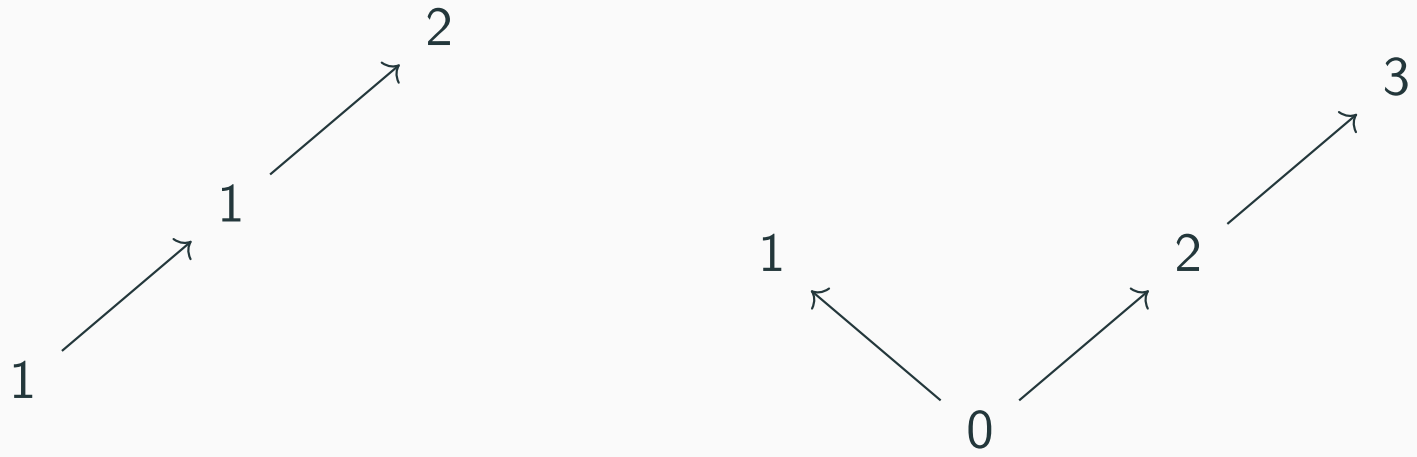
Result by Gordeev (1989): Provable in ATR_0 (but hard)

(Some of my) current work

New idea: Convert $\langle 1, 1, 2 \rangle$ and $\langle 1, 0, 2, 3 \rangle$ to trees:

(Some of my) current work

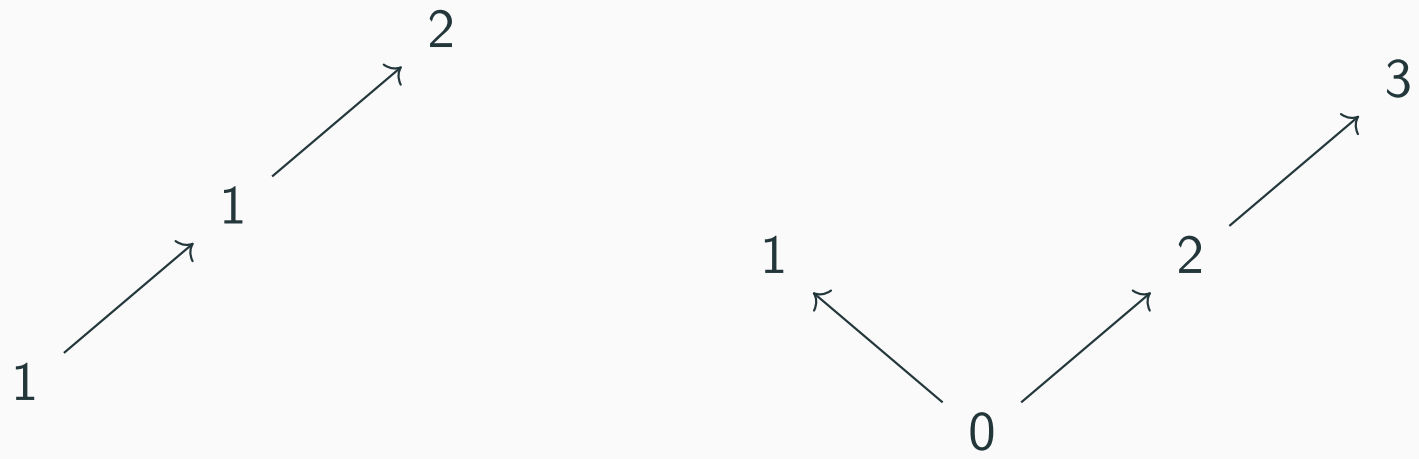
New idea: Convert $\langle 1, 1, 2 \rangle$ and $\langle 1, 0, 2, 3 \rangle$ to trees:



This is a quasi-embedding and we understand trees much better.

(Some of my) current work

New idea: Convert $\langle 1, 1, 2 \rangle$ and $\langle 1, 0, 2, 3 \rangle$ to trees:



This is a quasi-embedding and we understand trees much better.

Results:

- Simpler proof
- Calculation of all maximal order types
- Extension to wqo labels

Place:	Gif-sur-Yvette, France
Lab:	Deducteam/ Laboratoire Méthodes Formelles (LMF), ENS Paris-Saclay
Advisor:	Frédéric Blanqui
Title:	Translating proofs from and to Lean
Dates:	2023 – 2026

My Phd

- My PhD involves the design and implementation of an algorithm to translate between Lean (a theorem proving language popular among mathematicians) and Dedukti (a framework for translating between many different provers).
- My interests are in the **interoperability** of proof systems, both in terms of **direct translation** and the implementation of generic **user-facing tools**.
- I am also interested in contributing to the **user-friendliness** aspects of proof assistants (e.g. editor extensions, elaborators, compilers, UI frontends).

Following slides are taken from Frédéric Blanqui's CICM23 talk (https://blanqui.gitlabpages.inria.fr/talks/cicm23.pdf).

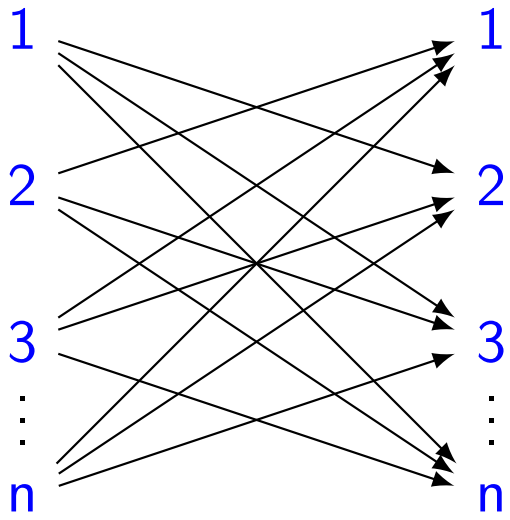
Interest of proof system interoperability

- Avoid duplicating developments and losing time
- Facilitate development of new proofs and new systems
- Increase reliability of formal proofs (cross-checking)
- Facilitate validation by certification authorities
- Relativize the choice of a system (school, industry)
- Provide multi-system data to machine learning

Difficulties of proof system interoperability

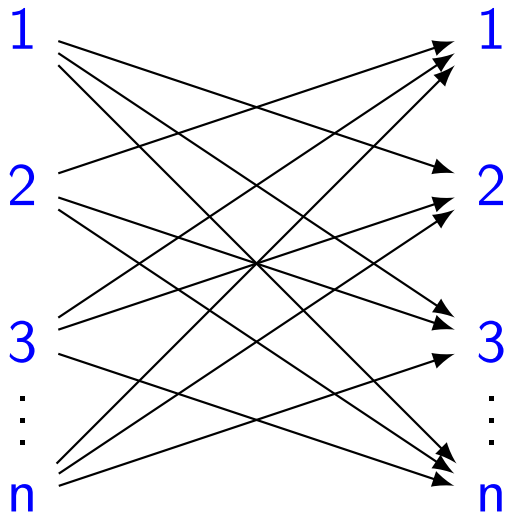
- Each system is based on different axioms and deduction rules
- It is usually non trivial and sometimes impossible to translate a proof from one system to the other (e.g. a classical proof in an intuitionistic system)

Interoperability between n systems ?



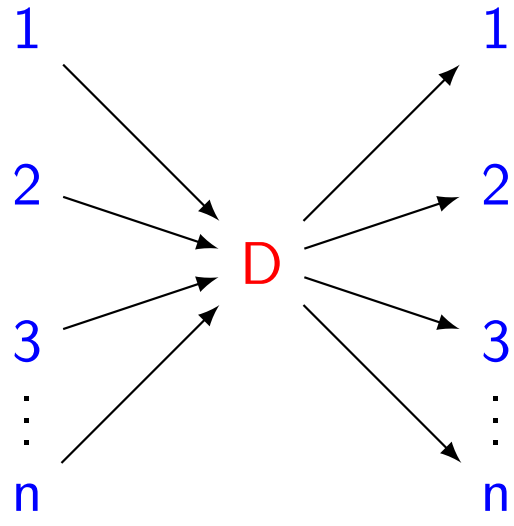
$n(n-1)$ translators

Interoperability between n systems ?



$n(n-1)$ translators

Can't we be more generic ?



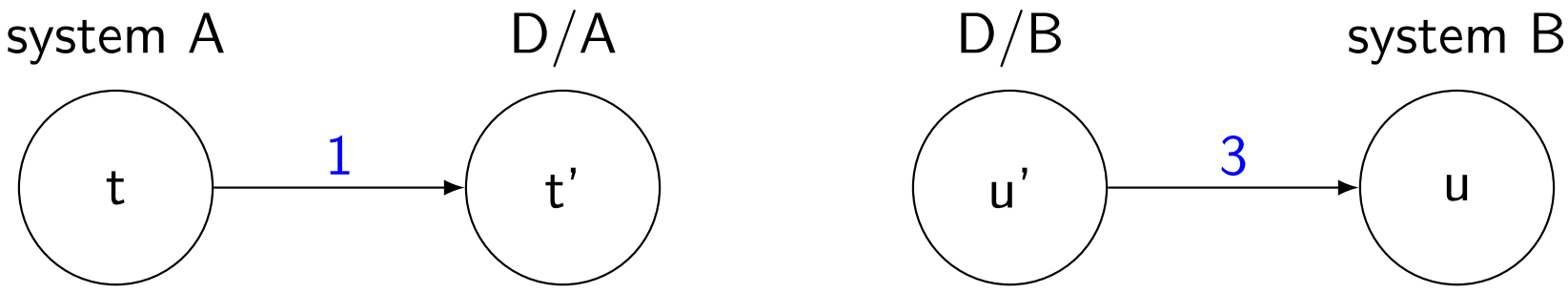
$2n$ translators

A common language for proofs?

A logical framework D

language for describing axioms, deduction rules and proofs of a system S as a theory D/S in D

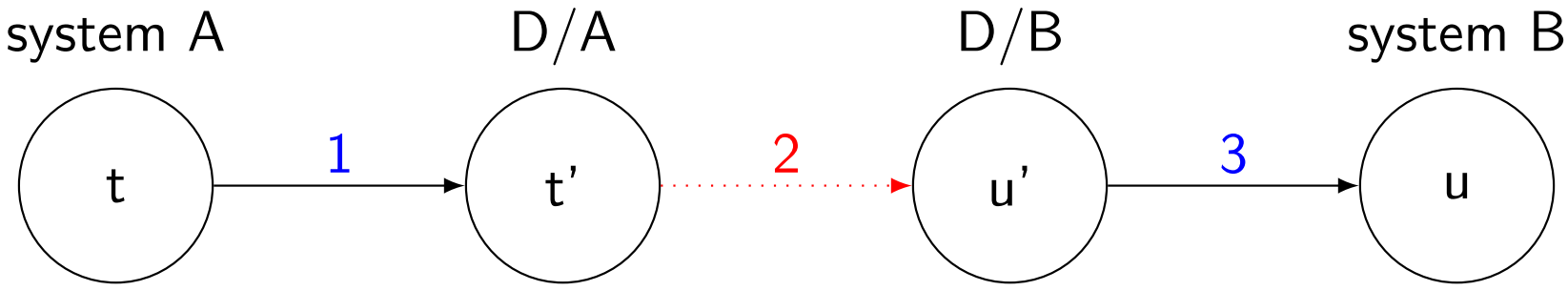
How to translate a proof $t \in A$ in a proof $u \in B$ in a logical framework D?



1. translate $t \in A$ in $t' \in D/A$

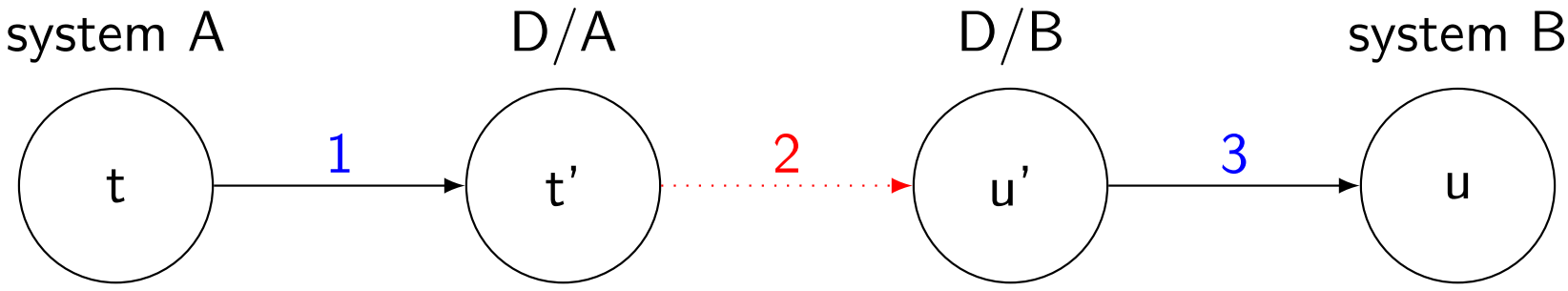
3. translate $u' \in D/B$ in $u \in B$

How to translate a proof $t \in A$ in a proof $u \in B$ in a logical framework D?



1. translate $t \in A$ in $t' \in D/A$
2. identify the axioms and deduction rules of A used in t'
   translate $t' \in D/A$ in $u' \in D/B$ if possible
3. translate $u' \in D/B$ in $u \in B$

How to translate a proof $t \in A$ in a proof $u \in B$ in a logical framework D?



1. translate $t \in A$ in $t' \in D/A$
2. identify the axioms and deduction rules of A used in t'
   translate $t' \in D/A$ in $u' \in D/B$ if possible
3. translate $u' \in D/B$ in $u \in B$

$\Rightarrow$ equally represent functionalities common to A and B

A common language for proofs?

A logical framework D

language for describing axioms, deduction rules and proofs of a system S as a theory D/S in D

Example: D = predicate calculus

allows one to represent S=geometry, S=arithmetic, S=set theory, ...
not well suited for computation and dependent types

A common language for proofs?

A logical framework D

language for describing axioms, deduction rules and proofs of a system S as a theory D/S in D

Example: D = predicate calculus

allows one to represent S=geometry, S=arithmetic, S=set theory, ...
not well suited for computation and dependent types

Better: $D = \lambda\Pi$-calculus modulo rewriting/Dedukti

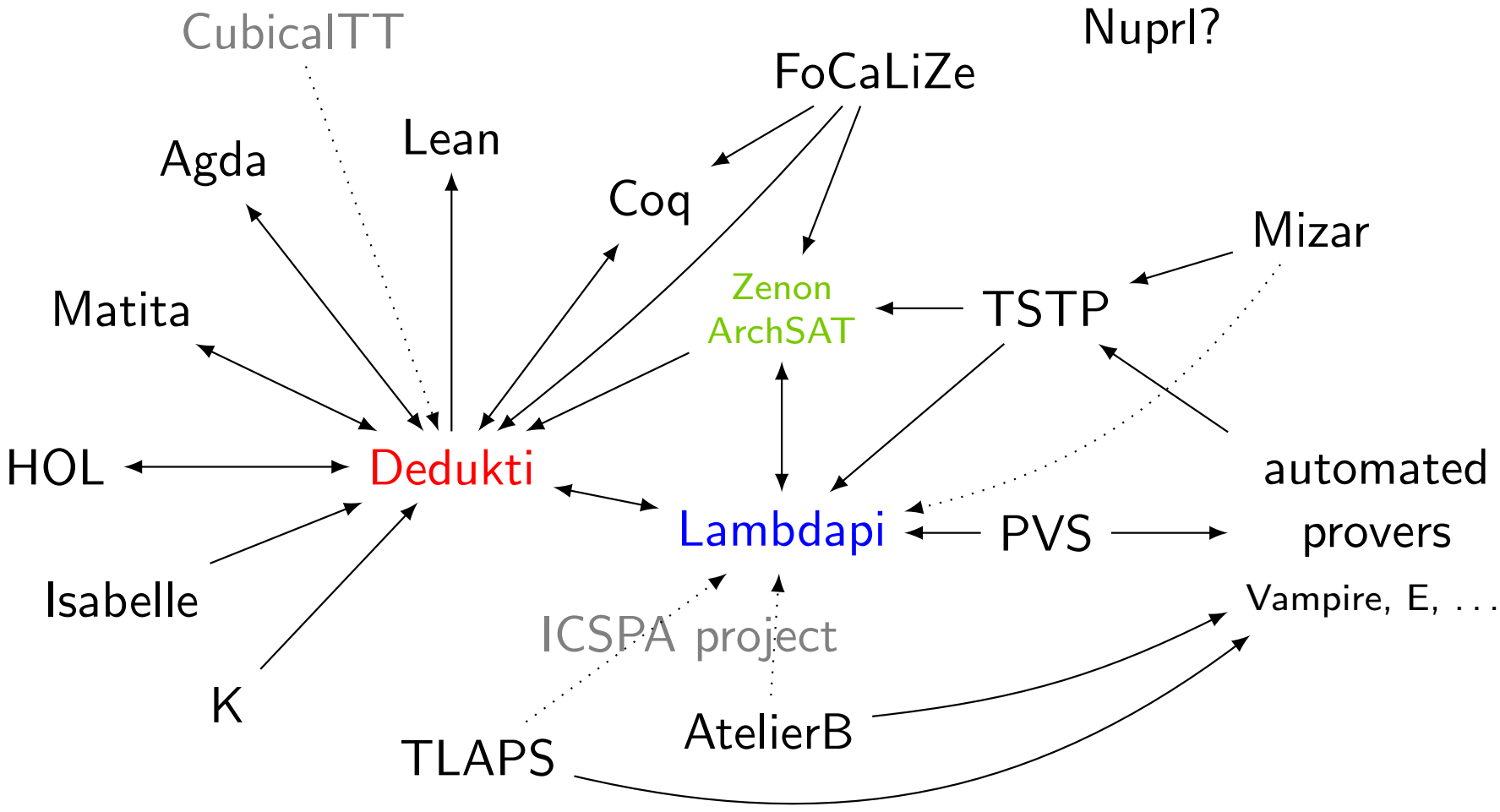
allows one to represent also:
S=HOL, S=Coq, S=Agda, S=PVS, ...

other options: λProlog, Twelf, Isabelle, Metamath, MMT...

The Dedukti world

- Zenon, ArchSAT, iProverModulo: ATPs generating Dedukti
- Holide: translator from OpenTheory to Dedukti
- Krajono: translator from Matita to Dedukti
- CoqInE: translator from Coq to Dedukti
- isabelle_dedukti: translator from Isabelle to Dedukti
- hol2dk: translator from HOL-Light to Dedukti and Lambdapi
- Agda2Dedukti: translator from Agda to Dedukti
- personoj: translator from PVS to Lambdapi
- ekstrakto: translator from TSTP to Lambdapi
- B-pog-translator: translator from Atelier B to Lambdapi
- sttfaxport: translator from Dedukti to OpenTheory, Matita, Coq, PVS and Lean3
- lambdapi: translator from Dedukti to Lambdapi, and from Lambdapi to Dedukti and Coq
- ...

Dedukti, an assembly language for proof systems



Lambdapi = Dedukti + implicit arguments/coercions, tactics, ...

https://github.com/Deducteam/Dedukti
https://github.com/Deducteam/lambdapi

What is the $\lambda\Pi$-calculus modulo rewriting?

$\lambda\Pi/\mathcal{R} = \lambda$ simply-typed λ-calculus

$+ \Pi$ dependent types, e.g. Array n

$+ \mathcal{R}$ identification of types modulo rewrites rules $l \hookrightarrow r$

What is the $\lambda\Pi$-calculus modulo rewriting?

$\lambda\Pi/\mathcal{R} = \lambda$ simply-typed λ-calculus

$+ \Pi$ dependent types, e.g. Array n

$+ \mathcal{R}$ identification of types modulo rewrites rules $l \hookrightarrow r$

---

typing = typing of Edinburg's Logical Framework LF including:

$$\text{(abs)}\ \frac{\Gamma, x : A \vdash t : B \quad \Gamma \vdash \Pi x : A, B : \mathtt{TYPE}}{\Gamma \vdash \lambda x : A, t : \Pi x : A, B}$$

$x \notin \Gamma$: types of local variables

$$\text{(app)}\ \frac{\Gamma \vdash t : \Pi x : A, B \quad \Gamma \vdash u : A}{\Gamma \vdash tu : B\{x \mapsto u\}}$$

$+$ the rule $\quad \text{(conv)}\ \dfrac{\Gamma \vdash t : A \quad A \equiv_{\beta\mathcal{R}} B}{\Gamma \vdash t : B}$

$\equiv_{\beta\mathcal{R}}$: equational theory generated by β and $\mathcal{R}$

---

concat : $\Pi p : \mathbb{N}$,Array $p \to \Pi q : \mathbb{N}$,Array $q \to$ Array$(p + q)$

concat 2 a 3 b : Array$(2 + 3) \equiv_{\beta\mathcal{R}}$ Array(5)

Functional Pure Type Systems $(\mathcal{S}, \mathcal{A}, \mathcal{P})$ $\mathcal{A} \subseteq \mathcal{S}^2, \mathcal{P} \subseteq \mathcal{S}^2 \times \mathcal{S}$

terms and types:

$$t := x \mid tt \mid \lambda x : t, t \mid \Pi x : t, t \mid s \in \mathcal{S}$$

typing rules:

$$\frac{}{\emptyset \vdash} \qquad \frac{\Gamma \vdash A : s}{\Gamma, x : A \vdash} \qquad \frac{\Gamma \vdash \quad (x, A) \in \Gamma}{\Gamma \vdash x : A}$$

$$(sort)\ \frac{\Gamma \vdash \quad (s_1, s_2) \in \mathcal{A}}{\Gamma \vdash s_1 : s_2}$$

$$(prod)\ \frac{\Gamma \vdash A : s_1 \quad \Gamma, x : A \vdash B : s_2 \quad ((s_1, s_2), s_3) \in \mathcal{P}}{\Gamma \vdash \Pi x : A, B : s_3}$$

$$\frac{\Gamma, x : A \vdash t : B \quad \Gamma \vdash \Pi x : A, B : s}{\Gamma \vdash \lambda x : A, t : \Pi x : A, B} \qquad \frac{\Gamma \vdash t : \Pi x : A, B \quad \Gamma \vdash u : A}{\Gamma \vdash tu : B\{(x, u)\}}$$

$$\frac{\Gamma \vdash t : A \quad A \simeq_\beta A' \quad \Gamma \vdash A' : s}{\Gamma \vdash t : A'}$$

Encoding Functional Pure Type Systems

[Cousineau & Dowek 2007]

signature:

$U_s : \mathtt{TYPE}$ for each sort $s \in \mathcal{S}$

$El_s : U_s \to \mathtt{TYPE}$

$s_1 : U_{s_2}$ for every $(s_1, s_2) \in \mathcal{A}$

$\pi_{s_1,s_2} : \Pi a : U_{s_1}, (El_{s_1}\ a \to U_{s_2}) \to U_{s_3}$ for every $((s_1, s_2), s_3) \in \mathcal{P}$

rules:

$El_{s_2}\ s_1 \hookrightarrow U_{s_1}$ for every $(s_1, s_2) \in \mathcal{A}$

$El_{s_3}(\pi_{s_1,s_2}\ a\ b) \hookrightarrow \Pi x : El_{s_1}\ a, El_{s_2}(b\ x)$ for every $((s_1, s_2), s_3) \in \mathcal{P}$

encoding:

$|x|_\Gamma = x$

$|s|_\Gamma = s$

$|\lambda x : A, t|_\Gamma = \lambda x : El_s |A|_\Gamma, |t|_{\Gamma,x:A}$ if $\Gamma \vdash A : s$

$|tu|_\Gamma = |t|_\Gamma |u|_\Gamma$

$|\Pi x : A, B|_\Gamma = \pi_{s_1,s_2} |A|_\Gamma (\lambda x : El_{s_1} |A|_\Gamma, |B|_{\Gamma,x:A})$

if $\Gamma \vdash A : s_1$ and $\Gamma, x : A \vdash B : s_2$

Lean's Construction

Terms t in Lean are constructed from the following (simplified) grammar:

$$\ell := 0 \mid S\ell \mid \mathsf{imax}(\ell, \ell') \mid \mathsf{max}(\ell, \ell') \mid v$$
$$t := x \mid \mathsf{U}_\ell \mid t\ t' \mid \lambda x : t.\ t' \mid \forall x : t.\ t'$$

where $v \in V$, a set of universe variables.

Lean as a Pure Type System

A subset of Lean's typing rules coincide those of a Pure Type System when we set:

$$
\begin{aligned}
S &= \{s_\ell \text{ for all levels } \ell\} \\
A &= \{\langle \ell, S\ell \rangle \text{ for all levels } \ell\} \\
R &= \{\langle \ell, \ell', \mathrm{imax}(\ell, \ell') \rangle \text{ for all levels } \ell\}
\end{aligned}
$$

We define the semantics of the imax ("impredicative max") operator as follows:

$$
\mathrm{imax}(\ell_1, \ell_2) = \begin{cases} 0, & \ell_2 = 0 \\ \max(\ell_1, \ell_2), & \text{otherwise} \end{cases}
$$

Encoding Additional Features

- Lean equates universes up to *semantic equality*. However, Dedukti has no knowledge of this form of equality – it expects equality modulo beta-reduction + rewrite rules.
- Additionally, the presence of universe variables means we cannot expect equality to arise by recursively applying the universe semantics.
  For example, the following two universe terms are equivalent (for every possible substitution of the variables):

$$s(\mathsf{imax}(\mathsf{imax}(u, v), \mathsf{imax}(\mathsf{imax}(u, v), w))) =_{\mathcal{L}} \mathsf{max}(s(w), \mathsf{imax}(s(\mathsf{imax}(u, v)), w))$$

Encoding Additional Features

- Checking universe equality is decidable via a bidirectional inequality checking algorithm, however Dedukti cannot simply accept the equality of terms based on the boolean output of a program.
- Therefore, we are led to construct a **normal form** for universe terms:

$$\mathsf{maxS}(\mathcal{A}(\{u, v, w\}, u, 1), \mathcal{A}(\{v, w\}, v, 1), \mathcal{A}(\{w\}, w, 1), \mathcal{B}(\{\}, 1))$$

where maxS is agnostic to the order of its arguments (implementable with "defac" in Dedukti for defining operators modulo associativity/commutativity).

Encoding Additional Features

Many features of Lean remain to be translated, including:

- Proof irrelevance (normal form for proof terms?)
- η-equivalence for functions/structures
- Quotient reduction rules
- Axioms (choice, functional/propositional extensionality)
- Mutual + nested inductives
- Mutual definitions

Groups as Types

ϟ Talk

Tom de Jong

j.w.w. Ulrik Buchholtz and Egbert Rijke

Proof and Computation

15 September 2023

Higher types

- We work in homotopy type theory (HoTT).

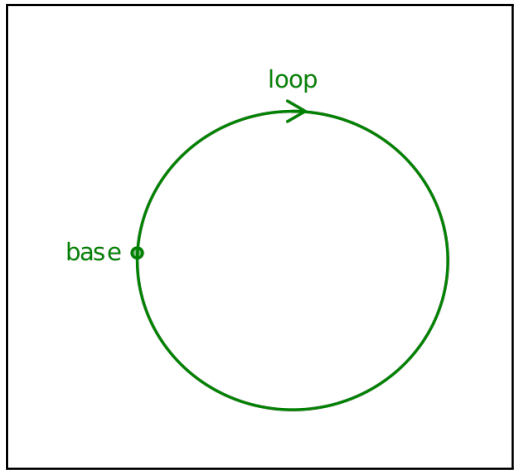
Higher types

- We work in homotopy type theory (HoTT).
- The circle $\mathbb{S}^1$

Higher inductive type

$\text{base} : \mathbb{S}^1$

$\text{loop} : \text{base} = \text{base}$



is a higher type:

$$(\text{base} = \text{base}) \simeq \mathbb{Z}.$$

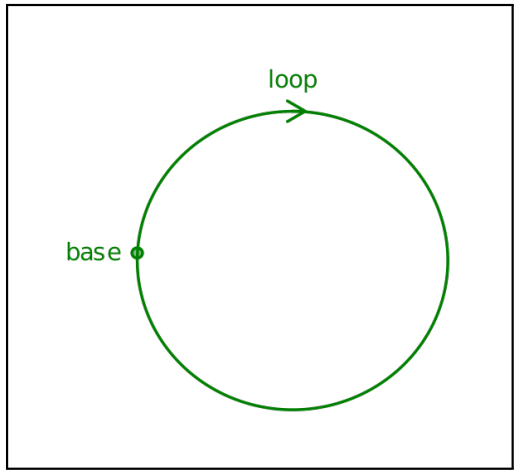
Higher types

- We work in homotopy type theory (HoTT).
- The circle $\mathbb{S}^1$

Higher inductive type

$\mathsf{base} : \mathbb{S}^1$

$\mathsf{loop} : \mathsf{base} = \mathsf{base}$



is a higher type:

$$(\mathsf{base} = \mathsf{base}) \simeq \mathbb{Z}.$$

- Everything we can do with the group $\mathbb{Z}$, we can also do with the (loop space of) the circle.

Groups as (certain) types

- In fact, given *any* group G, we can construct a (connected) 1-type $\mathrm{B}G$ with a point $\mathrm{pt} : \mathrm{B}G$ such that $(\mathrm{pt} = \mathrm{pt}) \simeq G$.

[1] `https://unimath.github.io/SymmetryBook/book.pdf`

Groups as (certain) types

- In fact, given *any* group G, we can construct a (connected) 1-type $\mathrm{B}G$ with a point $\mathrm{pt} : \mathrm{B}G$ such that $(\mathrm{pt} = \mathrm{pt}) \simeq G$.

- Group theory = the theory of pointed connected 1-types

  See the *Symmetry*[1] book by Bezem, Buchholtz, Cagne, Dundas and Grayson.

[1] `https://unimath.github.io/SymmetryBook/book.pdf`

Groups as (certain) types

- In fact, given *any* group G, we can construct a (connected) 1-type $\mathrm{B}G$ with a point $\mathrm{pt} : \mathrm{B}G$ such that $(\mathrm{pt} = \mathrm{pt}) \simeq G$.

- Group theory = the theory of pointed connected 1-types

  See the *Symmetry*[1] book by Bezem, Buchholtz, Cagne, Dundas and Grayson.

- This perspective allows for some very slick group-theoretic constructions and proofs.

  I will briefly talk about once such proof.

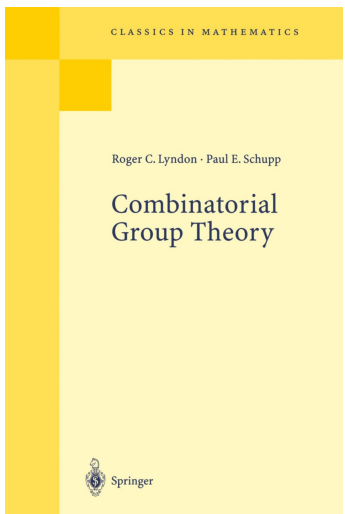
[1] `https://unimath.github.io/SymmetryBook/book.pdf`

The Higman group, traditionally

- The Higman group is defined using only 4 generators and 4 relations. It has remarkable group-theoretic properties.

The Higman group, traditionally

- The Higman group is defined using only 4 generators and 4 relations. It has remarkable group-theoretic properties.
- The group can be shown to be nontrivial, but the proof requires combinatorial group theory.



The Higman group in HoTT

- We can *completely avoid* combinatorial group theory!

[2] `https://dwarn.se/po-paths.pdf`

The Higman group in HoTT

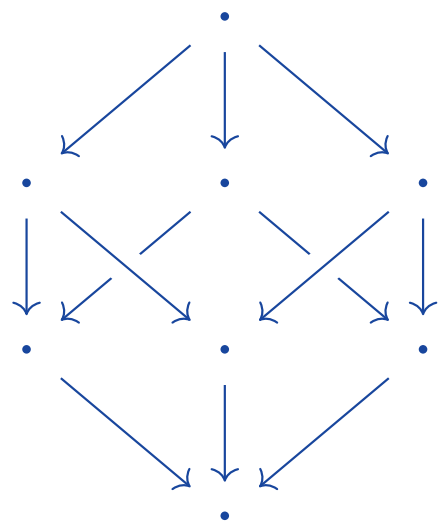
- We can *completely avoid* combinatorial group theory!
- Instead, we use tools from higher topos/type theory.
  - A recent result of David Wärn[2] on pushouts.
  - Descent: Interplay between pullbacks and pushouts.

[2] `https://dwarn.se/po-paths.pdf`

The Higman group in HoTT

- We can *completely avoid* combinatorial group theory!
- Instead, we use tools from higher topos/type theory.
  - A recent result of David Wärn[2] on pushouts.
  - Descent: Interplay between pullbacks and pushouts.

    If the bottom and top squares of a commutative cube are pushouts and the back sides are pullbacks,



    then the front sides are pullbacks too.

[2] `https://dwarn.se/po-paths.pdf`

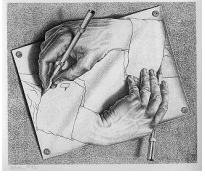
GL within HOL Light

Experiments on theorem provers within theorem provers

A project within the project "IT Matters: Methods and Tools for Trustworthy Smart Systems" (PRIN 2017FTXR7S)

Cosimo Perini Brogi
IMT School for Advanced Studies Lucca

"Proof and Computation" Autumn School 2023

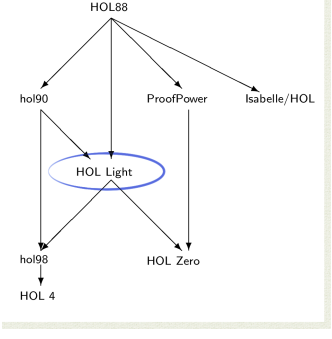


Picture credit: "The Magic of M.C. Escher", WikiMedia Commons

IMT SCHOOL FOR ADVANCED STUDIES LUCCA

Brief glance at HOL Light

(Harrison 2017)

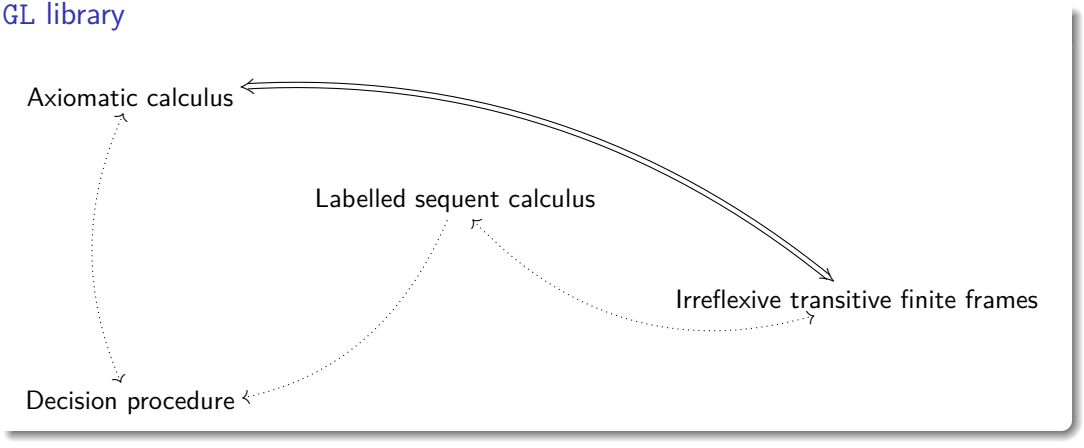


- Clean logical foundations ≈ *Principia Mathematica*
- LCF-style proof checker based on polymorphic simple type theory ≈ small class of *primitive inference rules* for creating theorems + *derived inference rules* to be programmed on top
  - ⇒ 10 primitive rules
  - ⇒ 2 conservative extension principles
  - ⇒ Axioms of choice, extensionality, and infinity
- Written as an OCaml program ≈ three datatypes for the logic: `hol_type`, `term`, and `thm`
- Goal-directed proof development ≈ tactic(al)s + automated methods (in the appropriate domains)

Despite its simple foundations, HOL Light includes a large library of mathematical results in topology, analysis, Euclidean geometry, QBF, floating point algorithms, FOL, limitative results, ...

The current prototype

GL library



Code: https://github.com/jrh13/hol-light/, directory GL

Paper: *Mechanising Gödel-Löb provability logic in HOL Light*, J. Autom. Reasoning 67, 29 (Open Access)

Short Demo

Many thanks for listening!