Some lessons after the formalization of the ctm approach to forcing

P. Sánchez Terraf[1]

Joint work with Emmanuel Gunther, Miguel Pagano, and Matías Steinberg

CIEM-FaMAF — Universidad Nacional de Córdoba

Coloquio Latinoamericano de Matemáticxs

Uruguay (Virtual), 13 / 09 / 2021

[1] Supported by Secyt-UNC project 33620180100465CB and Conicet.

I would like to acknowledge

- Ken Kunen for his inspiring expositions. This formalization is a tiny homage to his memory.
- Larry Paulson for his work and encouragement.
- Mikhail Mandrykin and the rest of the Isabelle community for support.
- The organizers for their invitation.

Verification

Formal verification: the use of *proof assistants* to check that a proof is correct to the finest detail.
No step is omitted, tracing the result to be proved to the very axioms.

Verification

Formal verification: the use of *proof assistants* to check that a proof is correct to the finest detail.
No step is omitted, tracing the result to be proved to the very axioms.

Heads up!

$$\text{Formalized proof} \neq \text{Automatic proof}$$

Verification

Formal verification: the use of *proof assistants* to check that a proof is correct to the finest detail.
No step is omitted, tracing the result to be proved to the very axioms.

Heads up!

$$\text{Formalized proof} \neq \text{Automatic proof}$$

Tools

Agda, Coq, HOL Light, ACL2, Lean, Isabelle.
These support a variety of logics [Avigad, 2021].

Verification

Notable formalization projects

- The Four Color Theorem [Gonthier, 2008], the odd order theorem [Gonthier et al., 2013].
- Univalent mathematics / homotopy type theory [Univalent Foundations Program, 2013], originally driven by Voevodsky.
- *Flyspeck* project [Hales et al., 2017, Forum Math. Pi] on Kepler's Conjecture.

Verification

Notable formalization projects

- The Four Color Theorem [Gonthier, 2008], the odd order theorem [Gonthier et al., 2013].
- Univalent mathematics / homotopy type theory [Univalent Foundations Program, 2013], originally driven by Voevodsky.
- *Flyspeck* project [Hales et al., 2017, Forum Math. Pi] on Kepler's Conjecture.

Ongoing formalization efforts

- Isabelle: HOL, AFP. See also Paulson [2018].
- Lean: `mathlib`-related projects promoted by Buzzard [Lean Community, 2021].

Programming, Meta, Logic, Internal

Isabelle has several layers.

Programming, Meta, Logic, Internal

Isabelle has several layers.

- The ML programming language.

Programming, Meta, Logic, Internal

Isabelle has several layers.

- The ML programming language.
- Meta-logic Pure: an intuitionistic fragment of higher order logic, with logical primitives are $\equiv$, $\Longrightarrow$, and $\bigwedge$ (meta-universal quantification), which operate on the meta-Booleans `prop` ⇝ rules.

Programming, Meta, Logic, Internal

Isabelle has several layers.

- The ML programming language.
- Meta-logic Pure: an intuitionistic fragment of higher order logic, with logical primitives are $\equiv$, $\Longrightarrow$, and $\bigwedge$ (meta-universal quantification), which operate on the meta-Booleans `prop` ⇝ rules.
- The object logic ZF [Paulson and Grabczewski, 1996] axiomatized over Pure.
  - It postulates two types: `i` (sets) and `o` (booleans).
  - Each of the Replacement and Separation axiom schemes feature one free higher order variables (a notational variant of NGB).

Programming, Meta, Logic, Internal

Isabelle has several layers.

- The ML programming language.
- Meta-logic Pure: an intuitionistic fragment of higher order logic, with logical primitives are $\equiv$, $\Longrightarrow$, and $\bigwedge$ (meta-universal quantification), which operate on the meta-Booleans `prop` ⇝ rules.
- The object logic ZF [Paulson and Grabczewski, 1996] axiomatized over Pure.
  - It postulates two types: `i` (sets) and `o` (booleans).
  - Each of the Replacement and Separation axiom schemes feature one free higher order variables (a notational variant of NGB).
- Inside `i`, one can develop model theory: The set `formula` of codes for first order formulas, satisfaction.

Programming, Meta, Logic, Internal

Isabelle has several layers.

- The ML programming language.
- Meta-logic Pure: an intuitionistic fragment of higher order logic, with logical primitives are $\equiv$, $\Longrightarrow$, and $\bigwedge$ (meta-universal quantification), which operate on the meta-Booleans `prop` ⇝ rules.
- The object logic ZF [Paulson and Grabczewski, 1996] axiomatized over Pure.
  - It postulates two types: `i` (sets) and `o` (booleans).
  - Each of the Replacement and Separation axiom schemes feature one free higher order variables (a notational variant of NGB).
- Inside `i`, one can develop model theory: The set `formula` of codes for first order formulas, satisfaction.

Induction/recursion works at the object logic level: It is based on set-theoretical proofs of well-foundedness— of $\mathbb{N}$, of Ord, etc.

ZF-Constructible

The ZF session reaches Hessenberg's $|A| \times |A| = |A|$.
Our decision to use Isabelle (during 2017) was triggered by its constructibility library [Paulson, 2003].

ZF-Constructible

The ZF session reaches Hessenberg's $|A| \times |A| = |A|$.
Our decision to use Isabelle (during 2017) was triggered by its constructibility library [Paulson, 2003].
It contains:

- a development of relativization and absoluteness for classes C (considered as functions from `i` to `o`);
- the construction of the set `formula` and satisfaction;
- a version of the reflection principle; and
- the development of L and the proof that it satisfies AC.

ZF-Constructible

The ZF session reaches Hessenberg's $|A| \times |A| = |A|$.
Our decision to use Isabelle (during 2017) was triggered by its constructibility library [Paulson, 2003].
It contains:

- a development of relativization and absoluteness for classes C (considered as functions from `i` to `o`);
- the construction of the set `formula` and satisfaction;
- a version of the reflection principle; and
- the development of L and the proof that it satisfies AC.

Relativization and synthesis discipline

1. $p = \mathtt{Pow}(x) :: \mathtt{i}$ (original term).

ZF-Constructible

The ZF session reaches Hessenberg's $|A| \times |A| = |A|$.
Our decision to use Isabelle (during 2017) was triggered by its constructibility library [Paulson, 2003].
It contains:

- a development of relativization and absoluteness for classes C (considered as functions from `i` to `o`);
- the construction of the set `formula` and satisfaction;
- a version of the reflection principle; and
- the development of L and the proof that it satisfies AC.

Relativization and synthesis discipline

I $p = \mathtt{Pow}(x) :: \mathtt{i}$ (original term).

II $\mathtt{is\_Pow}(C, x, p) :: \mathtt{o}$ (fully relational).

ZF-Constructible

The ZF session reaches Hessenberg's $|A| \times |A| = |A|$.
Our decision to use Isabelle (during 2017) was triggered by its constructibility library [Paulson, 2003].
It contains:

- a development of relativization and absoluteness for classes C (considered as functions from `i` to `o`);
- the construction of the set `formula` and satisfaction;
- a version of the reflection principle; and
- the development of L and the proof that it satisfies AC.

Relativization and synthesis discipline

I $p = \mathtt{Pow}(x) :: \mathtt{i}$ (original term).

II $\mathtt{is\_Pow}(C, x, p) :: \mathtt{o}$ (fully relational).

III $\mathtt{is\_Pow\_fm}(0, 1, 2) :: \mathtt{i}$ (member of `formula`).

De Bruijn indices

They provide a way of writing first order formulas which simplifies the management of bound variables.

$$(\forall\,.\,0 \in 1 \longrightarrow 0 \in 2)$$

De Bruijn indices

They provide a way of writing first order formulas which simplifies the management of bound variables.

$$(\forall\,.\,0 \in 1 \longrightarrow 0 \in 2)$$

- Assignments ("environments") are given by lists of elements of the model.

$$A, [x, y, z, w] \models 0 \in 1 \longrightarrow 0 \in 2 \iff (x \in y \text{ implies } x \in z)$$

De Bruijn indices

They provide a way of writing first order formulas which simplifies the management of bound variables.

$$(\forall\,.\,0 \in 1 \longrightarrow 0 \in 2)$$

- Assignments ("environments") are given by lists of elements of the model.

$$A, [x, y, z, w] \models 0 \in 1 \longrightarrow 0 \in 2 \iff (x \in y \text{ implies } x \in z)$$

- Quantifiers shift the indexing. The index indicates "how many quantifiers to jump."

$$A, [y, z, w] \models (\forall\,.\,0 \in 1 \longrightarrow 0 \in 2) \iff (y \subseteq z)$$

The Fundamental Theorems

Let $\langle \mathbb{P}, \preceq, \mathbb{1} \rangle \in M$ be a forcing notion. Given $G \subseteq \mathbb{P}$, we have
$M[G] := \{\mathrm{val}(P, G, \dot{a}) : \dot{a} \in M\}$
The following form of the Forcing Theorems is the one that we formalized.

The Fundamental Theorems

Let $\langle \mathbb{P}, \preceq, \mathbb{1} \rangle \in M$ be a forcing notion. Given $G \subseteq \mathbb{P}$, we have
$M[G] := \{\mathrm{val}(P, G, \dot{a}) : \dot{a} \in M\}$
The following form of the Forcing Theorems is the one that we formalized.

Theorem (Cohen [1963])

There exists a function $\mathrm{forces} : \texttt{formula} \to \texttt{formula}$ *such that for every* φ *and* $\dot{a}_0, \ldots, \dot{a}_n \in M$,

Truth Lemma *for every* M*-generic* G,

$$M[G], [\mathrm{val}(P, G, \dot{a}_0), \ldots, \mathrm{val}(P, G, \dot{a}_n)] \models \varphi$$
$$\iff$$
$$\exists p \in G.\ \ M, [p, \mathbb{P}, \preceq, \mathbb{1}, \dot{a}_0, \ldots, \dot{a}_n] \models \mathrm{forces}(\varphi).$$

The Fundamental Theorems

Let $\langle \mathbb{P}, \preceq, \mathbb{1} \rangle \in M$ be a forcing notion. Given $G \subseteq \mathbb{P}$, we have
$M[G] := \{\mathrm{val}(P, G, \dot{a}) : \dot{a} \in M\}$
The following form of the Forcing Theorems is the one that we formalized.

Theorem (Cohen [1963])

There exists a function $\mathrm{forces} : \texttt{formula} \to \texttt{formula}$ *such that for every* φ *and* $\dot{a}_0, \ldots, \dot{a}_n \in M$,

Truth Lemma *for every* M*-generic* G,

$$M[G], [\mathrm{val}(P, G, \dot{a}_0), \ldots, \mathrm{val}(P, G, \dot{a}_n)] \models \varphi$$

$$\iff$$

$$\exists p \in G.\ \ M, [p, \mathbb{P}, \preceq, \mathbb{1}, \dot{a}_0, \ldots, \dot{a}_n] \models \mathrm{forces}(\varphi).$$

$$\searrow\ p \Vdash^{M}_{\mathbb{P},\preceq} \varphi(\dot{a}_0, \ldots, \dot{a}_n)\ \swarrow$$

The Fundamental Theorems

Let $\langle \mathbb{P}, \preceq, \mathbb{1} \rangle \in M$ be a forcing notion. Given $G \subseteq \mathbb{P}$, we have
$M[G] := \{\mathrm{val}(P, G, \dot{a}) : \dot{a} \in M\}$
The following form of the Forcing Theorems is the one that we formalized.

Theorem (Cohen [1963])

There exists a function $\mathrm{forces} : \texttt{formula} \to \texttt{formula}$ *such that for every* φ *and* $\dot{a}_0, \ldots, \dot{a}_n \in M$,

Truth Lemma *for every* M*-generic* G,

$$M[G], [\mathrm{val}(P, G, \dot{a}_0), \ldots, \mathrm{val}(P, G, \dot{a}_n)] \models \varphi$$

$$\Longleftrightarrow$$

$$\exists p \in G.\ \ M, [p, \mathbb{P}, \preceq, \mathbb{1}, \dot{a}_0, \ldots, \dot{a}_n] \models \mathrm{forces}(\varphi).$$

$$\searrow \quad p \Vdash \varphi\ [\dot{a}_0, \ldots, \dot{a}_n] \quad \swarrow$$

The Fundamental Theorems

Let $\langle \mathbb{P}, \preceq, \mathbb{1} \rangle \in M$ be a forcing notion. Given $G \subseteq \mathbb{P}$, we have
$M[G] := \{\mathrm{val}(P, G, \dot{a}) : \dot{a} \in M\}$
The following form of the Forcing Theorems is the one that we formalized.

Theorem (Cohen [1963])

There exists a function $\mathrm{forces} : \texttt{formula} \to \texttt{formula}$ *such that for every* φ *and* $\dot{a}_0, \ldots, \dot{a}_n \in M$,

Truth Lemma *for every* M*-generic* G,

$$M[G], [\mathrm{val}(P, G, \dot{a}_0), \ldots, \mathrm{val}(P, G, \dot{a}_n)] \models \varphi$$
$$\iff$$
$$\exists p \in G.\ \ M, [p, \mathbb{P}, \preceq, \mathbb{1}, \dot{a}_0, \ldots, \dot{a}_n] \models \mathrm{forces}(\varphi).$$
$$\searrow \quad p \Vdash \varphi\ [\dot{a}_0, \ldots, \dot{a}_n] \quad \swarrow$$

Density Lemma $p \Vdash \varphi\ [\dot{a}_0, \ldots, \dot{a}_n] \iff \{q \in \mathbb{P} : q \Vdash \varphi\ [\dot{a}_0, \ldots, \dot{a}_n]\}$ *is dense below* p.

The definition of forces

$$\texttt{forces\_eq}(p, t_1, t_2) := \forall s \in \mathrm{domain}(t_1) \cup \mathrm{domain}(t_2).\ \forall q \preceq p.$$
$$\texttt{forces\_mem}(q, s, t_1) \longleftrightarrow \texttt{forces\_mem}(q, s, t_2),$$
$$\texttt{forces\_mem}(p, t_1, t_2) := \forall v \preceq p.\ \exists q \preceq v.$$
$$\exists s.\ \exists r \in \mathbb{P}.\ \langle s, r \rangle \in t_2 \wedge q \preceq r \wedge \texttt{forces\_eq}(q, t_1, s)$$

The definition of forces

$$\texttt{forces\_eq}(p, t_1, t_2) := \forall s \in \mathrm{domain}(t_1) \cup \mathrm{domain}(t_2).\ \forall q \preceq p.\ \texttt{forces\_mem}(q, s, t_1) \longleftrightarrow \texttt{forces\_mem}(q, s, t_2),$$

$$\texttt{forces\_mem}(p, t_1, t_2) := \forall v \preceq p.\ \exists q \preceq v.\ \exists s.\ \exists r \in \mathbb{P}.\ \langle s, r\rangle \in t_2 \wedge q \preceq r \wedge \texttt{forces\_eq}(q, t_1, s)$$

We codified this as a single recursion over the well-founded relation $frecR$.

$$\texttt{forces\_at}(\mathbb{P}, \preceq, \langle ftype, t_1, t_2, p\rangle) = 1 \iff \big(ftype = 0 \wedge (\cdots \text{forcing “}=\text{”} \cdots)\big) \vee \big(ftype = 1 \wedge (\cdots \text{forcing “}\in\text{”} \cdots)\big).$$

The definition of forces

$$\texttt{forces\_eq}(p,t_1,t_2) := \forall s \in \mathrm{domain}(t_1)\cup\mathrm{domain}(t_2).\ \forall q \preceq p.\ \texttt{forces\_mem}(q,s,t_1) \longleftrightarrow \texttt{forces\_mem}(q,s,t_2),$$

$$\texttt{forces\_mem}(p,t_1,t_2) := \forall v \preceq p.\ \exists q \preceq v.\ \exists s.\ \exists r \in \mathbb{P}.\ \langle s,r\rangle \in t_2 \wedge q \preceq r \wedge \texttt{forces\_eq}(q,t_1,s)$$

We codified this as a single recursion over the well-founded relation $frecR$.

$$\texttt{forces\_at}(\mathbb{P},\preceq,\langle ftype,t_1,t_2,p\rangle) = 1 \iff \big(ftype = 0 \wedge (\cdots \text{forcing “}=\text{”} \cdots)\big) \vee \big(ftype = 1 \wedge (\cdots \text{forcing “}\in\text{”} \cdots)\big).$$

lemma *forces_induction:*
assumes
"⋀τ ϑ. ⟦⋀σ. σ∈domain(ϑ) ⟹ Q(τ,σ)⟧ ⟹ R(τ,ϑ)"
"⋀τ ϑ. ⟦⋀σ. σ∈domain(τ) ∪ domain(ϑ) ⟹ R(σ,τ) ∧ R(σ,ϑ)⟧
⟹ Q(τ,ϑ)"
shows "Q(τ,ϑ) ∧ R(τ,ϑ)"

Accomplishments

What did we formalize?

- Extended treatment of relativization.

Accomplishments

What did we formalize?

- Extended treatment of relativization.
  - We redesigned Isabelle/ZF results on non-absolute concepts to work relative to a class.
  - Missing results (from basic ones involving cardinal successors, countable sets, . . .)

Accomplishments

What did we formalize?

- Extended treatment of relativization.
  - We redesigned Isabelle/ZF results on non-absolute concepts to work relative to a class.
  - Missing results (from basic ones involving cardinal successors, countable sets, . . .)
- A model-theoretic rendition of forcing.
  - Proper extensions of ctms of ZF.
  - The construction of a ctm of $ZFC + \neg CH$ given one of ZFC.

```
theorem ctm_of_not_CH:
  assumes
    "M ≈ ω" "Transset(M)" "M ⊨ ZFC"
  shows
    "∃N.
      M ⊆ N ∧ N ≈ ω ∧ Transset(N) ∧ N ⊨ ZFC ∪ {·¬·CH··} ∧
      (∀α. Ord(α) ⟶ (α ∈ M ⟷ α ∈ N))"
```

Accomplishments

What did we formalize?

- Extended treatment of relativization.
  - We redesigned Isabelle/ZF results on non-absolute concepts to work relative to a class.
  - Missing results (from basic ones involving cardinal successors, countable sets, . . .)
- A model-theoretic rendition of forcing.
  - Proper extensions of ctms of ZF.
  - The construction of a ctm of $ZFC + \neg CH$ given one of ZFC.

No meta-theoretic issues (consistency, FOL calculi, etc) were formalized.

Accomplishments

What did we formalize?

- Extended treatment of relativization.
  - We redesigned Isabelle/ZF results on non-absolute concepts to work relative to a class.
  - Missing results (from basic ones involving cardinal successors, countable sets, . . .)
- A model-theoretic rendition of forcing.
  - Proper extensions of ctms of ZF.
  - The construction of a ctm of $ZFC + \neg CH$ given one of ZFC.

No meta-theoretic issues (consistency, FOL calculi, etc) were formalized.

What can be deduced from the present formalization?

- Which instances of Separation and Replacement are needed to set the forcing machinery up.

Accomplishments

What did we formalize?

- Extended treatment of relativization.
  - We redesigned Isabelle/ZF results on non-absolute concepts to work relative to a class.
  - Missing results (from basic ones involving cardinal successors, countable sets, . . .)
- A model-theoretic rendition of forcing.
  - Proper extensions of ctms of ZF.
  - The construction of a ctm of $ZFC + \neg CH$ given one of ZFC.

No meta-theoretic issues (consistency, FOL calculi, etc) were formalized.

What can be deduced from the present formalization?

- Which instances of Separation and Replacement are needed to set the forcing machinery up.
- A verification that no choice is needed at the level of the object logic.

A new relativization discipline

relativize functional *"cardinal" "cardinal_rel"* **external**
relationalize *"cardinal_rel" "is_cardinal"*
synthesize *"is_cardinal"* **from_definition assuming** *"nonempty"*

These commands provide:

A new relativization discipline

relativize functional *"cardinal" "cardinal_rel"* **external**
relationalize *"cardinal_rel" "is_cardinal"*
synthesize *"is_cardinal"* **from_definition assuming** *"nonempty"*

These commands provide:

1. the definition of a function (form `i` to `i`) $|\cdot|^M$;

ZF-Constructible

The ZF session reaches Hessenberg's $|A| \times |A| = |A|$.
Our decision to use Isabelle (during 2017) was triggered by its constructibility library [Paulson, 2003].
It contains:

- a development of relativization and absoluteness for classes C (considered as functions from `i` to `o`);
- the construction of the set `formula` and satisfaction;
- a version of the reflection principle; and
- the development of L and the proof that it satisfies AC.

Relativization and synthesis discipline

I $p = \mathtt{Pow}(x) :: \mathtt{i}$ (original term).

II $\mathtt{is\_Pow}(C, x, p) :: \mathtt{o}$ (fully relational).

III $\mathtt{is\_Pow\_fm}(0, 1, 2) :: \mathtt{i}$ (member of `formula`).

A new relativization discipline

relativize functional *"cardinal" "cardinal_rel"* **external**
relationalize *"cardinal_rel" "is_cardinal"*
synthesize *"is_cardinal"* **from_definition assuming** *"nonempty"*

These commands provide:

1. the definition of a function (form `i` to `i`) $|\cdot|^M$;

A new relativization discipline

relativize functional *"cardinal" "cardinal_rel"* **external**
relationalize *"cardinal_rel" "is_cardinal"*
synthesize *"is_cardinal"* **from_definition assuming** *"nonempty"*

These commands provide:

1. the definition of a function (form `i` to `i`) $|\cdot|^M$;
2. its fully relational version `is_cardinal`;

A new relativization discipline

relativize functional *"cardinal" "cardinal_rel"* **external**
relationalize *"cardinal_rel" "is_cardinal"*
synthesize *"is_cardinal"* **from_definition assuming** *"nonempty"*

These commands provide:

1. the definition of a function (form `i` to `i`) $|\cdot|^M$;
2. its fully relational version `is_cardinal`;
3. a formula `is_cardinal_fm` whose satisfaction by a set is equivalent to the relational version;

A new relativization discipline

relativize functional *"cardinal" "cardinal_rel"* **external**
relationalize *"cardinal_rel" "is_cardinal"*
synthesize *"is_cardinal"* **from_definition assuming** *"nonempty"*

These commands provide:

1. the definition of a function (form `i` to `i`) $|\cdot|^M$;
2. its fully relational version `is_cardinal`;
3. a formula `is_cardinal_fm` whose satisfaction by a set is equivalent to the relational version;
4. the proof of the latter statement.

A new relativization discipline

relativize functional *"cardinal" "cardinal_rel"* **external**
relationalize *"cardinal_rel" "is_cardinal"*
synthesize *"is_cardinal"* **from_definition assuming** *"nonempty"*

These commands provide:

1. the definition of a function (form `i` to `i`) $|\cdot|^M$;
2. its fully relational version `is_cardinal`;
3. a formula `is_cardinal_fm` whose satisfaction by a set is equivalent to the relational version;
4. the proof of the latter statement.

Recall: We can't perform induction on Isabelle terms! Done in ML.

39 replacement instances to rule them all

Combinatorics

- 7 instances for iteration of operations.
  - Fixpoints: lists (2), formulas (2).
  - nth element of a list (iteration of the tail operation).
  - Transitive closure (2).

39 replacement instances to rule them all

Combinatorics

- 7 instances for iteration of operations.
  - Fixpoints: lists (2), formulas (2).
  - nth element of a list (iteration of the tail operation).
  - Transitive closure (2).
- 5 instances for definitions by well-founded recursion.
  - Ordertypes.
  - Rank.
  - Cumulative hierarchy (rank initial segments).
  - Aleph.
  - Recursive construction of sets using a choice function (Šanin's Δ).

39 replacement instances to rule them all

Combinatorics

- 7 instances for iteration of operations.
  - Fixpoints: lists (2), formulas (2).
  - nth element of a list (iteration of the tail operation).
  - Transitive closure (2).
- 5 instances for definitions by well-founded recursion.
  - Ordertypes.
  - Rank.
  - Cumulative hierarchy (rank initial segments).
  - Aleph.
  - Recursive construction of sets using a choice function (Šanin's Δ).
- 5 other replacements.
  - Definition of ordertypes (2).
  - Definition of cardinal successor.
  - $x \mapsto \aleph_x^M$.
  - $x \mapsto |x|^M$ (Šanin's Δ)

39 replacement instances to rule them all

Combinatorics

- 7 instances for iteration of operations.
  - Fixpoints: lists (2), formulas (2).
  - nth element of a list (iteration of the tail operation).
  - Transitive closure (2).
- 5 instances for definitions by well-founded recursion.
  - Ordertypes.
  - Rank.
  - Cumulative hierarchy (rank initial segments).
  - Aleph.
  - Recursive construction of sets using a choice function (Šanin's Δ).
- 5 other replacements.
  - Definition of ordertypes (2).
  - Definition of cardinal successor.
  - $x \mapsto \aleph_x^M$.
  - $x \mapsto |x|^M$ (Šanin's Δ)

39 replacement instances to rule them all

We also need a one extra replacement instance ψ on M for each φ of the previous ones to have them in $M[G]$ (except from the two neeeded for the Δ-system lemma).

$$\psi(x, \alpha, y_1, \ldots, y_n) := \ulcorner \alpha = \min\{\beta \mid \exists \tau \in V_\beta .\ snd(x) \Vdash \varphi\ [fst(x), \tau, y_1, \ldots, y_n]\}\urcorner$$

That makes $17 + 15 = 32$ up to now.

39 replacement instances to rule them all

We also need a one extra replacement instance ψ on M for each φ of the previous ones to have them in $M[G]$ (except from the two neeeded for the Δ-system lemma).

$$\psi(x, \alpha, y_1, \ldots, y_n) := \ulcorner \alpha = \min\{\beta \mid \exists \tau \in V_\beta .\ snd(x) \Vdash \varphi\ [fst(x), \tau, y_1, \ldots, y_n]\}\urcorner$$

That makes $17 + 15 = 32$ up to now.

Forcing

- 2 instances for definitions by well-founded recursion (check names and forcing for atomic formulas).
- Replacement through $x \mapsto \check{x}$.
- Definition of $\dot{G}$.
- Name of choice functions in $M[G]$.
- 2 instances for the slalom needed for cardinal preservation by ccc forcing.

No strength to apply the Fundamental Theorems

Application of the Forcing theorems do not require any extra Replacement instances on M.

No strength to apply the Fundamental Theorems

Application of the Forcing theorems do not require any extra Replacement instances on M.

That is, we obtain

$$M[G] \models \varphi \iff \exists p \in G.\, p \Vdash \varphi \tag{1}$$

uniformly on φ

No strength to apply the Fundamental Theorems

Application of the Forcing theorems do not require any extra Replacement instances on M.

That is, we obtain

$$M[G] \models \varphi \iff \exists p \in G.\, p \Vdash \varphi \tag{1}$$

uniformly on φ

Dependence of Separation on φ

The instances of Separation required for (1) grows recursively along the structure of φ.

Resolution: Backwards proof

A **rule**:

$$[\![P(0)\ ;\ \bigwedge n.\ (P(n)\implies P(n+1))]\!]\implies \forall n\in \mathsf{nat}.\ P(n) \tag{2}$$

Resolution: Backwards proof

A **rule**:

$$[\![P(0)\ ;\ \bigwedge n.\ (P(n) \implies P(n+1))]\!] \implies \forall n \in \mathsf{nat}.\ P(n) \tag{2}$$

Proof state

1. $\forall n \in \mathsf{nat}.\ n^2 \geq 0.$

Resolution: Backwards proof

A **rule**:

$$[\![P(0)\ ;\ \bigwedge n.\ (P(n) \implies P(n+1))]\!] \implies \forall n \in \mathsf{nat}.\ P(n) \qquad (2)$$

Proof state

1. $\forall n \in \mathsf{nat}.\ n^2 \geq 0.$

After "applying" the rule (2) (`apply rule (2)`):

Proof state

1. $0^2 \geq 0.$
2. $\bigwedge n.\quad n^2 \geq 0 \implies (n+1)^2 \geq 0.$

Procedural proof of DC

```
theorem pointed_DC  : "(∀x∈A. ∃y∈A. ⟨x,y⟩∈ R) ⟹
                       ∀a∈A. (∃f ∈ nat→A. f`0 = a ∧ (∀n ∈ nat. ⟨f`n,f`succ(n)⟩∈R))"
  apply (rule)
  apply (insert AC_func_Pow)
  apply (drule allI)
  apply (drule_tac x="A" in spec)
  apply (drule_tac P="λf .∀x∈Pow(A) - {0}. f ` x ∈ x"
              and A="Pow(A)-{0}→ A"
              and Q=" ∃f∈nat→ A. f ` 0 = a ∧ (∀n∈nat. ⟨f ` n, f ` succ(n)⟩ ∈ R)" in bexE
  prefer 2 apply (assumption)
  apply (rename_tac s)
  apply (rule_tac x="λn∈nat. dc_witness(n,A,a,s,R)" in bexI)
   prefer 2  apply (blast intro:witness_funtype)
  apply (rule  conjI, simp)
  apply (rule ballI, rename_tac m)
  apply (subst beta, simp)+
  apply (rule witness_related, auto)
  done
```

Procedural proof of DC

```
theorem pointed_DC  : "(∀x∈A. ∃y∈A. ⟨x,y⟩∈ R) ⟹
                       ∀a∈A. (∃f ∈ nat→A. f`0 = a ∧ (∀n ∈ nat. ⟨f`n,f`succ(n)⟩∈R))"
  apply (rule)
  apply (insert AC_func_Pow)
  apply (drule allI)
  apply (drule_tac x="A" in spec)
  apply (drule_tac P="λf .∀x∈Pow(A) - {0}. f ` x ∈ x"
              and A="Pow(A)-{0}→ A"
              and Q=" ∃f∈nat→ A. f ` 0 = a ∧ (∀n∈nat. ⟨f ` n, f ` succ(n)⟩ ∈ R)" in bexE
  prefer 2 apply (assumption)
  apply (rename_tac s)
  apply (rule_tac x="λn∈nat. dc_witness(n,A,a,s,R)" in bexI)
   prefer 2  apply (blast intro:witness_funtype)
  apply (rule  conjI, simp)
  apply (rule ballI, rename_tac m)
  apply (subst beta, simp)+
  apply (rule witness_related, auto)
  done
```

Not easy to read.

Procedural proof of DC

```
theorem pointed_DC  : "(∀x∈A. ∃y∈A. ⟨x,y⟩∈ R) ⟹
                       ∀a∈A. (∃f ∈ nat→A. f`0 = a ∧ (∀n ∈ nat. ⟨f`n,f`succ(n)⟩∈R))"
  apply (rule)
  apply (insert AC_func_Pow)
  apply (drule allI)
  apply (drule_tac x="A" in spec)
  apply (drule_tac P="λf .∀x∈Pow(A) - {0}. f ` x ∈ x"
                and A="Pow(A)-{0}→ A"
                and Q=" ∃f∈nat→ A. f ` 0 = a ∧ (∀n∈nat. ⟨f ` n, f ` succ(n)⟩ ∈ R)" in bexE
  prefer 2 apply (assumption)
  apply (rename_tac s)
  apply (rule_tac x="λn∈nat. dc_witness(n,A,a,s,R)" in bexI)
   prefer 2  apply (blast intro:witness_funtype)
  apply (rule  conjI, simp)
  apply (rule ballI, rename_tac m)
  apply (subst beta, simp)+
  apply (rule witness_related, auto)
  done
```

Not easy to read.

Goal

To provide a proof document, which can hopefully be read by mathematicians/set-theorists.

Isar: Forward proofs

lemma : $\forall x \in \{4, 5, 6\}.\ 0 < x$

Proof state

1 $\forall x \in \{4, 5, 6\}.\ 0 < x$

.

Isar: Forward proofs

lemma : $\forall x \in \{4, 5, 6\}.\ 0 < x$
proof

Proof state

1 $\bigwedge x.\ x \in \{4, 5, 6\} \implies 0 < x$

.

Isar: Forward proofs

lemma : $\forall x \in \{4,5,6\}.\ 0 < x$
proof
  fix x
  assume $x \in \{4,5,6\}$

Proof state

1 $\bigwedge x.\ x \in \{4,5,6\} \implies 0 < x$

.

Isar: Forward proofs

lemma : $\forall x \in \{4, 5, 6\}.\ 0 < x$
proof
  fix x
  assume $x \in \{4, 5, 6\}$
  then
  show $0 < x$

Proof state

1 $0 < x$

.

Isar: Forward proofs

lemma : $\forall x \in \{4, 5, 6\}.\ 0 < x$
proof
  fix x
  assume $x \in \{4, 5, 6\}$
  then
  show $0 < x$
    by auto

Proof state

No subgoals!

.

Isar: Forward proofs

lemma : $\forall x \in \{4, 5, 6\}.\ 0 < x$
proof
  fix x
  assume $x \in \{4, 5, 6\}$
  then
  show $0 < x$
    by auto
qed

Proof state

theorem $\forall x \in \{4, 5, 6\}.\ 0 < x$

.

Isar: Forward proofs

lemma : $\forall x \in \{4,5,6\}.\ 0 < x$
proof
  fix x
  assume $x \in \{4,5,6\}$
  then
  show $0 < x$
    by auto
qed

Proof state

theorem $\forall x \in \{4,5,6\}.\ 0 < x$

We now show some cherry-picked piece of the formalized math, and compare it with Kunen [2011].

Theorem IV.2.27 *Let M be a ctm for ZF, let $\mathbb{P} \in M$ be a forcing poset, and let G be $\mathbb{P}$-generic over M. Then $M[G] \models ZF$. Furthermore, $M[G] \models ZFC$ if $M \models ZFC$.*

For Power Set (similarly to Union above), it is sufficient to prove that whenever $a \in M[G]$, there is a $b \in M[G]$ such that $\mathcal{P}(a) \cap M[G] \subseteq b$. Fix $\tau \in M^{\mathbb{P}}$ such that $\tau_G = a$. Let $Q = (\mathcal{P}(\mathrm{dom}(\tau) \times \mathbb{P}))^M$. This is the set of all names $\vartheta \in M^{\mathcal{P}}$ such that $\mathrm{dom}(\vartheta) \subseteq \mathrm{dom}(\tau)$. Let $\pi = Q \times \{\mathbb{1}\}$ and let $b = \pi_G = \{\vartheta_G : \vartheta \in Q\}$. Now, consider any $c \in \mathcal{P}(a) \cap M[G]$; we need to show that $c \in b$.

```
lemma Pow_inter_MG:
  assumes
    "a∈M[G]"
  shows
    "Pow(a) ∩ M[G] ∈ M[G]"
proof -
  from assms obtain τ where "τ ∈ M" "val(P,G, τ) = a"
    using GenExtD by auto
  let ?Q="Pow(domain(τ)×P) ∩ M"
  from ‹τ∈M›
  have "domain(τ)×P ∈ M" "domain(τ) ∈ M" [2 lines]
  then
  have "?Q ∈ M" [17 lines]
  let ?π="?Q×{one}"
  let ?b="val(P,G,?π)"
  from ‹?Q∈M›
  have "?π∈M"
    using one_in_P P_in_M transitivity
    by (simp flip: setclass_iff)
  then
  have "?b ∈ M[G]"
    using GenExtI by simp
  have "Pow(a) ∩ M[G] ⊆ ?b"
  proof
    fix c
    assume "c ∈ Pow(a) ∩ M[G]"
    then obtain χ where "c∈M[G]" "χ ∈ M" "val(P,G,χ) = c"
      using GenExtD by auto
    let ?ϑ="{⟨σ,p⟩ ∈domain(τ)×P . p ⊩ ·0 ∈ 1· [σ,χ] }"
```

Theorem IV.2.27 *Let M be a ctm for ZF, let $\mathbb{P} \in M$ be a forcing poset, and let G be $\mathbb{P}$-generic over M. Then $M[G] \models ZF$. Furthermore, $M[G] \models ZFC$ if $M \models ZFC$.*

For Power Set (similarly to Union above), it is sufficient to prove that whenever $a \in M[G]$, there is a $b \in M[G]$ such that $\mathcal{P}(a) \cap M[G] \subseteq b$. Fix $\tau \in M^{\mathbb{P}}$ such that $\tau_G = a$. Let $Q = (\mathcal{P}(\mathrm{dom}(\tau) \times \mathbb{P}))^M$. This is the set of all names $\vartheta \in M^{\mathcal{P}}$ such that $\mathrm{dom}(\vartheta) \subseteq \mathrm{dom}(\tau)$. Let $\pi = Q \times \{\mathbb{1}\}$ and let $b = \pi_G = \{\vartheta_G : \vartheta \in Q\}$. Now, consider any $c \in \mathcal{P}(a) \cap M[G]$; we need to show that $c \in b$.

```
lemma Pow_inter_MG:
  assumes
    "a∈M[G]"
  shows
    "Pow(a) ∩ M[G] ∈ M[G]"
proof -
  from assms obtain τ where "τ ∈ M" "val(P,G, τ) = a"
    using GenExtD by auto
  let ?Q="Pow(domain(τ)×P) ∩ M"
  from ‹τ∈M›
  have "domain(τ)×P ∈ M" "domain(τ) ∈ M" [2 lines]
  then
  have "?Q ∈ M" [17 lines]
  let ?π="?Q×{one}"
  let ?b="val(P,G,?π)"
  from ‹?Q∈M›
  have "?π∈M"
    using one_in_P P_in_M transitivity
    by (simp flip: setclass_iff)
  then
  have "?b ∈ M[G]"
    using GenExtI by simp
  have "Pow(a) ∩ M[G] ⊆ ?b"
  proof
    fix c
    assume "c ∈ Pow(a) ∩ M[G]"
    then obtain χ where "c∈M[G]" "χ ∈ M" "val(P,G,χ) = c"
      using GenExtD by auto
    let ?ϑ="{⟨σ,p⟩ ∈domain(τ)×P . p ⊩ ·0 ∈ 1· [σ,χ] }"
```

Theorem IV.2.27 *Let M be a ctm for ZF, let $\mathbb{P} \in M$ be a forcing poset, and let G be $\mathbb{P}$-generic over M. Then $M[G] \models ZF$. Furthermore, $M[G] \models ZFC$ if $M \models ZFC$.*

For Power Set (similarly to Union above), it is sufficient to prove that whenever $a \in M[G]$, there is a $b \in M[G]$ such that $\mathcal{P}(a) \cap M[G] \subseteq b$. Fix $\tau \in M^{\mathbb{P}}$ such that $\tau_G = a$. Let $Q = (\mathcal{P}(\mathrm{dom}(\tau) \times \mathbb{P}))^M$. This is the set of all names $\vartheta \in M^{\mathcal{P}}$ such that $\mathrm{dom}(\vartheta) \subseteq \mathrm{dom}(\tau)$. Let $\pi = Q \times \{\mathbb{1}\}$ and let $b = \pi_G = \{\vartheta_G : \vartheta \in Q\}$. Now, consider any $c \in \mathcal{P}(a) \cap M[G]$; we need to show that $c \in b$.

```
lemma Pow_inter_MG:
  assumes
    "a∈M[G]"
  shows
    "Pow(a) ∩ M[G] ∈ M[G]"
proof -
  from assms obtain τ where "τ ∈ M" "val(P,G, τ) = a"
    using GenExtD by auto
  let ?Q="Pow(domain(τ)×P) ∩ M"
  from ‹τ∈M›
  have "domain(τ)×P ∈ M" "domain(τ) ∈ M" [2 lines]
  then
  have "?Q ∈ M" [17 lines]
  let ?π="?Q×{one}"
  let ?b="val(P,G,?π)"
  from ‹?Q∈M›
  have "?π∈M"
    using one_in_P P_in_M transitivity
    by (simp flip: setclass_iff)
  then
  have "?b ∈ M[G]"
    using GenExtI by simp
  have "Pow(a) ∩ M[G] ⊆ ?b"
  proof
    fix c
    assume "c ∈ Pow(a) ∩ M[G]"
    then obtain χ where "c∈M[G]" "χ ∈ M" "val(P,G,χ) = c"
      using GenExtD by auto
    let ?ϑ="{⟨σ,p⟩ ∈domain(τ)×P . p ⊩ ·0 ∈ 1· [σ,χ] }"
```

Theorem IV.2.27 *Let M be a ctm for ZF, let $\mathbb{P} \in M$ be a forcing poset, and let G be $\mathbb{P}$-generic over M. Then $M[G] \models ZF$. Furthermore, $M[G] \models ZFC$ if $M \models ZFC$.*

For Power Set (similarly to Union above), it is sufficient to prove that whenever $a \in M[G]$, there is a $b \in M[G]$ such that $\mathcal{P}(a) \cap M[G] \subseteq b$. Fix $\tau \in M^{\mathbb{P}}$ such that $\tau_G = a$. Let $Q = (\mathcal{P}(\mathrm{dom}(\tau) \times \mathbb{P}))^M$. This is the set of all names $\vartheta \in M^{\mathcal{P}}$ such that $\mathrm{dom}(\vartheta) \subseteq \mathrm{dom}(\tau)$. Let $\pi = Q \times \{\mathbb{1}\}$ and let $b = \pi_G = \{\vartheta_G : \vartheta \in Q\}$. Now, consider any $c \in \mathcal{P}(a) \cap M[G]$; we need to show that $c \in b$.

```
lemma Pow_inter_MG:
  assumes
    "a∈M[G]"
  shows
    "Pow(a) ∩ M[G] ∈ M[G]"
proof -
  from assms obtain τ where "τ ∈ M" "val(P,G, τ) = a"
    using GenExtD by auto
  let ?Q="Pow(domain(τ)×P) ∩ M"
  from ‹τ∈M›
  have "domain(τ)×P ∈ M" "domain(τ) ∈ M" [2 lines]
  then
  have "?Q ∈ M" [17 lines]
  let ?π="?Q×{one}"
  let ?b="val(P,G,?π)"
  from ‹?Q∈M›
  have "?π∈M"
    using one_in_P P_in_M transitivity
    by (simp flip: setclass_iff)
  then
  have "?b ∈ M[G]"
    using GenExtI by simp
  have "Pow(a) ∩ M[G] ⊆ ?b"
  proof
    fix c
    assume "c ∈ Pow(a) ∩ M[G]"
    then obtain χ where "c∈M[G]" "χ ∈ M" "val(P,G,χ) = c"
      using GenExtD by auto
    let ?ϑ="{⟨σ,p⟩ ∈domain(τ)×P . p ⊩ ·0 ∈ 1· [σ,χ] }"
```

Theorem IV.2.27 *Let M be a ctm for ZF, let $\mathbb{P} \in M$ be a forcing poset, and let G be $\mathbb{P}$-generic over M. Then $M[G] \models ZF$. Furthermore, $M[G] \models ZFC$ if $M \models ZFC$.*

For Power Set (similarly to Union above), it is sufficient to prove that whenever $a \in M[G]$, there is a $b \in M[G]$ such that $\mathcal{P}(a) \cap M[G] \subseteq b$. Fix $\tau \in M^{\mathbb{P}}$ such that $\tau_G = a$. Let $Q = (\mathcal{P}(\mathrm{dom}(\tau) \times \mathbb{P}))^M$. This is the set of all names $\vartheta \in M^{\mathcal{P}}$ such that $\mathrm{dom}(\vartheta) \subseteq \mathrm{dom}(\tau)$. Let $\pi = Q \times \{\mathbb{1}\}$ and let $b = \pi_G = \{\vartheta_G : \vartheta \in Q\}$. Now, consider any $c \in \mathcal{P}(a) \cap M[G]$; we need to show that $c \in b$.

```
lemma Pow_inter_MG:
  assumes
    "a∈M[G]"
  shows
    "Pow(a) ∩ M[G] ∈ M[G]"
proof -
  from assms obtain τ where "τ ∈ M" "val(P,G, τ) = a"
    using GenExtD by auto
  let ?Q="Pow(domain(τ)×P) ∩ M"
  from ‹τ∈M›
  have "domain(τ)×P ∈ M" "domain(τ) ∈ M" [2 lines]
  then
  have "?Q ∈ M" [17 lines]
  let ?π="?Q×{one}"
  let ?b="val(P,G,?π)"
  from ‹?Q∈M›
  have "?π∈M"
    using one_in_P P_in_M transitivity
    by (simp flip: setclass_iff)
  then
  have "?b ∈ M[G]"
    using GenExtI by simp
  have "Pow(a) ∩ M[G] ⊆ ?b"
  proof
    fix c
    assume "c ∈ Pow(a) ∩ M[G]"
    then obtain χ where "c∈M[G]" "χ ∈ M" "val(P,G,χ) = c"
      using GenExtD by auto
    let ?ϑ="{⟨σ,p⟩ ∈domain(τ)×P . p ⊩ ·0 ∈ 1· [σ,χ] }"
```

Theorem IV.2.27 *Let M be a ctm for ZF, let $\mathbb{P} \in M$ be a forcing poset, and let G be $\mathbb{P}$-generic over M. Then $M[G] \models ZF$. Furthermore, $M[G] \models ZFC$ if $M \models ZFC$.*

For Power Set (similarly to Union above), it is sufficient to prove that whenever $a \in M[G]$, there is a $b \in M[G]$ such that $\mathcal{P}(a) \cap M[G] \subseteq b$. Fix $\tau \in M^{\mathbb{P}}$ such that $\tau_G = a$. Let $Q = (\mathcal{P}(\mathrm{dom}(\tau) \times \mathbb{P}))^M$. This is the set of all names $\vartheta \in M^{\mathcal{P}}$ such that $\mathrm{dom}(\vartheta) \subseteq \mathrm{dom}(\tau)$. Let $\pi = Q \times \{\mathbb{1}\}$ and let $b = \pi_G = \{\vartheta_G : \vartheta \in Q\}$. Now, consider any $c \in \mathcal{P}(a) \cap M[G]$; we need to show that $c \in b$.

```
lemma Pow_inter_MG:
  assumes
    "a∈M[G]"
  shows
    "Pow(a) ∩ M[G] ∈ M[G]"
proof -
  from assms obtain τ where "τ ∈ M" "val(P,G, τ) = a"
    using GenExtD by auto
  let ?Q="Pow(domain(τ)×P) ∩ M"
  from ‹τ∈M›
  have "domain(τ)×P ∈ M" "domain(τ) ∈ M" [2 lines]
  then
  have "?Q ∈ M" [17 lines]
  let ?π="?Q×{one}"
  let ?b="val(P,G,?π)"
  from ‹?Q∈M›
  have "?π∈M"
    using one_in_P P_in_M transitivity
    by (simp flip: setclass_iff)
  then
  have "?b ∈ M[G]"
    using GenExtI by simp
  have "Pow(a) ∩ M[G] ⊆ ?b"
  proof
    fix c
    assume "c ∈ Pow(a) ∩ M[G]"
    then obtain χ where "c∈M[G]" "χ ∈ M" "val(P,G,χ) = c"
      using GenExtD by auto
    let ?ϑ="{⟨σ,p⟩ ∈domain(τ)×P . p ⊩ ·0 ∈ 1· [σ,χ] }"
```

Theorem IV.2.27 *Let M be a ctm for ZF, let $\mathbb{P} \in M$ be a forcing poset, and let G be $\mathbb{P}$-generic over M. Then $M[G] \models ZF$. Furthermore, $M[G] \models ZFC$ if $M \models ZFC$.*

For Power Set (similarly to Union above), it is sufficient to prove that whenever $a \in M[G]$, there is a $b \in M[G]$ such that $\mathcal{P}(a) \cap M[G] \subseteq b$. Fix $\tau \in M^{\mathbb{P}}$ such that $\tau_G = a$. Let $Q = (\mathcal{P}(\mathrm{dom}(\tau) \times \mathbb{P}))^M$. This is the set of all names $\vartheta \in M^{\mathcal{P}}$ such that $\mathrm{dom}(\vartheta) \subseteq \mathrm{dom}(\tau)$. Let $\pi = Q \times \{\mathbb{1}\}$ and let $b = \pi_G = \{\vartheta_G : \vartheta \in Q\}$. Now, consider any $c \in \mathcal{P}(a) \cap M[G]$; we need to show that $c \in b$.

```
lemma Pow_inter_MG:
  assumes
    "a∈M[G]"
  shows
    "Pow(a) ∩ M[G] ∈ M[G]"
proof -
  from assms obtain τ where "τ ∈ M" "val(P,G, τ) = a"
    using GenExtD by auto
  let ?Q="Pow(domain(τ)×P) ∩ M"
  from ‹τ∈M›
  have "domain(τ)×P ∈ M" "domain(τ) ∈ M" [2 lines]
  then
  have "?Q ∈ M" [17 lines]
  let ?π="?Q×{one}"
  let ?b="val(P,G,?π)"
  from ‹?Q∈M›
  have "?π∈M"
    using one_in_P P_in_M transitivity
    by (simp flip: setclass_iff)
  then
  have "?b ∈ M[G]"
    using GenExtI by simp
  have "Pow(a) ∩ M[G] ⊆ ?b"
  proof
    fix c
    assume "c ∈ Pow(a) ∩ M[G]"
    then obtain χ where "c∈M[G]" "χ ∈ M" "val(P,G,χ) = c"
      using GenExtD by auto
    let ?ϑ="{⟨σ,p⟩ ∈domain(τ)×P . p ⊩ ·0 ∈ 1· [σ,χ] }"
```

$\{\vartheta_G : \vartheta \in Q\}$. Now, consider any $c \in \mathcal{P}(a) \cap M[G]$; we need to show that $c \in b$. Fix $\varkappa \in M^{\mathbb{P}}$ such that $\varkappa_G = c$, and let $\vartheta = \{\langle \sigma, p\rangle : \sigma \in \mathrm{dom}(\tau) \wedge p \Vdash \sigma \in \varkappa\}$; $\vartheta \in M$ by the Definability Lemma. Since $\vartheta \in Q$, we are done if we can show that $\vartheta_G = c$. $\vartheta_G \subseteq c$ holds because $\vartheta_G = \{\sigma_G : \exists p \in G\ p \Vdash \sigma \in \varkappa\}$ and all these σ_G lie in $\varkappa_G = c$ by the definition of $\Vdash$. To prove $c \subseteq \vartheta_G$: since $c \subseteq a = \tau_G$, every element of c is of the form σ_G for some $\sigma \in \mathrm{dom}(\tau)$. Since $\sigma_G \in c = \varkappa_G$, apply the Truth Lemma and fix $p \in G$ such that $p \Vdash \sigma \in \varkappa$; then $\langle \sigma, p\rangle \in \vartheta$, so $\sigma_G \in \vartheta_G$.

```
    using GenExtI by simp
  have "Pow(a) ∩ M[G] ⊆ ?b"
  proof
    fix c
    assume "c ∈ Pow(a) ∩ M[G]"
    then obtain χ where "c∈M[G]" "χ ∈ M" "val(P,G,χ) = c"
      using GenExtD by auto
    let ?ϑ="{⟨σ,p⟩ ∈domain(τ)×P . p ⊩ ·0 ∈ 1· [σ,χ] }"
    have "arity(forces(Member(0,1))) = 6" [1 lines]
    with ‹domain(τ) ∈ M› ‹χ ∈ M›
    have "?ϑ ∈ M"
      using P_in_M one_in_M leq_in_M sats_fst_snd_in_M
      by simp
    then
    have "?ϑ ∈ ?Q" by auto
    then
    have "val(P,G,?ϑ) ∈ ?b"
      using one_in_G one_in_P generic val_of_elem [of ?ϑ one ?π G]
      by auto
    have "val(P,G,?ϑ) = c"
    proof(intro equalityI subsetI) [24 lines]
    next
      fix x
      assume "x ∈ c"
      with ‹c ∈ Pow(a) ∩ M[G]›
      have "x ∈ a" "c∈M[G]" "x∈M[G]"
        using transitivity_MG by auto
      with ‹val(P,G, τ) = a›
      obtain σ where "σ∈domain(τ)" "val(P,G,σ) = x"
```

$\{\vartheta_G : \vartheta \in Q\}$. Now, consider any $c \in \mathcal{P}(a) \cap M[G]$; we need to show that $c \in b$. Fix $\varkappa \in M^{\mathbb{P}}$ such that $\varkappa_G = c$, and let $\vartheta = \{\langle \sigma, p\rangle : \sigma \in \mathrm{dom}(\tau) \wedge p \Vdash \sigma \in \varkappa\}$; $\vartheta \in M$ by the Definability Lemma. Since $\vartheta \in Q$, we are done if we can show that $\vartheta_G = c$. $\vartheta_G \subseteq c$ holds because $\vartheta_G = \{\sigma_G : \exists p \in G\ p \Vdash \sigma \in \varkappa\}$ and all these σ_G lie in $\varkappa_G = c$ by the definition of $\Vdash$. To prove $c \subseteq \vartheta_G$: since $c \subseteq a = \tau_G$, every element of c is of the form σ_G for some $\sigma \in \mathrm{dom}(\tau)$. Since $\sigma_G \in c = \varkappa_G$, apply the Truth Lemma and fix $p \in G$ such that $p \Vdash \sigma \in \varkappa$; then $\langle \sigma, p\rangle \in \vartheta$, so $\sigma_G \in \vartheta_G$.

```
    using GenExtI by simp
  have "Pow(a) ∩ M[G] ⊆ ?b"
  proof
    fix c
    assume "c ∈ Pow(a) ∩ M[G]"
    then obtain χ where "c∈M[G]" "χ ∈ M" "val(P,G,χ) = c"
      using GenExtD by auto
    let ?ϑ="{⟨σ,p⟩ ∈domain(τ)×P . p ⊩ ·0 ∈ 1· [σ,χ] }"
    have "arity(forces(Member(0,1))) = 6" [1 lines]
    with ‹domain(τ) ∈ M› ‹χ ∈ M›
    have "?ϑ ∈ M"
      using P_in_M one_in_M leq_in_M sats_fst_snd_in_M
      by simp
    then
    have "?ϑ ∈ ?Q" by auto
    then
    have "val(P,G,?ϑ) ∈ ?b"
      using one_in_G one_in_P generic val_of_elem [of ?ϑ one ?π G]
      by auto
    have "val(P,G,?ϑ) = c"
    proof(intro equalityI subsetI) [24 lines]
    next
      fix x
      assume "x ∈ c"
      with ‹c ∈ Pow(a) ∩ M[G]›
      have "x ∈ a" "c∈M[G]" "x∈M[G]"
        using transitivity_MG by auto
      with ‹val(P,G, τ) = a›
      obtain σ where "σ∈domain(τ)" "val(P,G,σ) = x"
```

$\{\vartheta_G : \vartheta \in Q\}$. Now, consider any $c \in \mathcal{P}(a) \cap M[G]$; we need to show that $c \in b$. Fix $\varkappa \in M^{\mathbb{P}}$ such that $\varkappa_G = c$, and let $\vartheta = \{\langle \sigma, p\rangle : \sigma \in \mathrm{dom}(\tau) \wedge p \Vdash \sigma \in \varkappa\}$; $\vartheta \in M$ by the Definability Lemma. Since $\vartheta \in Q$, we are done if we can show that $\vartheta_G = c$. $\vartheta_G \subseteq c$ holds because $\vartheta_G = \{\sigma_G : \exists p \in G\ p \Vdash \sigma \in \varkappa\}$ and all these σ_G lie in $\varkappa_G = c$ by the definition of $\Vdash$. To prove $c \subseteq \vartheta_G$: since $c \subseteq a = \tau_G$, every element of c is of the form σ_G for some $\sigma \in \mathrm{dom}(\tau)$. Since $\sigma_G \in c = \varkappa_G$, apply the Truth Lemma and fix $p \in G$ such that $p \Vdash \sigma \in \varkappa$; then $\langle \sigma, p\rangle \in \vartheta$, so $\sigma_G \in \vartheta_G$.

```
    using GenExtI by simp
  have "Pow(a) ∩ M[G] ⊆ ?b"
  proof
    fix c
    assume "c ∈ Pow(a) ∩ M[G]"
    then obtain χ where "c∈M[G]" "χ ∈ M" "val(P,G,χ) = c"
      using GenExtD by auto
    let ?ϑ="{⟨σ,p⟩ ∈domain(τ)×P . p ⊩ ·0 ∈ 1· [σ,χ] }"
    have "arity(forces(Member(0,1))) = 6" [1 lines]
    with ‹domain(τ) ∈ M› ‹χ ∈ M›
    have "?ϑ ∈ M"
      using P_in_M one_in_M leq_in_M sats_fst_snd_in_M
      by simp
    then
    have "?ϑ ∈ ?Q" by auto
    then
    have "val(P,G,?ϑ) ∈ ?b"
      using one_in_G one_in_P generic val_of_elem [of ?ϑ one ?π G]
      by auto
    have "val(P,G,?ϑ) = c"
    proof(intro equalityI subsetI) [24 lines]
    next
      fix x
      assume "x ∈ c"
      with ‹c ∈ Pow(a) ∩ M[G]›
      have "x ∈ a" "c∈M[G]" "x∈M[G]"
        using transitivity_MG by auto
      with ‹val(P,G, τ) = a›
      obtain σ where "σ∈domain(τ)" "val(P,G,σ) = x"
```

$\{\vartheta_G : \vartheta \in Q\}$. Now, consider any $c \in \mathcal{P}(a) \cap M[G]$; we need to show that $c \in b$. Fix $\varkappa \in M^{\mathbb{P}}$ such that $\varkappa_G = c$, and let $\vartheta = \{\langle \sigma, p\rangle : \sigma \in \mathrm{dom}(\tau) \wedge p \Vdash \sigma \in \varkappa\}$; $\vartheta \in M$ by the Definability Lemma. Since $\vartheta \in Q$, we are done if we can show that $\vartheta_G = c$. $\vartheta_G \subseteq c$ holds because $\vartheta_G = \{\sigma_G : \exists p \in G\ p \Vdash \sigma \in \varkappa\}$ and all these σ_G lie in $\varkappa_G = c$ by the definition of $\Vdash$. To prove $c \subseteq \vartheta_G$: since $c \subseteq a = \tau_G$, every element of c is of the form σ_G for some $\sigma \in \mathrm{dom}(\tau)$. Since $\sigma_G \in c = \varkappa_G$, apply the Truth Lemma and fix $p \in G$ such that $p \Vdash \sigma \in \varkappa$; then $\langle \sigma, p\rangle \in \vartheta$, so $\sigma_G \in \vartheta_G$.

```
    using GenExtI by simp
  have "Pow(a) ∩ M[G] ⊆ ?b"
  proof
    fix c
    assume "c ∈ Pow(a) ∩ M[G]"
    then obtain χ where "c∈M[G]" "χ ∈ M" "val(P,G,χ) = c"
      using GenExtD by auto
    let ?ϑ="{⟨σ,p⟩ ∈domain(τ)×P . p ⊩ ·0 ∈ 1· [σ,χ] }"
    have "arity(forces(Member(0,1))) = 6" [1 lines]
    with ‹domain(τ) ∈ M› ‹χ ∈ M›
    have "?ϑ ∈ M"
      using P_in_M one_in_M leq_in_M sats_fst_snd_in_M
      by simp
    then
    have "?ϑ ∈ ?Q" by auto
    then
    have "val(P,G,?ϑ) ∈ ?b"
      using one_in_G one_in_P generic val_of_elem [of ?ϑ one ?π G]
      by auto
    have "val(P,G,?ϑ) = c"
    proof(intro equalityI subsetI) [24 lines]
    next
      fix x
      assume "x ∈ c"
      with ‹c ∈ Pow(a) ∩ M[G]›
      have "x ∈ a" "c∈M[G]" "x∈M[G]"
        using transitivity_MG by auto
      with ‹val(P,G, τ) = a›
      obtain σ where "σ∈domain(τ)" "val(P,G,σ) = x"
```

$\{\vartheta_G : \vartheta \in Q\}$. Now, consider any $c \in \mathcal{P}(a) \cap M[G]$; we need to show that $c \in b$. Fix $\varkappa \in M^{\mathbb{P}}$ such that $\varkappa_G = c$, and let $\vartheta = \{\langle \sigma, p\rangle : \sigma \in \mathrm{dom}(\tau) \wedge p \Vdash \sigma \in \varkappa\}$; $\vartheta \in M$ by the Definability Lemma. Since $\vartheta \in Q$, we are done if we can show that $\vartheta_G = c$. $\vartheta_G \subseteq c$ holds because $\vartheta_G = \{\sigma_G : \exists p \in G\ p \Vdash \sigma \in \varkappa\}$ and all these σ_G lie in $\varkappa_G = c$ by the definition of $\Vdash$. To prove $c \subseteq \vartheta_G$: since $c \subseteq a = \tau_G$, every element of c is of the form σ_G for some $\sigma \in \mathrm{dom}(\tau)$. Since $\sigma_G \in c = \varkappa_G$, apply the Truth Lemma and fix $p \in G$ such that $p \Vdash \sigma \in \varkappa$; then $\langle \sigma, p\rangle \in \vartheta$, so $\sigma_G \in \vartheta_G$.

```
    using GenExtI by simp
  have "Pow(a) ∩ M[G] ⊆ ?b"
  proof
    fix c
    assume "c ∈ Pow(a) ∩ M[G]"
    then obtain χ where "c∈M[G]" "χ ∈ M" "val(P,G,χ) = c"
      using GenExtD by auto
    let ?ϑ="{⟨σ,p⟩ ∈domain(τ)×P . p ⊩ ·0 ∈ 1· [σ,χ] }"
    have "arity(forces(Member(0,1))) = 6" [1 lines]
    with ‹domain(τ) ∈ M› ‹χ ∈ M›
    have "?ϑ ∈ M"
      using P_in_M one_in_M leq_in_M sats_fst_snd_in_M
      by simp
    then
    have "?ϑ ∈ ?Q" by auto
    then
    have "val(P,G,?ϑ) ∈ ?b"
      using one_in_G one_in_P generic val_of_elem [of ?ϑ one ?π G]
      by auto
    have "val(P,G,?ϑ) = c"
    proof(intro equalityI subsetI) [24 lines]
    next
      fix x
      assume "x ∈ c"
      with ‹c ∈ Pow(a) ∩ M[G]›
      have "x ∈ a" "c∈M[G]" "x∈M[G]"
        using transitivity_MG by auto
      with ‹val(P,G, τ) = a›
      obtain σ where "σ∈domain(τ)" "val(P,G,σ) = x"
```

Other approaches to set theory and forcing

- Lean: Full formalization of the Boolean-valued approach to forcing and the independence of CH [Han and van Doorn, 2020].

Other approaches to set theory and forcing

- Lean: Full formalization of the Boolean-valued approach to forcing and the independence of CH [Han and van Doorn, 2020].
- Set theory over Isabelle/HOL: `ZFC_in_HOL` [Paulson, 2019]

Other approaches to set theory and forcing

- Lean: Full formalization of the Boolean-valued approach to forcing and the independence of CH [Han and van Doorn, 2020].
- Set theory over Isabelle/HOL: `ZFC_in_HOL` [Paulson, 2019]

A word on consistency strength

Isabelle/ZF	equiconsistent with ZF (?) [Paulson, 1989].
`ZFC_in_HOL`	approximately ZF + one inaccessible.
Lean (CiC)	ZF + ω inaccessibles [Carneiro, 2019].

Some mathporn (I)

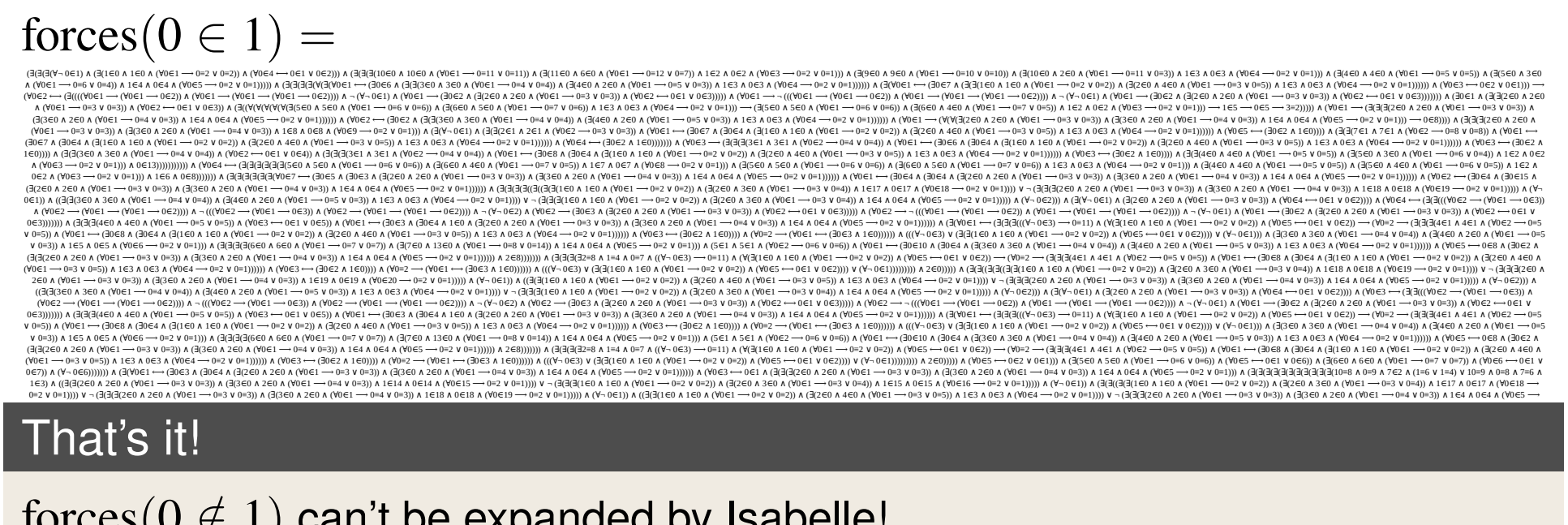
$$\mathrm{forces}(0 \in 1) =$$

Some mathporn (I)

$$\mathrm{forces}(0 \in 1) =$$

Some mathporn (I)

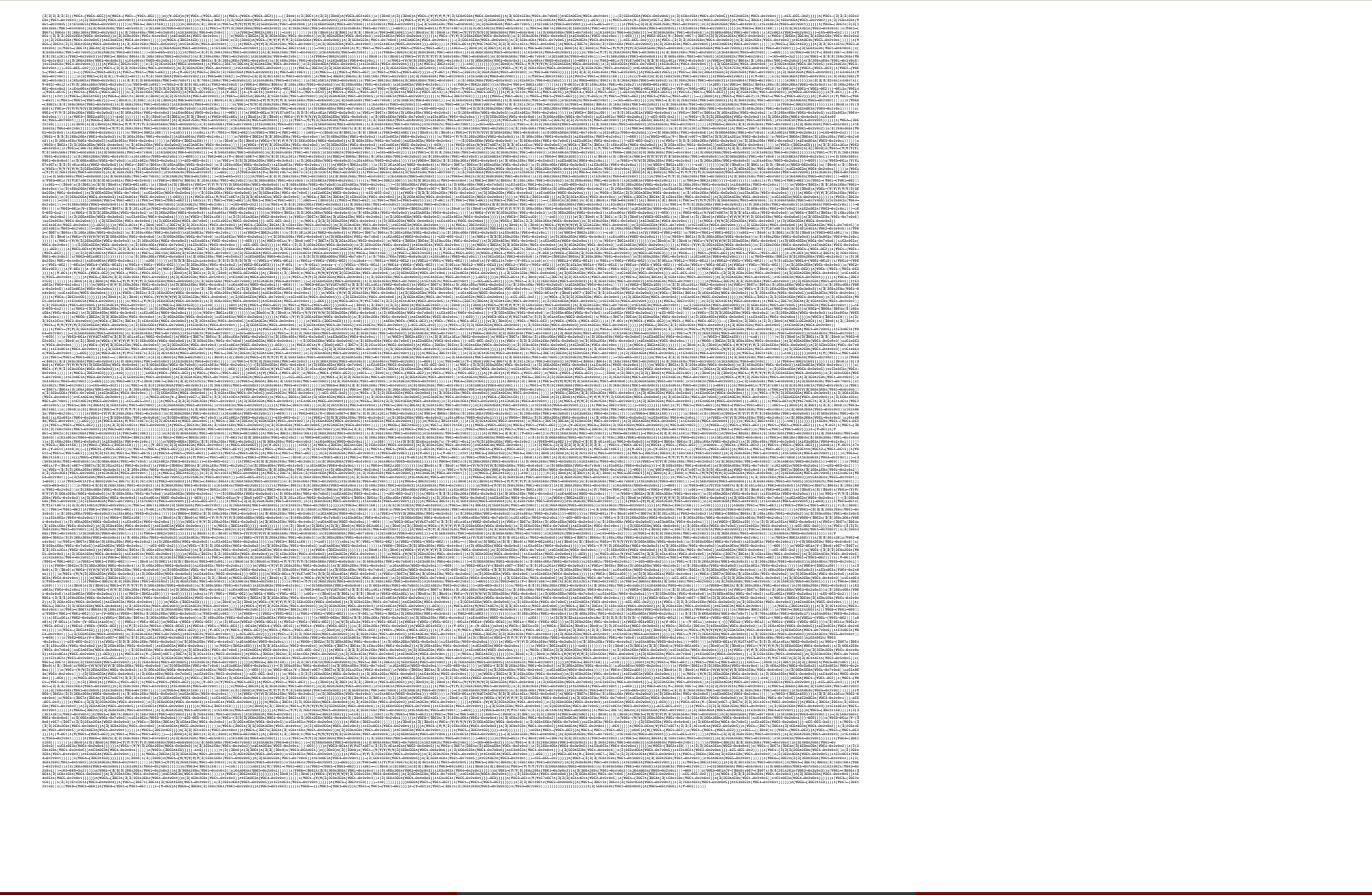
$\mathrm{forces}(0 \in 1) =$



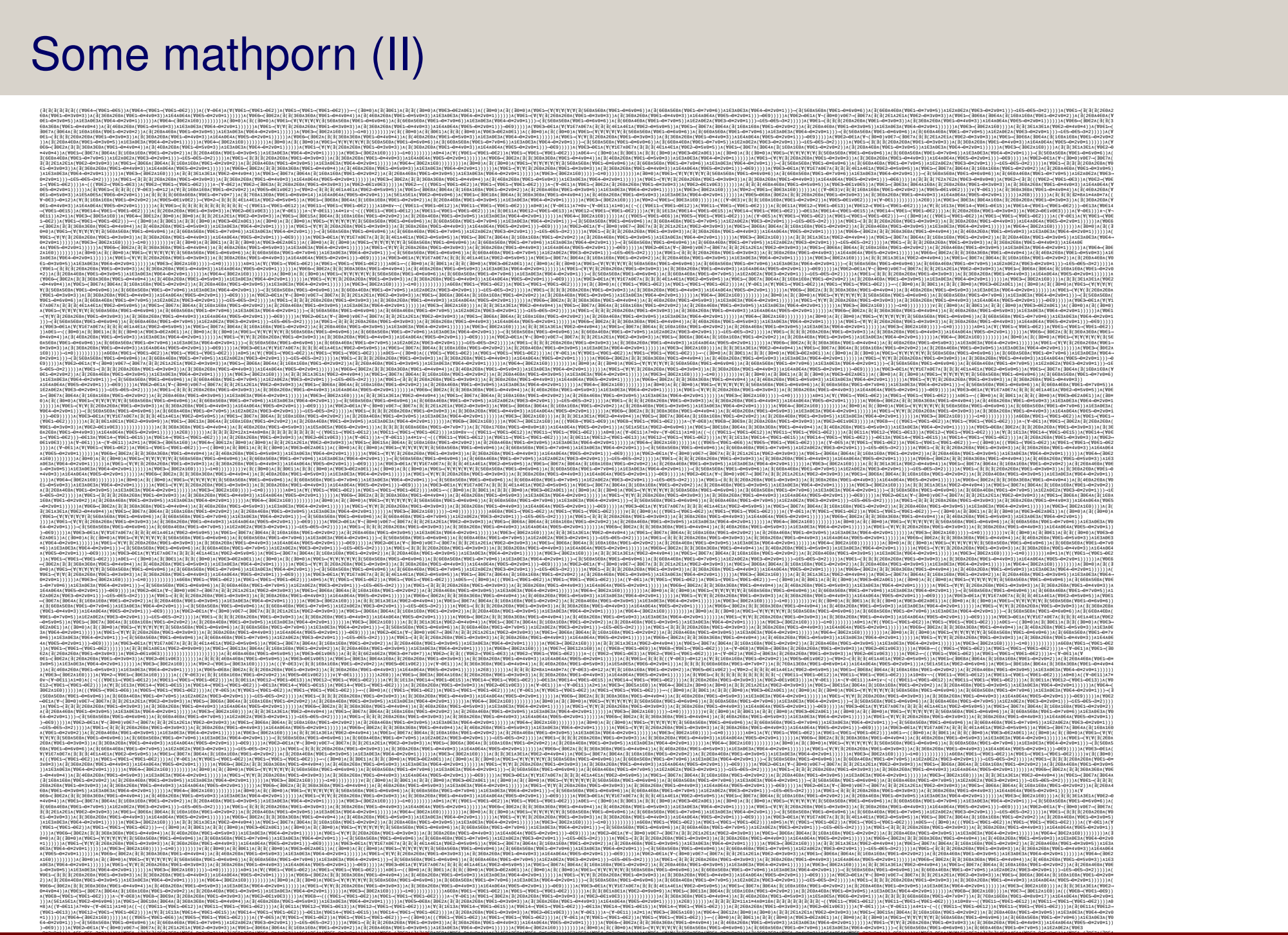
That's it!

$\mathrm{forces}(0 \notin 1)$ can't be expanded by Isabelle!

Some mathporn (II)



Some mathporn (II)

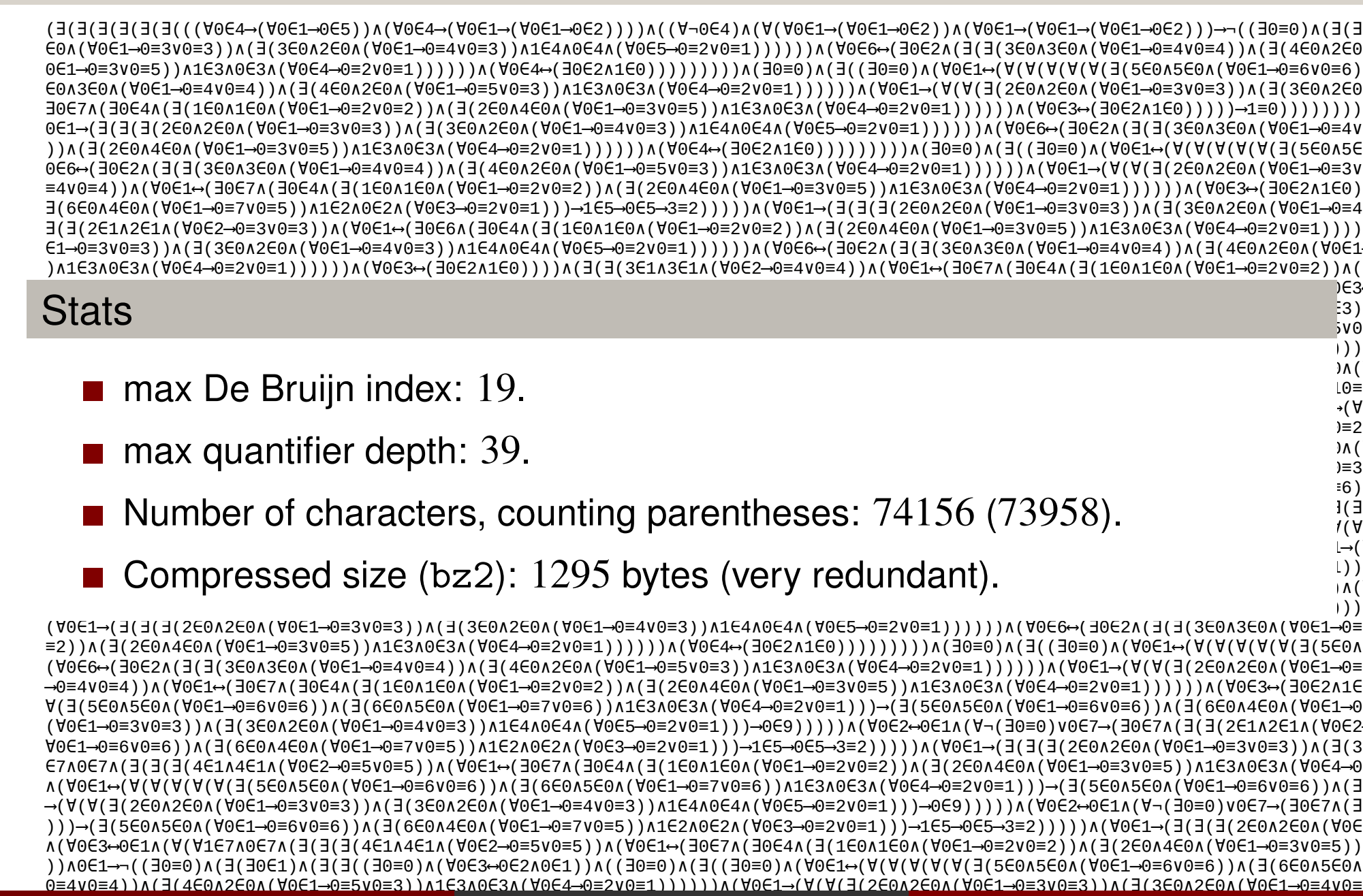


Some mathpori (II)

Some mathporn (II)

Some mathporn (II): CH

Some mathporn (II): CH



Stats

- max De Bruijn index: 19.
- max quantifier depth: 39.
- Number of characters, counting parentheses: 74156 (73958).
- Compressed size (bz2): 1295 bytes (very redundant).

How to read our work

Some links

- Site of the project: https://cs.famaf.unc.edu.ar/~pedro/forcing/
- Code repository: https://bitbucket.org/miguelpagano/forcing/src/master/src/

How to read our work

Some links

- Site of the project: `https://cs.famaf.unc.edu.ar/~pedro/forcing/`
- Code repository: `https://bitbucket.org/miguelpagano/forcing/src/master/src/`

Where to start

- Check out the outline which spares you from proofs. The last section, Main Definitions. . . , contains a summary.
- That section corresponds to the theory `Definitions_Main`, cross-linked to navigate the results.

Future work

On formalization

- Construction of ctms:
  - of ZFC from an inaccessible;
  - of the “forcing-abling” fragment, unconditionally.
- Collapse to obtain CH.
- Class forcing, using some definable $\mathbb{P} \subseteq M$ instead of $\mathbb{P} \in M$.
- Connect this development to Isabelle/HOL.

Future work

On formalization

- Construction of ctms:
  - of ZFC from an inaccessible;
  - of the "forcing-abling" fragment, unconditionally.
- Collapse to obtain CH.
- Class forcing, using some definable $\mathbb{P} \subseteq M$ instead of $\mathbb{P} \in M$.
- Connect this development to Isabelle/HOL.

More meta-theoretic mining

- Analyzing Separation instances in detail.
- Postpone uses of Powerset in ZF-Constructible.

```
— ‹Kunen IV.3.5›
lemma ccc_fun_approximation_lemma:
  notes le_trans[trans]
  assumes "cccᴹ(P,leq)" "A∈M" "B∈M" "f∈M[G]" "f : A → B"
  shows
    "∃F∈M. F : A → Pow(B) ∧ (∀a∈A. f`a ∈ F`a ∧ |F`a|ᴹ ≤ ω)"
proof -
  from ‹f∈M[G]›
  obtain f_dot where "f = val(P,G,f_dot)" "f_dot∈M" using GenE
  with assms
  obtain p where "p ⊩ ·0:1→2· [f_dot, Aᵛ, Bᵛ]" "p∈G" "p∈M"
```

Thanks! Obrigado! ¡Gracias!

```
lemma Aleph2_extension_le_conti
  includes G_generic_lemmas
  shows "ℵ₂^M[G] ≤ 2^(↑ℵ₀^M[G],M[G])"
proof -
  have "ℵ₂^M ∈ M[G]" "Ord(ℵ₂^M)"
  moreover from this
  have "ℵ₂^M ≲^M[G] ω →^M[G] 2" [4
  moreover from calculation
  have "ℵ₂^M ≲^M[G] |ω →^M[G] 2|^M[G]
  ultimately
  have "|ℵ₂^M|^M[G] ≤ 2^(↑ℵ₀^M[G],M[G])" [
  then
  show "ℵ₂^M[G] ≤ 2^(↑ℵ₀^M[G],M[G])" [4
qed
```

References I

J. AVIGAD, Foundations, (2021).

M. CARNEIRO, "The Type Theory of Lean", Master's thesis, Carnegie Mellon University (2019).

P. COHEN, The independence of the continuum hypothesis, *Proc. Nat. Acad. Sci. U.S.A.* **50**: 1143–1148 (1963).

G. GONTHIER, Formal proof—the four-color theorem, *Notices Amer. Math. Soc.* **55**: 1382–1393 (2008).

G. GONTHIER, A. ASPERTI, J. AVIGAD, Y. BERTOT, C. COHEN, F. GARILLOT, S. LE ROUX, A. MAHBOUBI, R. O'CONNOR, S.O. BIHA, I. PASCA, L. RIDEAU, A. SOLOVYEV, E. TASSI, L. THÉRY, A machine-checked proof of the odd order theorem, in: Proceedings of the 4th International Conference on Interactive Theorem Proving, ITP'13, Springer-Verlag, Berlin, Heidelberg: 163–179 (2013).

T. HALES, M. ADAMS, G. BAUER, T.D. DANG, J. HARRISON, L.T. HOANG, C. KALISZYK, V. MAGRON, S. MCLAUGHLIN, T.T. NGUYEN, ET AL., A formal proof of the Kepler conjecture, *Forum Math. Pi* **5**: e2, 29 (2017).

J.M. HAN, F. VAN DOORN, A formal proof of the independence of the continuum hypothesis, in: J. Blanchette, C. Hritcu (Eds.), Proceedings of the 9th ACM SIGPLAN International Conference on Certified Programs and Proofs, CPP 2020, New Orleans, LA, USA, January 20-21, 2020, ACM (2020).

K. KUNEN, "Set Theory", Studies in Logic, College Publications (2011), second edition. Revised edition, 2013.

LEAN COMMUNITY, `mathlib`, Webpage, (2021 — accessed September 2021). `https://leanprover-community.github.io/mathlib-overview.html`.

References II

L.C. PAULSON, The foundation of a generic theorem prover, *Journal of Automated Reasoning* **5**: 363–397 (1989).

L.C. PAULSON, The relative consistency of the axiom of choice mechanized using Isabelle/ZF, *LMS Journal of Computation and Mathematics* **6**: 198–248 (2003).

L.C. PAULSON, ALEXANDRIA: Large-scale formal proof for the working mathematician, Webpage, (2017 — accessed September 2018). EC Project: `https://bit.ly/2Nb26ys`.

L.C. PAULSON, Zermelo Fraenkel set theory in higher-order logic, *Archive of Formal Proofs* (2019). `http://isa-afp.org/entries/ZFC_in_HOL.html`, Formal proof development.

L.C. PAULSON, K. GRABCZEWSKI, Mechanizing set theory, *J. Autom. Reasoning* **17**: 291–323 (1996).

T. UNIVALENT FOUNDATIONS PROGRAM, "Homotopy Type Theory: Univalent Foundations of Mathematics", `https://homotopytypetheory.org/book`, Institute for Advanced Study (2013).