The Independence of the Continuum Hypothesis in Lean

Floris van Doorn

University of Paris-Saclay in Orsay

8 September 2023

j.w.w. Jesse Han

Overview

Definition

The continuum hypothesis (CH) states that there is no set whose cardinality is strictly between those of $\mathbb{N}$ and $\mathbb{R}$.

Theorem (Cohen, 1963)

The usual axioms ZFC of set theory can neither prove nor disprove CH.

Overview

Definition

The continuum hypothesis (CH) states that there is no set whose cardinality is strictly between those of $\mathbb{N}$ and $\mathbb{R}$.

Theorem (Cohen, 1963)

The usual axioms ZFC of set theory can neither prove nor disprove CH.

Together with Jesse Han I formalized this in the Flypitch[1] project.

FLYPITCH

formally proving the independence of the continuum hypothesis

[1] **F**orma**lly** **p**roving the **i**ndependence of **t**he **c**ontinuum **h**ypothesis

Brief History

- In 1878 Georg Cantor conjectured that CH is true.
- CH was Hilbert's first problem (1900)
- In 1940 Kurt Gödel proved that ZFC cannot disprove CH using the constructible universe.
- In 1963 Paul Cohen introduces forcing and proves that ZFC cannot prove CH.

Brief History

- In 1878 Georg Cantor conjectured that CH is true.
- CH was Hilbert's first problem (1900)
- In 1940 Kurt Gödel proved that ZFC cannot disprove CH using the constructible universe.
- In 1963 Paul Cohen introduces forcing and proves that ZFC cannot prove CH.

We didn't follow the standard proof:

- We use forcing using Boolean-valued models (Solovay, Scott 1965);
- We prove both parts using forcing.

Logic

How do we show that something cannot be proven?

Logic

How do we show that something cannot be proven?

I will give you a crash course in first-order logic to make sense of this.

Logic

How do we show that something cannot be proven?

I will give you a crash course in first-order logic to make sense of this.

Note that first-order logic is not the theory of Lean, which is a version of dependent type theory.

Language

Before we do first-order logic,[2] we have to fix a language:

```
structure Language where
  Functions : ℕ → Type u
  Relations : ℕ → Type v
```

[2] We will do single sorted logic with a designated binary relation symbol =.

Language

Before we do first-order logic,[2] we have to fix a language:

```
structure Language where
  Functions : ℕ → Type u
  Relations : ℕ → Type v
```

Examples

- The language of groups: $L_{\mathsf{Group}} := \{\cdot, 1, {}^{-1}\}$.
- The language of ordered rings: $L_{\mathsf{ordRing}} := \{+, \cdot, 0, 1, -, \leq\}$.
- The language of modules over a fixed ring R: $L_{R\text{-}\mathsf{Mod}} := \{+, 0, -\} \cup \{c \cdot (-) \mid c \in R\}$
- The language of set theory: $L_{\mathsf{sets}} := \{\in\}$
- You can write languages for any algebraic theory, graphs, planar geometry, ...

[2] We will do single sorted logic with a designated binary relation symbol =.

Terms

Inductive types are used to build recursive data types:

```
inductive ℕ : Type
| 0 : ℕ
| succ : ℕ → ℕ
```

Terms

Inductive types are used to build recursive data types:

```
inductive ℕ : Type
| 0 : ℕ
| succ : ℕ → ℕ
```

Given a language L then the terms in the language with variables from α are either variables, or an n-ary function symbol of L applied to n terms.

```
inductive Term (α : Type _) : Type _
| var : α → Term α
| func : ∀ {n : ℕ} (f : L.Functions n)
  (ts : Fin n → Term α), Term α
```

Terms

Inductive types are used to build recursive data types:

```
inductive ℕ : Type
| 0 : ℕ
| succ : ℕ → ℕ
```

Given a language L then the terms in the language with variables from α are either variables, or an n-ary function symbol of L applied to n terms.

```
inductive Term (α : Type _) : Type _
| var : α → Term α
| func : ∀ {n : ℕ} (f : L.Functions n)
  (ts : Fin n → Term α), Term α
```

Examples

- x and $y \cdot (y \cdot z)$ and $(x \cdot 1^{-1})^{-1} \cdot x$ are terms in L_{Group}.
- All terms in L_{sets} are variables

Formulas

Formulas are now given by

- $\bot$, the false formula
- $t = s$ where t and s are terms
- $R(t_1, \ldots, t_n)$ where R is an n-ary relation symbol and the t_i are terms
- $\varphi \Rightarrow \psi$, $\varphi \Leftrightarrow \psi$, $\varphi \wedge \psi$, $\varphi \vee \psi$ or $\neg\varphi$ where φ and ψ are formula
- $\forall x, \varphi$ and $\exists x, \varphi$ where φ is a formula. Any variable x occurring in φ is *captured* by this universal quantification (just as in $\int f(x)\,dx$).

Variables in a formula not captured by any quantifier are called free variables.

Formulas in Lean

```
inductive BoundedFormula : ℕ → Type _
| rel {n l} (R : L.Relations l)
  (ts : Fin l → L.Term (α ⊕ Fin n)) : BoundedFormula n
| falsum {n} : BoundedFormula n
| equal {n} (t₁ t₂ : L.Term (α ⊕ Fin n)) : BoundedFormula n
| imp {n} (f₁ f₂ : BoundedFormula n) : BoundedFormula n
| all {n} (f : BoundedFormula (n + 1)) : BoundedFormula n

def Formula := L.BoundedFormula α 0
def Sentence := L.Formula Empty
def Theory := Set L.Sentence
```

Theories

A sentence is a formula without free variables and a theory is a set of sentences.

Theories

A sentence is a formula without free variables and a theory is a set of sentences.

Examples

- The theory of groups might contain the axioms:
  - $\forall g,\ g \cdot 1 = g$
  - $\forall g_1\ g_2\ g_3,\ g_1 \cdot (g_2 \cdot g_3) = (g_1 \cdot g_2) \cdot g_3$
  - $\forall g,\ g \cdot g^{-1} = 1.$
- The language ZFC of set theory contains axioms like the following:
  - Empty set: $\exists s,\ \forall x,\ \neg(x \in s)$ $(s = \varnothing)$
  - Pairing: $\forall x\ y,\ \exists s,\ \forall z,\ z \in s \iff z = x \vee z = y$ $(s = \{x, y\})$
  - Power set: $\forall s,\ \exists P,\ \forall t,\ t \in P \iff \underbrace{\forall x,\ x \in t \Rightarrow x \in s}_{t \subseteq s}$ $(P = \mathcal{P}(s))$

$$\vdots$$

Proofs

Given a set of formulas Γ and a formula φ, we define the predicate $\Gamma \vdash \varphi$: φ is provable from assumptions in Γ. If we restrict ourselves to $\bot$, $\Rightarrow$ and $\forall$, the following rules are sufficient to define provability:

- If $\varphi \in \Gamma$ then $\Gamma \vdash \varphi$;
- If $\Gamma \cup \{\varphi\} \vdash \psi$ then $\Gamma \vdash \varphi \Rightarrow \psi$;
- If $\Gamma \vdash \varphi$ and $\Gamma \vdash \varphi \Rightarrow \psi$ then $\Gamma \vdash \psi$;
- If $\Gamma \cup \{\varphi \Rightarrow \bot\} \vdash \bot$ then $\Gamma \vdash \varphi$;
- If $\Gamma \vdash \varphi$ and x does not occur in any formula in Γ, then $\Gamma \vdash \forall x, \varphi$;
- If $\Gamma \vdash \forall x, \varphi$ then $\Gamma \vdash \varphi[t/x]$, where $\varphi[t/x]$ is the formula φ with each occurrence of x replaced by the term t;
- $\Gamma \vdash t = t$ for any term t;
- If $\Gamma \vdash s = t$ and $\Gamma \vdash \varphi[t/x]$ then $\Gamma \vdash \varphi[s/x]$.

Defined relation symbols

In set theory we can define predicates such as

- $s \subseteq t$
- α is an ordinal
- f is a function
- there is a surjection from s onto t (notation: $t \le s$)

We can also define sets and operations on sets, such as $\aleph_0$, the least infinite cardinal and $\mathcal{P}(s)$, the power set of s.

Defined relation symbols

In set theory we can define predicates such as

- $s \subseteq t$
- α is an ordinal
- f is a function
- there is a surjection from s onto t (notation: $t \leq s$)

We can also define sets and operations on sets, such as $\aleph_0$, the least infinite cardinal and $\mathcal{P}(s)$, the power set of s.
Then we can state the continuum hypothesis as

$$\mathsf{CH} := \forall s,\ s \leq \aleph_0 \vee \mathcal{P}(\aleph_0) \leq s$$

So the independence of CH is the statement

$$\mathsf{ZFC} \nvdash \mathsf{CH} \quad \text{and} \quad \mathsf{ZFC} \nvdash \neg\mathsf{CH}.$$

Defined relation symbols

In set theory we can define predicates such as

- $s \subseteq t$
- α is an ordinal
- f is a function
- there is a surjection from s onto t (notation: $t \leq s$)

We can also define sets and operations on sets, such as $\aleph_0$, the least infinite cardinal and $\mathcal{P}(s)$, the power set of s.
Then we can state the continuum hypothesis as

$$\mathsf{CH} := \forall s,\ s \leq \aleph_0 \vee \mathcal{P}(\aleph_0) \leq s$$

So the independence of CH is the statement

$$\mathsf{ZFC} \nvdash \mathsf{CH} \quad \text{and} \quad \mathsf{ZFC} \nvdash \neg\mathsf{CH}.$$

Important: adding definable predicates, constants or operations to the language does not change the sentences that you can prove; it is a conservative extension.

Models

Given a language L, an L-structure M consists of

- a carrier set, also denoted M;
- for each n-ary function symbol f a function $f^M : M^n \to M$;
- for each n-ary relation symbol R a subset $R^M \subseteq M^n$.

Models

Given a language L, an L-structure M consists of

- a carrier set, also denoted M;
- for each n-ary function symbol f a function $f^M : M^n \to M$;
- for each n-ary relation symbol R a subset $R^M \subseteq M^n$.

If t is a term of L then we can interpret t in M (assuming we have an interpretation of all variables in t).

Models

Given a language L, an L-structure M consists of

- a carrier set, also denoted M;
- for each n-ary function symbol f a function $f^M : M^n \to M$;
- for each n-ary relation symbol R a subset $R^M \subseteq M^n$.

If t is a term of L then we can interpret t in M (assuming we have an interpretation of all variables in t).

If φ is a formula of L then φ is true or false in M (assuming we have an interpretation of all free variables in φ).

Models

Given a language L, an L-structure M consists of

- a carrier set, also denoted M;
- for each n-ary function symbol f a function $f^M : M^n \to M$;
- for each n-ary relation symbol R a subset $R^M \subseteq M^n$.

If t is a term of L then we can interpret t in M (assuming we have an interpretation of all variables in t).
If φ is a formula of L then φ is true or false in M (assuming we have an interpretation of all free variables in φ).
If φ is a sentence of L then φ is true or false in M.

Models

Given a language L, an L-structure M consists of

- a carrier set, also denoted M;
- for each n-ary function symbol f a function $f^M : M^n \to M$;
- for each n-ary relation symbol R a subset $R^M \subseteq M^n$.

If t is a term of L then we can interpret t in M (assuming we have an interpretation of all variables in t).
If φ is a formula of L then φ is true or false in M (assuming we have an interpretation of all free variables in φ).
If φ is a sentence of L then φ is true or false in M.

If T is a theory, then M is a model of T if every sentence in T is true in M.
We say that $\Gamma \vDash \varphi$, Γ models φ, if for every model of Γ the sentence φ holds.

Provability vs truth

We have a notion of provability: $\Gamma \vdash \varphi$;

We have a notion of truth: $\Gamma \vDash \varphi$;

How do these two notions relate?

Provability vs truth

We have a notion of provability: $\Gamma \vdash \varphi$;

We have a notion of truth: $\Gamma \vDash \varphi$;

How do these two notions relate?

Soundness theorem (easy): if $\Gamma \vdash \varphi$ then $\Gamma \vDash \varphi$.

Provability vs truth

We have a notion of provability: $\Gamma \vdash \varphi$;

We have a notion of truth: $\Gamma \vDash \varphi$;

How do these two notions relate?

Soundness theorem (easy): if $\Gamma \vdash \varphi$ then $\Gamma \vDash \varphi$.

Gödel's completeness theorem: $\Gamma \vDash \varphi$ then $\Gamma \vdash \varphi$.

Provability vs truth

We have a notion of provability: $\Gamma \vdash \varphi$;

We have a notion of truth: $\Gamma \vDash \varphi$;

How do these two notions relate?

Soundness theorem (easy): if $\Gamma \vdash \varphi$ then $\Gamma \vDash \varphi$.

Gödel's completeness theorem: $\Gamma \vDash \varphi$ then $\Gamma \vdash \varphi$.

To show that a theory doesn't prove φ it is sufficient to construct a model of Γ where φ fails.

Boolean-valued models

Given a language L and a Boolean algebra $\mathbb{B}$. A $\mathbb{B}$-valued structure M of L consists of

- a carrier set M;
- for each n-ary function symbol f a function $f^M : M^n \to M$;
- for each n-ary relation symbol R a function $R^M : M^n \to \mathbb{B}$.
- A function $=^M : M^2 \to \mathbb{B}$ satisfying the following conditions:
  - $(x =^M x) = \top$
  - $(x =^M y) = (y =^M x)$
  - $(x =^M y) \sqcap (y =^M z) \le (x =^M z)$
  - $\bigsqcap_i (x_i =^M y_i) \le (f(x_1, \ldots, x_n) =^M f(y_1, \ldots, y_n))$
  - $R(x_1, \ldots, x_n) \sqcap \bigsqcap_i (x_i =^M y_i) \le (R(y_1, \ldots, y_n))$

Boolean-valued soundness

Terms can be interpreted as elements in M and formulas as elements of $\mathbb{B}$, assuming we have interpreted their free variables.
Sentences φ are interpreted by an element $[\![\varphi]\!]_M$ of $\mathbb{B}$.

We say that $\Gamma \vDash_{\mathbb{B}} \varphi$ if for every $\mathbb{B}$-valued structure M we have

$$\prod_{\psi \in \Gamma} [\![\psi]\!]_M \leq [\![\varphi]\!]_M .$$

We then have the Boolean-valued soundness theorem: If $\Gamma \vdash \varphi$ then $\Gamma \vDash_{\mathbb{B}} \varphi$.

Type-theoretic model of ZFC

The Aczel-Werner encoding of set theory in type theory.

```
inductive V : Type (u+1)
| mk (α : Type u) (A : α → V) : V
```

Think of $s = \langle \alpha, A \rangle : V$ as a set where α is an indexing type and $A : \alpha \to V$ as pointing to the elements of s.
This is a model of set theory if we quotient by some equivalence relation.

Type-theoretic model of ZFC

The Aczel-Werner encoding of set theory in type theory.

```
inductive V : Type (u+1)
| mk (α : Type u) (A : α → V) : V
```

Think of $s = \langle \alpha, A \rangle : V$ as a set where α is an indexing type and $A : \alpha \to V$ as pointing to the elements of s.
This is a model of set theory if we quotient by some equivalence relation.

We will use a Boolean-valued version $V^{\mathbb{B}}$ of this:

```
inductive VB (𝔹 : Type u)
    [CompleteBooleanAlgebra 𝔹] : Type (u+1)
| mk (α : Type u) (A : α → VB 𝔹) (B : α → 𝔹) : VB 𝔹
```

Forcing

Theorem

If $\mathbb{B}$ is a complete Boolean algebra, then $V^{\mathbb{B}}$ is a $\mathbb{B}$-valued model of ZFC.

Forcing

Theorem

If $\mathbb{B}$ is a complete Boolean algebra, then $V^{\mathbb{B}}$ is a $\mathbb{B}$-valued model of ZFC.

Strategy: find a well-chosen complete Boolean algebra $\mathbb{B}_{\text{cohen}}$, such that CH fails in $V^{\mathbb{B}_{\text{cohen}}}$ and another complete Boolean algebra $\mathbb{B}_{\text{collapse}}$ such that CH holds in $V^{\mathbb{B}_{\text{collapse}}}$.

Check-names

We have an inclusion $x \mapsto \check{x} : V \to V^{\mathbb{B}}$

```
def check : V → VB 𝔹
| ⟨α, A⟩ := ⟨α, λ a ↦ check (A a), λ a ↦ ⊤⟩
```

This allows us to construct many sets explicitly in $V^{\mathbb{B}}$, such as ordinals.

Check-names

We have an inclusion $x \mapsto \check{x} : V \to V^{\mathbb{B}}$

```
def check : V → VB 𝔹
| ⟨α, A⟩ := ⟨α, λ a ↦ check (A a), λ a ↦ ⊤⟩
```

This allows us to construct many sets explicitly in $V^{\mathbb{B}}$, such as ordinals.

Warning: There is no guarantee that $\check{x}$ satisfies the same properties as x. For example, if $\aleph_1 \in V$ is the first uncountable cardinal, then $\check{\aleph}_1$ is not necessarily uncountable, and it is possible that there are infinitely many uncountable cardinalities below $\check{\aleph}_1$. It depends on $\mathbb{B}$.

Countable chain condition

Definition

Let P be a poset.
Two elements $a, b \in P$ are incomparable if neither $a \leq b$ nor $b \leq a$.
$A \subseteq P$ is an antichain if any two distinct element of A are incomparable.
P satisfies the countable chain condition (CCC) if every antichain included in P is countable.

Countable chain condition

Definition

Let P be a poset.
Two elements $a, b \in P$ are incomparable if neither $a \leq b$ nor $b \leq a$.
$A \subseteq P$ is an antichain if any two distinct element of A are incomparable.
P satisfies the countable chain condition (CCC) if every antichain included in P is countable.

Theorem

If $\mathbb{B}$ satisfies the CCC then $V^{\mathbb{B}}$ preserves cardinal inequalities, i.e. if A has a smaller cardinality than B in V then $\check{A}$ has a smaller cardinality than $\check{B}$ in $V^{\mathbb{B}}$.

Cohen forcing

Idea: find a complete Boolean algebra $\mathbb{B}$ that satisfies CCC but has a large number of extra subsets of $\mathbb{N}$, i.e. a large number of maps $\mathbb{N} \to \mathbb{B}$.

Cohen forcing

Idea: find a complete Boolean algebra $\mathbb{B}$ that satisfies CCC but has a large number of extra subsets of $\mathbb{N}$, i.e. a large number of maps $\mathbb{N} \to \mathbb{B}$.

Let $\aleph_2$ be the second uncountable cardinal.
Let $X = 2^{\aleph_2 \times \mathbb{N}}$, endowed with the product topology.

Cohen forcing

Idea: find a complete Boolean algebra $\mathbb{B}$ that satisfies CCC but has a large number of extra subsets of $\mathbb{N}$, i.e. a large number of maps $\mathbb{N} \to \mathbb{B}$.

Let $\aleph_2$ be the second uncountable cardinal.
Let $X = 2^{\aleph_2 \times \mathbb{N}}$, endowed with the product topology.

Let $\mathbb{B}_{\mathsf{cohen}} := \mathsf{RO}(X)$ be the complete Boolean algebra of regular opens in X (the opens U such that $\mathsf{int}(\overline{U}) = U$).

Cohen forcing

Idea: find a complete Boolean algebra $\mathbb{B}$ that satisfies CCC but has a large number of extra subsets of $\mathbb{N}$, i.e. a large number of maps $\mathbb{N} \to \mathbb{B}$.

Let $\aleph_2$ be the second uncountable cardinal.
Let $X = 2^{\aleph_2 \times \mathbb{N}}$, endowed with the product topology.

Let $\mathbb{B}_{\text{cohen}} := \text{RO}(X)$ be the complete Boolean algebra of regular opens in X (the opens U such that $\text{int}(\overline{U}) = U$).

For each $\alpha \in \aleph_2$ we have a map $\chi_\alpha : \mathbb{N} \to \mathbb{B}_{\text{cohen}}$ by

$$\chi_\alpha(n) := \{f \in X \mid f(\alpha, n) = 1\}.$$

This gives (with some work) internally in $V^{\mathbb{B}_{\text{cohen}}}$ an injective map $\aleph_2 \hookrightarrow \mathcal{P}(\mathbb{N})$.

Cohen forcing

Idea: find a complete Boolean algebra $\mathbb{B}$ that satisfies CCC but has a large number of extra subsets of $\mathbb{N}$, i.e. a large number of maps $\mathbb{N} \to \mathbb{B}$.

Let $\aleph_2$ be the second uncountable cardinal.
Let $X = 2^{\aleph_2 \times \mathbb{N}}$, endowed with the product topology.

Let $\mathbb{B}_{\text{cohen}} := \text{RO}(X)$ be the complete Boolean algebra of regular opens in X (the opens U such that $\text{int}(\overline{U}) = U$).

For each $\alpha \in \aleph_2$ we have a map $\chi_\alpha : \mathbb{N} \to \mathbb{B}_{\text{cohen}}$ by

$$\chi_\alpha(n) := \{f \in X \mid f(\alpha, n) = 1\}.$$

This gives (with some work) internally in $V^{\mathbb{B}_{\text{cohen}}}$ an injective map $\aleph_2 \hookrightarrow \mathcal{P}(\mathbb{N})$.

And $\mathbb{B}_{\text{cohen}}$ satisfies the CCC, so $\breve{\aleph}_0 < \breve{\aleph}_1 < \breve{\aleph}_2$.
Therefore, CH fails in $V^{\mathbb{B}_{\text{cohen}}}$.

Collapse forcing

To find a model where CH holds, we would like a surjection $\aleph_1 \to \mathcal{P}(\aleph_0)$.

Let $\mathbb{P}_{\text{collapse}}$ be the poset of countable partial functions $\aleph_1 \to \mathcal{P}(\aleph_0)$.
$\mathbb{B}_{\text{collapse}} := \text{RO}(\mathcal{P}(\aleph_0)^{\aleph_1})$ where the topology is generated by D_p for $p \in \mathbb{P}_{\text{collapse}}$ where

$$D_p := \{g : \aleph_1 \to \mathcal{P}(\aleph_0) \mid g \text{ extends } p\}.$$

Collapse forcing

To find a model where CH holds, we would like a surjection $\aleph_1 \to \mathcal{P}(\aleph_0)$.

Let $\mathbb{P}_{\text{collapse}}$ be the poset of countable partial functions $\aleph_1 \to \mathcal{P}(\aleph_0)$. $\mathbb{B}_{\text{collapse}} := \text{RO}(\mathcal{P}(\aleph_0)^{\aleph_1})$ where the topology is generated by D_p for $p \in \mathbb{P}_{\text{collapse}}$ where

$$D_p := \{g : \aleph_1 \to \mathcal{P}(\aleph_0) \mid g \text{ extends } p\}.$$

This choice of complete Boolean algebra gives a surjection $\check{\aleph}_1 \to \widecheck{\mathcal{P}(\aleph_0)}$ in $V^{\mathbb{B}_{\text{collapse}}}$.

Then we can show that $\check{\aleph}_1$ is the first uncountable ordinal in $V^{\mathbb{B}_{\text{collapse}}}$ and that $\widecheck{\mathcal{P}(\aleph_0)}$ is the same as $\mathcal{P}(\aleph_0)$ in $V^{\mathbb{B}_{\text{collapse}}}$.

Cautionary tale

Much of this formalized work did not end up in mathlib.

We focused on finishing the goal, and did not do all intermediate steps in a proper generality, which means they were not general enough for mathlib.

Cautionary tale

Much of this formalized work did not end up in mathlib.

We focused on finishing the goal, and did not do all intermediate steps in a proper generality, which means they were not general enough for mathlib.

Aaron Anderson did a lot of work porting the first-order logic (terms, formulas, models) to mathlib, and he improved the presentation in the process.

Improvement: terms

Old (Lean 3):

```
inductive preterm : ℕ → Type u
| var : ∀ (k : ℕ), preterm 0
| func : ∀ {l : ℕ} (f : L.functions l), preterm l
| app : ∀ {l : ℕ} (t : preterm (l + 1)) (s : preterm 0),
    preterm l

def term := preterm L 0
```

New (Lean 4):

```
inductive Term (α : Type _) : Type _
| var : α → Term α
| func : ∀ {n : ℕ} (f : L.Functions n)
  (ts : Fin n → Term α), Term α
```

Inequalities in complete Boolean algebras

We used automation for proving inequalities in complete boolean algebras. Suppose we want to prove

```
example {a b c : 𝔹} : (a ⇒ b) ⊓ (b ⇒ c) ≤ a ⇒ c
```

Inequalities in complete Boolean algebras

We used automation for proving inequalities in complete boolean algebras. Suppose we want to prove

```
example {a b c : 𝔹} : (a ⇒ b) ⊓ (b ⇒ c) ≤ a ⇒ c
```

This corresponds to the following tactic state:

```
a b c : Prop
h : (a → b) ∧ (b → c)
⊢ a → c
```

This is easy to prove using `rcases`, `intro` and `apply`.

Inequalities in complete Boolean algebras

We used automation for proving inequalities in complete boolean algebras. Suppose we want to prove

```
example {a b c : 𝔹} : (a ⇒ b) ⊓ (b ⇒ c) ≤ a ⇒ c
```

This corresponds to the following tactic state:

```
a b c : Prop
h : (a → b) ∧ (b → c)
⊢ a → c
```

This is easy to prove using `rcases`, `intro` and `apply`.

Trick: use a Yoneda-like lemma:

```
lemma yoneda (H : ∀ Γ, Γ ≤ a → Γ ≤ b) : a ≤ b
```

Inequalities in complete Boolean algebras

$\vdash$ `(a` $\Rightarrow$ `b)` $\sqcap$ `(b` $\Rightarrow$ `c)` $\leq$ `a` $\Rightarrow$ `c`

Inequalities in complete Boolean algebras

$\vdash$ `(a` $\Rightarrow$ `b)` $\sqcap$ `(b` $\Rightarrow$ `c)` $\leq$ `a` $\Rightarrow$ `c`

`h` : Γ $\leq$ `(a` $\Rightarrow$ `b)` $\sqcap$ `(b` $\Rightarrow$ `c)`
$\vdash$ Γ $\leq$ `a` $\Rightarrow$ `c`

Inequalities in complete Boolean algebras

$\vdash$ `(a ⇒ b) ⊓ (b ⇒ c)` $\leq$ `a ⇒ c`

`h` : $\Gamma \leq$ `(a ⇒ b) ⊓ (b ⇒ c)`
$\vdash \Gamma \leq$ `a ⇒ c`

`h1` : $\Gamma \leq$ `a ⇒ b`
`h2` : $\Gamma \leq$ `b ⇒ c`
$\vdash \Gamma \leq$ `a ⇒ c`

Inequalities in complete Boolean algebras

$\vdash$ (a $\Rightarrow$ b) $\sqcap$ (b $\Rightarrow$ c) $\leq$ a $\Rightarrow$ c

h : $\Gamma \leq$ (a $\Rightarrow$ b) $\sqcap$ (b $\Rightarrow$ c)
$\vdash \Gamma \leq$ a $\Rightarrow$ c

h1 : $\Gamma \leq$ a $\Rightarrow$ b
h2 : $\Gamma \leq$ b $\Rightarrow$ c
$\vdash \Gamma \leq$ a $\Rightarrow$ c

h1 : $\Gamma' \leq$ a $\Rightarrow$ b
h2 : $\Gamma' \leq$ b $\Rightarrow$ c
h3 : $\Gamma' \leq$ a
$\vdash \Gamma' \leq$ c

Inequalities in complete Boolean algebras

$\vdash$ `(a ⇒ b) ⊓ (b ⇒ c) ≤ a ⇒ c`

`h` : Γ `≤ (a ⇒ b) ⊓ (b ⇒ c)`
$\vdash$ Γ `≤ a ⇒ c`

`h1` : Γ `≤ a ⇒ b`
`h2` : Γ `≤ b ⇒ c`
$\vdash$ Γ `≤ a ⇒ c`

`h1` : Γ' `≤ a ⇒ b`
`h2` : Γ' `≤ b ⇒ c`
`h3` : Γ' `≤ a`
$\vdash$ Γ' `≤ c`

Now we can "apply" `h2` which gives us the new goal Γ' `≤ b`, and then we can "apply" `h1` to get the goal Γ' `≤ a`, which is true by assumption.

Conclusions

- We can formalize complicated forcing arguments.
- Try to do intermediate results in higher generality than needed and PR to mathlib early.
- Some domain-specific automation is very helpful.