A case study of forcing in classical realizability: Herbrand's theorem, proof and extracted algorithm

Lionel Rieg

LIP, ENS Lyon

Récré Meeting, November 15th 2012

Motivations

Extracting Herbrand trees

- through formalization in Coq and extraction:
  safe (adequacy) but slow (a lot of backtracks on the tree)
- direct program:
  faster but unsafe (implemented in a modified KAM)
- through a forcing transformation:
  the best of both worlds:
  - safe (adequacy with forcing)
  - fast (intuitionistic proof: no backtrack on the tree)

Outline

Outline

Idea behind Herbrand theorem

We start from an inconsistent universal theory U.
(*i.e.* a set of universal formulæ $U_i = \forall \vec{x}.F_i\,\vec{x}$ with F_i quantifier-free)

infinite interpretations $\rightarrow$ $\boxed{\text{compactness}}$ $\rightarrow$ finite information for each interpretation

Idea behind Herbrand theorem

We start from an inconsistent universal theory U.
(*i.e.* a set of universal formulæ $U_i = \forall \vec{x}.F_i\,\vec{x}$ with F_i quantifier-free)

infinite interpretations $\rightarrow$ $\boxed{\text{compactness}}$ $\rightarrow$ finite information for each interpretation

infinite number of interpretations $\rightarrow$ $\boxed{\text{Herbrand theorem}}$ $\rightarrow$ finite information for all interpretations

This information can be presented as a decision tree or BDD.

What is a Herbrand tree?

Definition (Herbrand tree)

A binary tree where

- inner nodes are labeled by atomic formulæ
- branches represent partial interpretations
- leaves contain contradictions

What is a Herbrand tree?

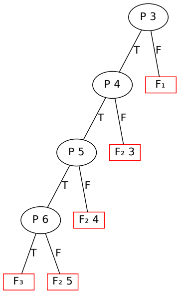
Definition (Herbrand tree)

A binary tree where

- inner nodes are labeled by atomic formulæ
- branches represent partial interpretations
- leaves contain contradictions

Example

- $F_1 = P\,3$
- $F_2 = \forall n. P\,n \to P\,(n+1)$
- $F_3 = \neg P\,6$



What is a Herbrand tree?

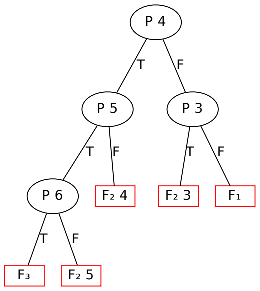
Definition (Herbrand tree)

A binary tree where

- inner nodes are labeled by atomic formulæ
- branches represent partial interpretations
- leaves contain contradictions

Example

- $F_1 = P\,3$
- $F_2 = \forall n. P\,n \to P\,(n+1)$
- $F_3 = \neg P\,6$



Usual proof of Herbrand's theorem

Theorem (Herbrand)

If for all interpretations $\mathscr{I}$, $\mathscr{I} \not\models U$, then U has a Herbrand tree.

Usual proof of Herbrand's theorem

Theorem (Herbrand)

If for all interpretations $\mathscr{I}$, $\mathscr{I} \not\models U$, then U has a Herbrand tree.

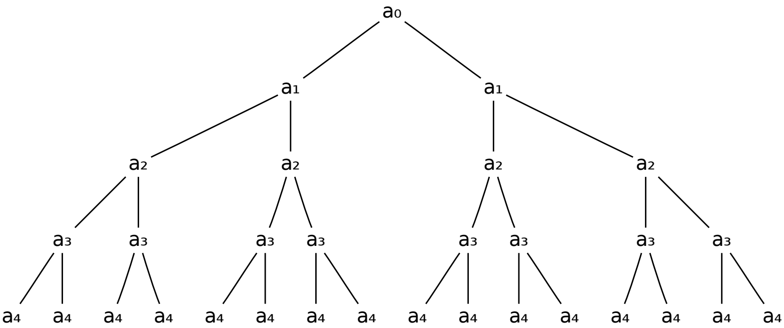
Let us fix an enumeration $(a_i)_{i \in \mathbb{N}}$ of the atoms.

(atoms = atomic formulæ)

Usual proof of Herbrand's theorem

Theorem (Herbrand)

If for all interpretations $\mathscr{I}$, $\mathscr{I} \not\models U$, then U has a Herbrand tree.

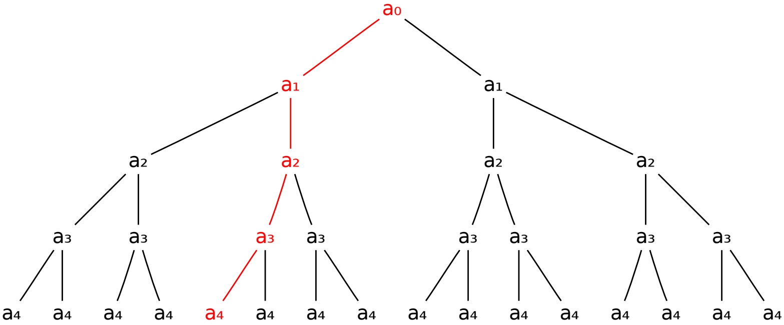


consider the atom-enumerating complete infinite tree

Usual proof of Herbrand's theorem

Theorem (Herbrand)

If for all interpretations $\mathscr{I}$, $\mathscr{I} \not\models U$, then U has a Herbrand tree.

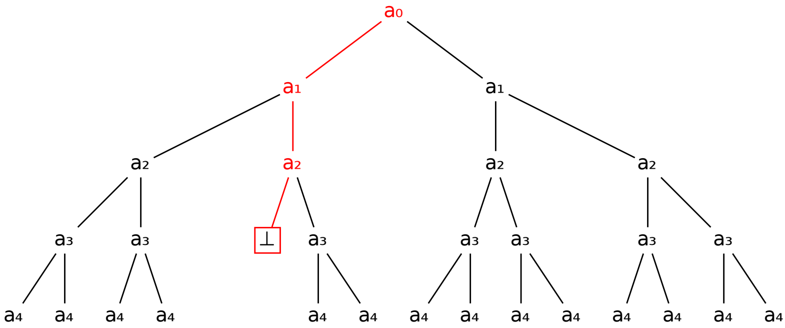


pick any infinite branch

Usual proof of Herbrand's theorem

Theorem (Herbrand)

If for all interpretations $\mathscr{I}$, $\mathscr{I} \not\models U$, then U has a Herbrand tree.

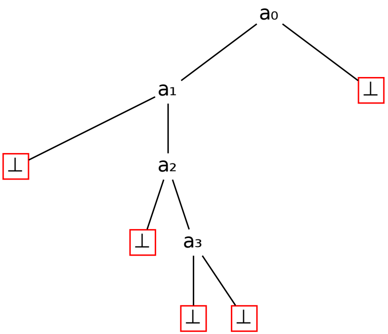


by hypothesis (and compactness), we can cut it at finite depth

Usual proof of Herbrand's theorem

Theorem (Herbrand)

If for all interpretations $\mathscr{I}$, $\mathscr{I} \not\models U$, then U has a Herbrand tree.



conclude using weak Kőnig's lemma

What can be improved

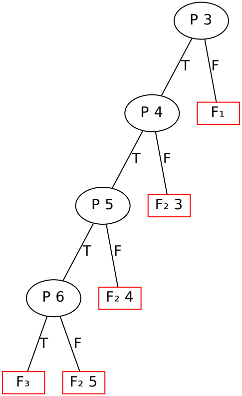
We want to avoid:

- the fixed enumeration of atoms
- weak Kőnig's lemma

What can be improved

We want to avoid:

- the fixed enumeration of atoms
- weak Kőnig's lemma

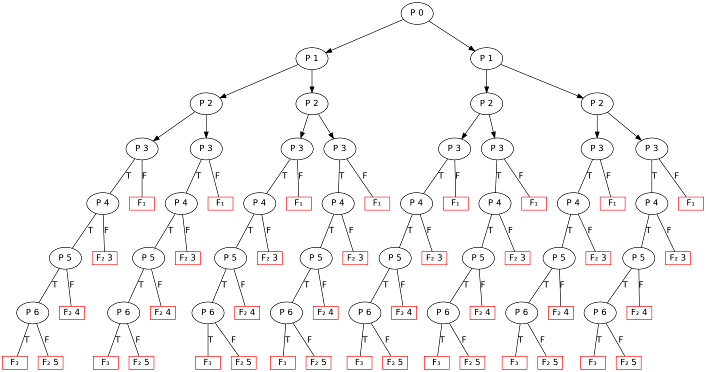


A good enumeration

What can be improved

We want to avoid:

- the fixed enumeration of atoms
- weak Kőnig's lemma



A bad enumeration

A simpler proof of Herbrand's theorem

Restriction: $U = \{\forall n.\, F\, n\}$ (take $F\,\langle i, \vec{x}\rangle = F_i\,\vec{x}$)

A simpler proof of Herbrand's theorem

Restriction: $U = \{\forall n.\, F\, n\}$ (take $F\,\langle i, \vec{x}\rangle = F_i\,\vec{x}$)

Proof. (by contraposition)

Assume that there is no Herbrand tree (for U) and build a model of U.

1. We start from a leaf.

A simpler proof of Herbrand's theorem

Restriction: $U = \{\forall n.\, F\, n\}$ (take $F\,\langle i, \vec{x}\rangle = F_i\, \vec{x}$)

Proof. (by contraposition)

Assume that there is no Herbrand tree (for U) and build a model of U.

1. We start from a leaf.
2. It is not a Herbrand tree so one of its leaves does not contain a contradiction.

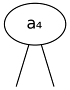
A simpler proof of Herbrand's theorem

Restriction: $U = \{\forall n.\, F\, n\}$ (take $F\,\langle i, \vec{x}\rangle = F_i\,\vec{x}$)

Proof. (by contraposition)

Assume that there is no Herbrand tree (for U) and build a model of U.

1. We start from a leaf.
2. It is not a Herbrand tree so one of its leaves does not contain a contradiction.
3. Replace this leaf with an inner node.



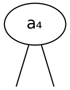
A simpler proof of Herbrand's theorem

Restriction: $U = \{\forall n.\, F\, n\}$ (take $F\,\langle i, \vec{x}\rangle = F_i\, \vec{x}$)

Proof. (by contraposition)

Assume that there is no Herbrand tree (for U) and build a model of U.

1. We start from a leaf.
2. It is not a Herbrand tree so one of its leaves does not contain a contradiction.
3. Replace this leaf with an inner node.
4. Repeat this process.



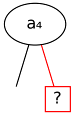
A simpler proof of Herbrand's theorem

Restriction: $U = \{\forall n.\, F\, n\}$ (take $F\,\langle i, \vec{x}\rangle = F_i\,\vec{x}$)

Proof. (by contraposition)

Assume that there is no Herbrand tree (for U) and build a model of U.

1. We start from a leaf.
2. It is not a Herbrand tree so one of its leaves does not contain a contradiction.
3. Replace this leaf with an inner node.
4. Repeat this process.



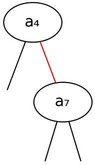
A simpler proof of Herbrand's theorem

Restriction: $U = \{\forall n.\, F\, n\}$ (take $F\,\langle i, \vec{x}\rangle = F_i\,\vec{x}$)

Proof. (by contraposition)

Assume that there is no Herbrand tree (for U) and build a model of U.

1. We start from a leaf.
2. It is not a Herbrand tree so one of its leaves does not contain a contradiction.
3. Replace this leaf with an inner node.
4. Repeat this process.



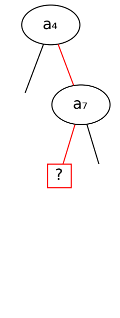
A simpler proof of Herbrand's theorem

Restriction: $U = \{\forall n.\, F\, n\}$ (take $F\,\langle i, \vec{x}\rangle = F_i\,\vec{x}$)

Proof. (by contraposition)

Assume that there is no Herbrand tree (for U) and build a model of U.

1. We start from a leaf.
2. It is not a Herbrand tree so one of its leaves does not contain a contradiction.
3. Replace this leaf with an inner node.
4. Repeat this process.



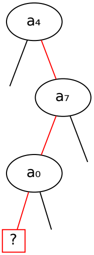
A simpler proof of Herbrand's theorem

Restriction: $U = \{\forall n.\, F\, n\}$ (take $F\,\langle i, \vec{x}\rangle = F_i\, \vec{x}$)

Proof. (by contraposition)

Assume that there is no Herbrand tree (for U) and build a model of U.

1. We start from a leaf.
2. It is not a Herbrand tree so one of its leaves does not contain a contradiction.
3. Replace this leaf with an inner node.
4. Repeat this process.



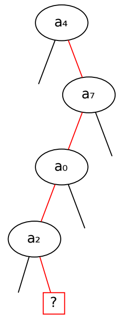
A simpler proof of Herbrand's theorem

Restriction: $U = \{\forall n.\, F\, n\}$ (take $F\,\langle i, \vec{x}\rangle = F_i\, \vec{x}$)

Proof. (by contraposition)

Assume that there is no Herbrand tree (for U) and build a model of U.

1. We start from a leaf.
2. It is not a Herbrand tree so one of its leaves does not contain a contradiction.
3. Replace this leaf with an inner node.
4. Repeat this process.



A simpler proof of Herbrand's theorem

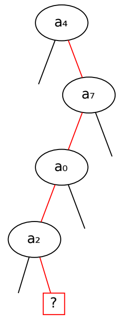
Restriction: $U = \{\forall n.\, F\, n\}$ (take $F\,\langle i, \vec{x}\rangle = F_i\, \vec{x}$)

Proof. (by contraposition)

Assume that there is no Herbrand tree (for U) and build a model of U.

1. We start from a leaf.
2. It is not a Herbrand tree so one of its leaves does not contain a contradiction.
3. Replace this leaf with an inner node.
4. Repeat this process.

The infinite branch is a model.



A simpler proof of Herbrand's theorem

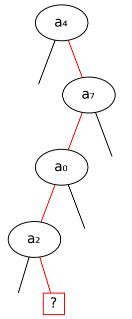
Restriction: $U = \{\forall n.\, F\, n\}$ (take $F\,\langle i, \vec{x}\rangle = F_i\,\vec{x}$)

Proof. (by contraposition)

Assume that there is no Herbrand tree (for U) and build a model of U.

1. We start from a leaf.
2. It is not a Herbrand tree so one of its leaves does not contain a contradiction.
3. Replace this leaf with an inner node.
4. Repeat this process.

The infinite branch is a model.



But proof is classical $\rightsquigarrow$ backtracking

A simpler proof of Herbrand's theorem

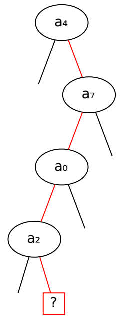
Restriction: $U = \{\forall n.\, F\, n\}$ (take $F\,\langle i, \vec{x}\rangle = F_i\, \vec{x}$)

Proof. (by contraposition)

Assume that there is no Herbrand tree (for U) and build a model of U.

1. We start from a leaf.
2. It is not a Herbrand tree so one of its leaves does not contain a contradiction.
3. Replace this leaf with an inner node.
4. Repeat this process.

The infinite branch is a model.



But proof is classical $\leadsto$ backtracking

This infinite branch can be provided by a Cohen real

Checking Herbrand trees

How can we check that a tree t is a Herbrand tree?

Checking Herbrand trees

How can we check that a tree t is a Herbrand tree?

By induction on t:

- while descending along t, remember a finite valuation p
- use this finite valuation p for evaluation at the leaves

Checking Herbrand trees

How can we check that a tree t is a Herbrand tree?

By induction on t: subHtree : finite valuation → tree → Bool

- while descending along t, remember a finite valuation p
  $\text{subHtree}\, p\, (\text{Node}\, a\, t_1\, t_2) =$
  $\text{subHtree}\, (a^+ \cup p)\, t_1 \;\&\&\; \text{subHtree}\, (a^- \cup p)\, t_2$
- use this finite valuation p for evaluation at the leaves
  $\text{subHtree}\, p\, (\text{Leaf}\, n) = \text{eval}\, p\, (F\, n)\, 0$

Checking Herbrand trees

How can we check that a tree t is a Herbrand tree?

By induction on t: subHtree : finite valuation $\to$ tree $\to$ Bool

- while descending along t, remember a finite valuation p
  subHtree p (Node a t_1 t_2) $=$
  subHtree $(a^+ \cup p)$ t_1 && subHtree $(a^- \cup p)$ t_2
- use this finite valuation p for evaluation at the leaves
  subHtree p (Leaf n) $=$ eval p $(F\ n)$ 0

t is a Herbrand tree $\iff$ subHtree $\varnothing$ $t = 1$

Checking Herbrand trees

How can we check that a tree t is a Herbrand tree?

By induction on t: subHtree : finite valuation $\to$ tree $\to$ Bool

- while descending along t, remember a finite valuation p
  $$\text{subHtree}\, p\, (\text{Node}\, a\, t_1\, t_2) = \text{subHtree}\, (a^+ \cup p)\, t_1\ \&\&\ \text{subHtree}\, (a^- \cup p)\, t_2$$
- use this finite valuation p for evaluation at the leaves
  $$\text{subHtree}\, p\, (\text{Leaf}\, n) = \text{eval}\, p\, (F\, n)\, 0$$

t is a Herbrand tree $\iff$ subHtree $\varnothing\, t = 1$

Remarks

- $\text{eval}\, p\, (F\, n)\, b = 1$ means:
  1. p has enough information to evaluate $F\, n$
  2. the result of evaluation is b
- building the tree bottom-up:
  unordered partial valuation $\rightsquigarrow$ ordered branches

Outline

1. Herbrand theorem
2. PAω
3. Forcing
4. The proof

Higher order Peano Arithmetic

1. Logical setting:
   - minimal language: $\forall$, $\rightarrow$
   - multi-sorted classical logic

     basic sorts: ι individuals

     o propositions (higher-order)

Higher order Peano Arithmetic

1. Logical setting:
   - minimal language: $\forall$, $\rightarrow$
   - multi-sorted classical logic

     basic sorts: ι individuals

     o propositions (higher-order)
2. datatype: relativization predicate on the sort ι

Example

$$\begin{array}{lrcl} \text{Bool}: & b \in \text{Bool} & := & \forall Z.\, Z(0) \rightarrow Z(1) \rightarrow Z(b) \\ \mathbb{N}: & x \in \mathbb{N} & := & \forall Z.\, Z(0) \rightarrow \big(\forall n, Z(n) \rightarrow Z(n+1)\big) \rightarrow Z(x) \end{array}$$

Notations

$$\forall n \in \mathbb{N}.\, A := \forall n.\, n \in \mathbb{N} \rightarrow A \qquad \exists n \in \mathbb{N}.\, A := \exists n.\, n \in \mathbb{N} \wedge A$$

Datatypes used in the formalization

- Inductive definitions:

Atom	(atomic formulæ)	`<abstract datatype>`
Comp	($\forall$/$\exists$-free formulæ)	$c \;:=\; \bot \;\mid\; \mathrm{Atomic}\, a \;\mid\; c \Rightarrow c$
tree	(binary trees)	$t \;:=\; \mathrm{Leaf}\, n \;\mid\; \mathrm{Node}\, a\; t\; t$
FVal	(finite valuations)	$p \;:=\; \varnothing \;\mid\; a^{+} :: p \;\mid\; a^{-} :: p$
		$a \in \mathrm{Atom}, a \notin p$

Datatypes used in the formalization

- Inductive definitions:

Atom	(atomic formulæ)	`<abstract datatype>`
Comp	(∀/∃-free formulæ)	$c \ := \ \bot \mid \mathrm{Atomic}\, a \mid c \Rightarrow c$
tree	(binary trees)	$t \ := \ \mathrm{Leaf}\, n \mid \mathrm{Node}\, a\, t\, t$
FVal	(finite valuations)	$p \ := \ \varnothing \mid a^{+} :: p \mid a^{-} :: p$ $a \in \mathrm{Atom}, a \notin p$

- Relativization predicates:
  - $$c \in \mathrm{Comp} := \forall Z.\ Z(\bot) \to \big(\forall a \in \mathrm{Atom}\,.\, Z(\mathrm{Atomic}\, a)\big) \to \big(\forall c_1.\forall c_2.\, Z(c_1) \to Z(c_2) \to Z(c_1 \Rightarrow c_2)\big) \to Z(c)$$
  - $$p \in \mathrm{FVal} := \forall Z.\ Z(\varnothing) \to \big(\forall q.\forall a \in \mathrm{Atom}.\, a \notin q \mapsto Z(q) \to Z(a^{+} :: q)\big) \to \big(\forall q.\forall a \in \mathrm{Atom}.\, a \notin q \mapsto Z(q) \to Z(a^{-} :: q)\big) \to Z(p)$$
  - $t \in \mathrm{tree} := \ldots$

Herbrand's theorem in PAω

Usual statement: $(\forall \mathscr{I}.\, \mathscr{I} \not\models U) \to \exists t.\, t$ is a Herbrand tree (for U)

- premise:
- conclusion:

Herbrand's theorem in PAω

Usual statement: $(\forall \mathscr{I}.\, \mathscr{I} \not\models U) \rightarrow \exists t.\, t$ is a Herbrand tree (for U)

- premise: some transformations
$$\begin{aligned} \forall \mathscr{I}.\, \mathscr{I} \not\models U \quad &\equiv \quad \forall \mathscr{I}.\, \mathscr{I} \not\models (\forall n \in \mathbb{N}.\, F\, n) \\ &\Longleftrightarrow \forall \mathscr{I}.\, \exists n \in \mathbb{N}.\, \mathscr{I} \not\models F\, n \\ &\Longleftrightarrow \forall^{\iota \rightarrow o} \rho.\, \exists n \in \mathbb{N}.\, \neg(\mathrm{interp}\, \rho\, (F\, n)) \end{aligned}$$
- conclusion:

Herbrand's theorem in PAω

Usual statement: $(\forall \mathscr{I}.\, \mathscr{I} \not\models U) \to \exists t.\, t$ is a Herbrand tree (for U)

- premise: some transformations
$$\begin{aligned} \forall \mathscr{I}.\, \mathscr{I} \not\models U \;\; &\equiv \;\; \forall \mathscr{I}.\, \mathscr{I} \not\models (\forall n \in \mathbb{N}.\, F\, n) \\ &\Longleftrightarrow \forall \mathscr{I}.\, \exists n \in \mathbb{N}.\, \mathscr{I} \not\models F\, n \\ &\Longleftrightarrow \forall^{\iota \to o} \rho.\, \exists n \in \mathbb{N}.\, \neg(\mathrm{interp}\, \rho\, (F\, n)) \end{aligned}$$
- conclusion: direct translation
$\exists t \in \mathrm{tree}.\; \mathrm{subHtree}\, \varnothing\, t = 1$

Herbrand's theorem in PAω

Usual statement: $(\forall \mathscr{I}.\, \mathscr{I} \not\models U) \to \exists t.\, t$ is a Herbrand tree (for U)

- premise: some transformations
$$\begin{aligned} \forall \mathscr{I}.\, \mathscr{I} \not\models U \quad &\equiv \quad \forall \mathscr{I}.\, \mathscr{I} \not\models (\forall n \in \mathbb{N}.\, F\, n) \\ &\Longleftrightarrow \forall \mathscr{I}.\, \exists n \in \mathbb{N}.\, \mathscr{I} \not\models F\, n \\ &\Longleftrightarrow \forall^{\iota \to o} \rho.\, \exists n \in \mathbb{N}.\, \neg(\mathrm{interp}\, \rho\, (F\, n)) \end{aligned}$$
- conclusion: direct translation
$\exists t \in \mathrm{tree}.\ \mathrm{subHtree}\, \varnothing\, t = 1$

We write $\mathrm{subH}\, p := \exists t \in \mathrm{tree}.\ \mathrm{subHtree}\, p\, t = 1$.

$$\big(\forall^{\iota \to o} \rho.\, \exists n \in \mathbb{N}.\, \neg(\mathrm{interp}\, \rho\, (U\, n))\big) \;\to\; \mathrm{subH}\, \varnothing$$

Outline

Recall on the forcing transformation

Taken from [Kri11] and [Miq11]

Input (forcing structure)

A forcing structure is given by

- a set (κ, C) of forcing conditions ($p \in C$ written $C[p]$)
- a product operation $\cdot$ on forcing conditions
- a maximal condition 1
- a bunch of proof terms $\alpha_0, \ldots, \alpha_8$

Output (program transformation)

A logical transformation:

$$t : A \quad \rightsquigarrow \quad t^* : p \Vdash A$$

Adequacy lemmas:

$$t : A \rightarrow t \Vvdash A \qquad\qquad t : A \rightarrow t^* \Vvdash p \Vdash A$$

A wrapper w s.t. $\quad t : 1 \Vdash A \rightarrow w\, t : A \quad$ (if A arithmetical)

In our case (1/3): prerequisites

Implementation of finite relations over Atom $\times$ Bool into ι:

In our case (1/3): prerequisites

Implementation of finite relations over Atom $\times$ Bool into ι:

- Primitives:

empty relation	$\varnothing$	:	ι
singleton relation	sing	:	$\iota \rightarrow \iota \rightarrow \iota$ (notations: a^+ and a^-)
union of relation	$\cup$	:	$\iota \rightarrow \iota \rightarrow \iota$
test of a binding	test	:	$\iota \rightarrow \iota \rightarrow \iota \rightarrow \iota$

In our case (1/3): prerequisites

Implementation of finite relations over Atom $\times$ Bool into ι:

- Primitives:

empty relation	$\varnothing$	$: \iota$
singleton relation	sing	$: \iota \to \iota \to \iota$ (notations: a^+ and a^-)
union of relation	$\cup$	$: \iota \to \iota \to \iota$
test of a binding	test	$: \iota \to \iota \to \iota \to \iota$

- Properties:
  - commutativity, associativity, idempotence of $\cup$
  - $\varnothing$ is a neutral element for $\cup$
  - totality of test: $\forall a \in \text{Atom}\,.\forall b \in \text{Bool}\,.\forall p.\ \text{test}\ a\ b\ p \in \text{Bool}$
  - behavior of test w.r.t. $\cup$, $\varnothing$, a^+, a^-

In our case (1/3): prerequisites

Implementation of finite relations over Atom $\times$ Bool into ι:

- Primitives:

empty relation	$\varnothing$	:	ι
singleton relation	sing	:	$\iota \to \iota \to \iota$ (notations: a^+ and a^-)
union of relation	$\cup$	:	$\iota \to \iota \to \iota$
test of a binding	test	:	$\iota \to \iota \to \iota \to \iota$

- Properties:
  - commutativity, associativity, idempotence of $\cup$
  - $\varnothing$ is a neutral element for $\cup$
  - totality of test: $\forall a \in \text{Atom}\,.\forall b \in \text{Bool}\,.\forall p.\ \text{test}\ a\ b\ p \in \text{Bool}$
  - behavior of test w.r.t. $\cup$, $\varnothing$, a^+, a^-
- Two definitions:

membership test	$\text{mem}\ a\ p$	:=	$\text{test}\ a\ 0\ p \parallel \text{test}\ a\ 1\ p$
adding a binding	$(a, b) :: p$	:=	$\text{sing}\ a\ b \cup p$

In our case (2/3): the forcing structure

Our forcing structure:

- $\kappa := \iota, \qquad C[p] := p \in \mathrm{FVal} \wedge (\mathrm{subH}\, p \to \mathrm{subH}\, \varnothing)$
- $p \cdot q := p \cup q$
- $1 := \varnothing$
- $\alpha_0, \ldots, \alpha_8$ comes from the interface of finite relations

Remarks

- C is a datatype
- FVal represents finite functions from ω to 2
- $C[p] = p \in FVal \qquad$ gives Cohen forcing conditions
  $\rightsquigarrow$ add a single new (non computable) real number

In our case (3/3): the generic set G

Forcing universe = Base universe + a new set G

$PA\omega + G$	$\longrightarrow$	$\boxed{\text{Forcing translation}}$	$\longrightarrow$	$PA\omega$
A				$p \Vdash A$
$t : A$				$t^* : p \Vdash A$
$q \in G$				$p \leqslant q$

In our case (3/3): the generic set G

Forcing universe = Base universe + a new set G

$PA\omega + G$	$\longrightarrow$	$\boxed{\text{Forcing translation}}$	$\longrightarrow$	$PA\omega$
A				$p \Vdash A$
$t : A$				$t^* : p \Vdash A$
$q \in G$				$p \leqslant q$

- Nice properties of G in the forcing universe:

non empty	$G(1)$
included in C	$\forall p.\, G(p) \to C[p]$
filter	$\forall p \forall q.\, G(p) \to G(q) \to G(p \cdot q)$
genericity	(we shall use instead a simple instance)

- Simple translation in the base universe

$$p \Vdash q \in G \quad \equiv \quad p \leqslant q \quad := \quad \forall r.\, C[p \cdot r] \to C[q \cdot r]$$

Outline

The big picture

The proof is split across the forcing and base universes.

Base universe	Forcing universe

The big picture

The proof is split across the forcing and base universes.

Base universe	Forcing universe
1. Build the forcing structure	

The big picture

The proof is split across the forcing and base universes.

Base universe	Forcing universe
1. Build the forcing structure 2. Assume the premise	

The big picture

The proof is split across the forcing and base universes.

Base universe	Forcing universe
1. Build the forcing structure	
2. Assume the premise	
	3. Lift the premise

The big picture

The proof is split across the forcing and base universes.

Base universe

1. Build the forcing structure
2. Assume the premise

Forcing universe

3. Lift the premise
4. Make the proof
   t : subH $\varnothing$

The big picture

The proof is split across the forcing and base universes.

Base universe

1. Build the forcing structure
2. Assume the premise
5. Use the forcing translation $t^* : 1 \Vdash \mathrm{subH}\,\varnothing$

Forcing universe

3. Lift the premise
4. Make the proof $t : \mathrm{subH}\,\varnothing$

The big picture

The proof is split across the forcing and base universes.

Base universe	Forcing universe
1. Build the forcing structure	
2. Assume the premise	
	3. Lift the premise
	4. Make the proof $t : \mathrm{subH}\,\varnothing$
5. Use the forcing translation $t^* : 1 \Vdash \mathrm{subH}\,\varnothing$	
6. Remove forcing $w\,t^* : \mathrm{subH}\,\varnothing$	

The big picture

The proof is split across the forcing and base universes.

Base universe

1. Build the forcing structure
2. Assume the premise
5. Use the forcing translation
   $t^* : 1 \Vdash \mathrm{subH}\, \varnothing$
6. Remove forcing
   $w\, t^* : \mathrm{subH}\, \varnothing$
7. Extract a witness
   (classical realizability)

Forcing universe

3. Lift the premise
4. Make the proof
   $t : \mathrm{subH}\, \varnothing$

Step 4: The proof (in the forcing universe)

Best done on the blackboard

Step 5: Translating axioms

The usual axioms of G are already done.

(G non empty, G included in C, G filter)

Step 5: Translating axioms

The usual axioms of G are already done.

(G non empty, G included in C, G filter)

Last axiom to translate:

$$\forall a \in \text{Atom}.\, \exists p \in G.\, \exists b \in \text{Bool}.\ \text{test}\ a\ b\ p = 1$$

i.e. find a proof term for

$$1 \Vdash \forall a \in \text{Atom}.\, \exists p \in G.\, \exists b \in \text{Bool}.\ \text{test}\ a\ b\ p = 1$$

Two solutions $\rightsquigarrow$ two ways of building the tree:

left-to-right scan | right-to-left scan

Step 6: Absolute formulæ

Definition (Absolute formula)

A formula is absolute when $\forall p \in C.\, A \leftrightarrow p \Vdash A$
that is there exists proof terms
$\xi_A : p \Vdash A \to C[p] \to A$
$\xi'_A : (C[p] \to A) \to p \Vdash A$

- Absolute sort: $\iota^* = \iota$ whereas $o^* = \kappa \to o$
  everything in $\iota \quad \leadsto \quad$ no computational object changes
- Absolute set = absolute sort + absolute relativization predicate
  - example: datatypes ($\mathbb{N}$, Bool, FVal, Atom, Comp, tree)
    equality on ι
  - counter-example: total valuations $\iota \to o$

$\text{subH}\,\varnothing \;:=\; \exists t \in \text{tree}.\ \text{subHtree}\,\varnothing\, t = 1$ is absolute

The wrapper w is simply ξ

Extracted algorithm

- No fixed enumeration needed

  $\leadsto$ enumeration built by the proof of the premise
- Herbrand tree inside forcing condition:

$p \in \mathrm{FVal}$	position in the tree
$\mathrm{subH}\, p \to \mathrm{subH}\, \varnothing$	context of the current tree
- Intuitionistic proof: no backtrack in real mode
- Execution in the KFAM

Conclusion

- One example of forcing on individuals
- Example of forcing to find better proofs, not new results
- Goal reached: fast and safe algorithm

Conclusion

- One example of forcing on individuals
- Example of forcing to find better proofs, not new results
- Goal reached: fast and safe algorithm

Thank you