Unified Classical Logic Completeness

A Coinductive Pearl

Jasmin Blanchette   Andrei Popescu   Dmitriy Traytel

Technische Universität München

Logic for Computer Science

Foundations of Automatic Theorem Proving

Jean H. Gallier

All too often, proof-theoretic methods are neglected in favor of shorter, and superficially more elegant semantic arguments.

[In contrast, in Gallier's book] the treatment of the proof theory of the Gentzen system is oriented towards computation with proofs. For example, a pseudo-Pascal version of a complete search procedure for first-order cut-free Gentzen proofs is presented.

Frank Pfenning

A Proof

$$\forall x.\, p(x) \vdash p(y) \wedge p(z)$$

A Proof

$$\dfrac{\forall x.\, p(x) \vdash p(y) \qquad\qquad \forall x.\, p(x) \vdash p(z)}{\forall x.\, p(x) \vdash p(y) \land p(z)}\ \textsc{ConjR}_{p(y),\, p(z)}$$

A Proof

$$\dfrac{\textsc{AllL}_{x,p(x),y}\ \dfrac{\forall x.\, p(x),\ p(y) \vdash p(y)}{\forall x.\, p(x) \vdash p(y)} \qquad \forall x.\, p(x) \vdash p(z)}{\forall x.\, p(x) \vdash p(y) \land p(z)}\ \textsc{ConjR}_{p(y),\, p(z)}$$

A Proof

$$
\dfrac{\dfrac{}{\forall x.\, p(x),\, p(y) \vdash p(y)}\ \textsc{Ax}_{p(y)}}{\forall x.\, p(x) \vdash p(y)}\ \textsc{AllL}_{x,p(x),y} \qquad \forall x.\, p(x) \vdash p(z)
$$

$$
\dfrac{\forall x.\, p(x) \vdash p(y) \qquad \forall x.\, p(x) \vdash p(z)}{\forall x.\, p(x) \vdash p(y) \wedge p(z)}\ \textsc{ConjR}_{p(y),\, p(z)}
$$

A Proof

$$
\dfrac{\dfrac{\dfrac{}{\forall x.\, p(x),\, p(y) \vdash p(y)}\;\text{Ax}_{p(y)}}{\forall x.\, p(x) \vdash p(y)}\;\text{AllL}_{x,p(x),y} \qquad \dfrac{\forall x.\, p(x),\, p(z) \vdash p(z)}{\forall x.\, p(x) \vdash p(z)}\;\text{AllL}_{x,p(x),z}}{\forall x.\, p(x) \vdash p(y) \wedge p(z)}\;\text{ConjR}_{p(y),\,p(z)}
$$

A Proof

$$
\dfrac{
\dfrac{
\dfrac{}{\forall x.\, p(x),\ p(y) \vdash p(y)}\ \textsc{Ax}_{p(y)}
}{\forall x.\, p(x) \vdash p(y)}\ \textsc{AllL}_{x,p(x),y}
\qquad
\dfrac{
\dfrac{}{\forall x.\, p(x),\ p(z) \vdash p(z)}\ \textsc{Ax}_{p(z)}
}{\forall x.\, p(x) \vdash p(z)}\ \textsc{AllL}_{x,p(x),z}
}{\forall x.\, p(x) \vdash p(y) \wedge p(z)}\ \textsc{ConjR}_{p(y),\,p(z)}
$$

A Failing Proof

$$
\dfrac{\dfrac{\dfrac{}{\forall x.\, p(x),\, p(y) \vdash p(y)}\,\textsc{Ax}_{p(y)}}{\forall x.\, p(x) \vdash p(y)}\,\textsc{AllL}_{x,p(x),y} \qquad \forall x.\, p(x) \vdash p(z)}{\forall x.\, p(x) \vdash p(y) \wedge p(z)}\,\textsc{ConjR}_{p(y),\,p(z)}
$$

A Failing Proof

$$\dfrac{\textsc{AllL}_{x,p(x),y}\ \dfrac{\textsc{Ax}_{p(y)}\ \dfrac{}{\forall x.\,p(x),\ p(y) \vdash p(y)}}{\forall x.\,p(x) \vdash p(y)} \qquad \dfrac{\forall x.\,p(x),\ p(y) \vdash p(z)}{\forall x.\,p(x) \vdash p(z)}\ \textsc{AllL}_{x,p(x),y}}{\forall x.\,p(x) \vdash p(y) \wedge p(z)}\ \textsc{ConjR}_{p(y),\,p(z)}$$

A Failing Proof

$$
\dfrac{
  \dfrac{
    \dfrac{}{\forall x.\, p(x),\ p(y) \vdash p(y)}\ \textsc{Ax}_{p(y)}
  }{\forall x.\, p(x) \vdash p(y)}\ \textsc{AllL}_{x,p(x),y}
  \qquad
  \dfrac{
    \dfrac{\forall x.\, p(x),\ p(y) \vdash p(z)}{\forall x.\, p(x),\ p(y) \vdash p(z)}\ \textsc{AllL}_{x,p(x),y}
  }{\forall x.\, p(x) \vdash p(z)}\ \textsc{AllL}_{x,p(x),y}
}{\forall x.\, p(x) \vdash p(y) \wedge p(z)}\ \textsc{ConjR}_{p(y),\,p(z)}
$$

A Failing Proof

$$
\dfrac{
\dfrac{
\dfrac{}{\forall x.\, p(x),\ p(y) \vdash p(y)}\ \textsc{Ax}_{p(y)}
}{\forall x.\, p(x) \vdash p(y)}\ \textsc{AllL}_{x,p(x),y}
\qquad
\dfrac{
\dfrac{
\dfrac{
\dfrac{\vdots}{\forall x.\, p(x),\ p(y) \vdash p(z)}\ \textsc{AllL}_{x,p(x),y}
}{\forall x.\, p(x),\ p(y) \vdash p(z)}\ \textsc{AllL}_{x,p(x),y}
}{\forall x.\, p(x),\ p(y) \vdash p(z)}\ \textsc{AllL}_{x,p(x),y}
}{\forall x.\, p(x) \vdash p(z)}\ \textsc{AllL}_{x,p(x),y}
}{\forall x.\, p(x) \vdash p(y) \wedge p(z)}\ \textsc{ConjR}_{p(y),\, p(z)}
$$

A Systematic Proof

$$
\dfrac{
  \dfrac{
    \dfrac{}{\forall x.\, p(x),\ p(y) \vdash p(y)}\ \textsc{Ax}_{p(y)}
  }{\forall x.\, p(x) \vdash p(y)}\ \textsc{AllL}_{x,p(x),y}
  \qquad
  \dfrac{
    \dfrac{
      \dfrac{
        \dfrac{}{\forall x.\, p(x),\ p(z) \vdash p(z)}\ \textsc{Ax}_{p(z)}
      }{\forall x.\, p(x),\ p(y) \vdash p(z)}\ \textsc{AllL}_{x,p(x),z}
    }{\forall x.\, p(x),\ p(x) \vdash p(z)}\ \textsc{AllL}_{x,p(x),y}
  }{\forall x.\, p(x) \vdash p(z)}\ \textsc{AllL}_{x,p(x),x}
}{\forall x.\, p(x) \vdash p(y) \wedge p(z)}\ \textsc{ConjR}_{p(y),\, p(z)}
$$

A Failing Systematic Proof

$$\dfrac{\dfrac{\dfrac{}{\forall x.\,p(x),\,p(y)\vdash p(y)}\,\mathrm{Ax}_{p(y)}}{\forall x.\,p(x)\vdash p(y)}\,\mathrm{AllL}_{x,p(x),y} \qquad \dfrac{\dfrac{\dfrac{\dfrac{\vdots}{\forall x.\,p(x),\,p(z)\vdash q(z)}\,\mathrm{AllL}_{x,p(x),y}}{\forall x.\,p(x),\,p(y)\vdash q(z)}\,\mathrm{AllL}_{x,p(x),z}}{\forall x.\,p(x),\,p(x)\vdash q(z)}\,\mathrm{AllL}_{x,p(x),y}}{\forall x.\,p(x)\vdash q(z)}\,\mathrm{AllL}_{x,p(x),x}}{\forall x.\,p(x)\vdash p(y)\land q(z)}\,\mathrm{ConjR}_{p(y),\,p(z)}$$

Our Interest in Gallier's Proof

$$\underbrace{\text{Polymorphic FOL}}_{\text{TFF1}} \longrightarrow \underbrace{\text{Monomorphic FOL}}_{\text{TFF0}} \longrightarrow \underbrace{\text{Untyped FOL}}_{\text{FOF}}$$

Our Interest in Gallier's Proof

$$\underbrace{\text{Polymorphic FOL}}_{\text{TFF1}} \longrightarrow \underbrace{\text{Monomorphic FOL}}_{\text{TFF0}} \longrightarrow \underbrace{\text{Untyped FOL}}_{\text{FOF}}$$

Claessen, Lillieström, Smallbone CADE 2011

Blanchette, Böhme, Popescu, Smallbone TACAS 2013

Our Interest in Gallier's Proof

$\underbrace{\text{Polymorphic FOL}}_{\text{TFF1}} \longrightarrow \underbrace{\text{Monomorphic FOL}}_{\text{TFF0}} \longrightarrow \underbrace{\text{Untyped FOL}}_{\text{FOF}}$

Claessen, Lillieström, Smallbone CADE 2011
Blanchette, Böhme, Popescu, Smallbone TACAS 2013

Blanchette, Popescu FroCoS 2013

How to Formalize Completeness?

Harrison TPHOLs 1998	HOL Light	Henkin
Berghofer 2002	Isabelle/HOL	Henkin
Ridge, Margetson TPHOLs 2005	Isabelle/HOL	Beth–Hintikka
Ilik 2010	Coq	Henkin
Schlöder, Koepke 2012	Mizar	Henkin

Henkin $\approx$ Gödel $\approx$ canonical models $\approx$ semantic

Beth–Hintikka $\approx$ Gallier $\approx$ complete prover $\approx$ syntactic

How to Formalize Completeness?

Harrison TPHOLs 1998	HOL Light	Henkin
Berghofer 2002	Isabelle/HOL	Henkin
Ridge, Margetson TPHOLs 2005	Isabelle/HOL	Beth–Hintikka
Ilik 2010	Coq	Henkin
Schlöder, Koepke 2012	Mizar	Henkin

Henkin ≈ Gödel ≈ canonical models ≈ semantic

Beth–Hintikka ≈ Gallier ≈ complete prover ≈ syntactic

Our Version

- Isabelle/HOL, Beth–Hintikka
- Abstract proof + instantiation with rich FOLs
- Codatatype of possibly infinite trees
- Code generation to Haskell

Our Version

- Isabelle/HOL, Beth–Hintikka
- Abstract proof + instantiation with rich FOLs
- Codatatype of possibly infinite trees
- Code generation to Haskell

A pseudo-Pascal version of a complete search procedure for first-order cut-free Gentzen proofs is presented.

Our Version

- Isabelle/HOL, Beth–Hintikka
- Abstract proof + instantiation with rich FOLs
- Codatatype of possibly infinite trees
- Code generation to Haskell

Haskell

A ~~pseudo-Pascal~~ version of a complete search procedure for first-order cut-free Gentzen proofs is presented.

Isabelle/HOL Demonstration

Codatatypes

Inductive (or algebraic) datatypes:

$\textbf{datatype}\ \alpha\ \mathsf{list} = \mathsf{Nil} \mid \mathsf{Cons}\ \alpha\ (\alpha\ \mathsf{list})$

Codatatypes

Inductive (or algebraic) datatypes:

datatype α list $=$ Nil $|$ Cons α $(\alpha$ list$)$

Theorems: Distinctness, injectivity, exhaustiveness, induction

Codatatypes

Inductive (or algebraic) datatypes:

datatype α list $=$ Nil $|$ Cons α $(\alpha$ list$)$

Theorems: Distinctness, injectivity, exhaustiveness, induction

Coinductive (or coalgebraic) datatypes:

codatatype α llist $=$ LNil $|$ LCons α $(\alpha$ llist$)$

Codatatypes

Inductive (or algebraic) datatypes:

datatype α list $=$ Nil $|$ Cons α $(\alpha$ list$)$

Theorems: Distinctness, injectivity, exhaustiveness, induction

Coinductive (or coalgebraic) datatypes:

codatatype α llist $=$ LNil $|$ LCons α $(\alpha$ llist$)$

codatatype α stream $=$ SCons α $(\alpha$ stream$)$

Codatatypes

Inductive (or algebraic) datatypes:

datatype α list $=$ Nil $|$ Cons α $(\alpha$ list$)$

Theorems: Distinctness, injectivity, exhaustiveness, induction

Coinductive (or coalgebraic) datatypes:

codatatype α llist $=$ LNil $|$ LCons α $(\alpha$ llist$)$

codatatype α stream $=$ SCons α $(\alpha$ stream$)$

Theorems: Distinctness, injectivity, exhaustiveness, coinduction

Syntax and Semantics

datatype fmla $=$ Atm atom $|$ Neg fmla $|$ Conj fmla fmla $|$ All var fmla

Syntax and Semantics

datatype fmla $=$ Atm atom $|$ Neg fmla $|$ Conj fmla fmla $|$ All var fmla

Terms	$[\![x]\!]_\xi^{\mathscr{S}}$	$=$	$\xi\ x$
	$[\![f(t_1, \ldots, t_n)]\!]_\xi^{\mathscr{S}}$	$=$	$F_f\left([\![t_1]\!]_\xi^{\mathscr{S}}, \ldots, [\![t_n]\!]_\xi^{\mathscr{S}}\right)$
Atoms	$\mathscr{S} \models_\xi p(t_1, \ldots, t_n)$	$=$	$P_p\left([\![t_1]\!]_\xi^{\mathscr{S}}, \ldots, [\![t_n]\!]_\xi^{\mathscr{S}}\right)$

Syntax and Semantics

datatype fmla = Atm atom | Neg fmla | Conj fmla fmla | All var fmla

Terms

$$[\![x]\!]^{\mathscr{S}}_{\xi} = \xi\ x$$

$$[\![f(t_1, \ldots, t_n)]\!]^{\mathscr{S}}_{\xi} = F_f\left([\![t_1]\!]^{\mathscr{S}}_{\xi}, \ldots, [\![t_n]\!]^{\mathscr{S}}_{\xi}\right)$$

Atoms

$$\mathscr{S} \models_{\xi} p(t_1, \ldots, t_n) = P_p\left([\![t_1]\!]^{\mathscr{S}}_{\xi}, \ldots, [\![t_n]\!]^{\mathscr{S}}_{\xi}\right)$$

Formulas

$$\mathscr{S} \models_{\xi} \text{Atm}\ a = \mathscr{S} \models_{\xi} a$$

$$\mathscr{S} \models_{\xi} \text{Neg}\ \varphi = \mathscr{S} \not\models_{\xi} \varphi$$

$$\mathscr{S} \models_{\xi} \text{Conj}\ \varphi\ \psi = \mathscr{S} \models_{\xi} \varphi\ \wedge\ \mathscr{S} \models_{\xi} \psi$$

$$\mathscr{S} \models_{\xi} \text{All}\ x\ \varphi = \forall a \in S.\ \mathscr{S} \models_{\xi[x \leftarrow a]} \varphi$$

A Gentzen System

$$\frac{}{\Gamma, \mathsf{Atm}\ a \vdash \Delta, \mathsf{Atm}\ a}\ \textsc{Ax} \qquad \frac{\Gamma \vdash \Delta, \varphi}{\Gamma, \mathsf{Neg}\ \varphi \vdash \Delta}\ \textsc{NegL} \qquad \frac{\Gamma, \varphi \vdash \Delta}{\Gamma \vdash \Delta, \mathsf{Neg}\ \varphi}\ \textsc{NegR}$$

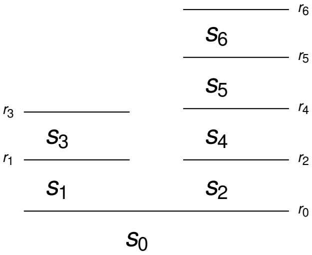
$$\frac{\Gamma, \varphi, \psi \vdash \Delta}{\Gamma, \mathsf{Conj}\ \varphi\ \psi \vdash \Delta}\ \textsc{ConjL} \qquad \frac{\Gamma \vdash \Delta, \varphi \qquad \Gamma \vdash \Delta, \psi}{\Gamma \vdash \Delta, \mathsf{Conj}\ \varphi\ \psi}\ \textsc{ConjR}$$

$$\frac{\Gamma, \mathsf{All}\ x\ \varphi, \varphi[t/x] \vdash \Delta}{\Gamma, \mathsf{All}\ x\ \varphi \vdash \Delta}\ \textsc{AllL} \qquad \frac{\Gamma \vdash \Delta, \varphi[y/x]}{\Gamma \vdash \Delta, \mathsf{All}\ x\ \varphi}\ \begin{array}{l}\textsc{AllR} \\ (y \text{ fresh})\end{array}$$

Abstracting Away

$$
\dfrac{
  \dfrac{
    \dfrac{}{\forall x.\, p(x),\ p(y) \vdash p(y)}\ \textsc{Ax}_{p(y)}
  }{\forall x.\, p(x) \vdash p(y)}\ \textsc{AllL}_{x,p(x),y}
  \qquad
  \dfrac{
    \dfrac{
      \dfrac{
        \dfrac{}{\forall x.\, p(x),\ p(z) \vdash p(z)}\ \textsc{Ax}_{p(z)}
      }{\forall x.\, p(x),\ p(y) \vdash p(z)}\ \textsc{AllL}_{x,p(x),z}
    }{\forall x.\, p(x),\ p(x) \vdash p(z)}\ \textsc{AllL}_{x,p(x),y}
  }{\forall x.\, p(x) \vdash p(z)}\ \textsc{AllL}_{x,p(x),x}
}{\forall x.\, p(x) \vdash p(y) \land p(z)}\ \textsc{ConjR}_{p(y),\,p(z)}
$$

Abstracting Away



Abstracting Away

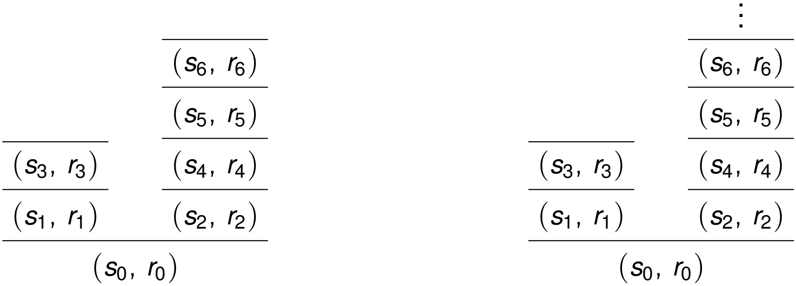
$$\dfrac{\dfrac{\dfrac{}{(s_3,\ r_3)}}{(s_1,\ r_1)} \qquad \dfrac{\dfrac{\dfrac{\dfrac{}{(s_6,\ r_6)}}{(s_5,\ r_5)}}{(s_4,\ r_4)}}{(s_2,\ r_2)}}{(s_0,\ r_0)}$$

Abstracting Away

$$\dfrac{\dfrac{\dfrac{}{(s_3,\ r_3)}}{(s_1,\ r_1)} \qquad \dfrac{\dfrac{\dfrac{\dfrac{}{(s_6,\ r_6)}}{(s_5,\ r_5)}}{(s_4,\ r_4)}}{(s_2,\ r_2)}}{(s_0,\ r_0)}$$

$$\dfrac{\textsc{AllL}_{x,p(x),y}\ \dfrac{\textsc{Ax}_{p(y)}\ \dfrac{}{\forall x.\, p(x),\ p(y) \vdash p(y)}}{\forall x.\, p(x) \vdash p(y)} \qquad \dfrac{\dfrac{\dfrac{\dfrac{\vdots}{\forall x.\, p(x),\ p(y) \vdash p(z)}\ \textsc{Ax}_{p(z)}}{\forall x.\, p(x),\ p(y) \vdash p(z)}\ \textsc{AllL}_{x,p(x),y}}{\forall x.\, p(x),\ p(y) \vdash p(z)}\ \textsc{AllL}_{x,p(x),y}}{\forall x.\, p(x) \vdash p(z)}\ \textsc{AllL}_{x,p(x),y}}{\forall x.\, p(x) \vdash p(y) \wedge p(z)}\ \textsc{ConjR}_{p(y),\, p(z)}$$

Abstracting Away



Abstracting Away

$$\dfrac{\dfrac{\dfrac{}{(s_3,\ r_3)}}{(s_1,\ r_1)} \qquad \dfrac{\dfrac{\dfrac{\dfrac{\vdots}{(s_6,\ r_6)}}{(s_5,\ r_5)}}{(s_4,\ r_4)}}{(s_2,\ r_2)}}{(s_0,\ r_0)}$$

Abstracting Away

eff : rule $\rightarrow$ state $\rightarrow$ state fset $\rightarrow$ bool   enabled $r\ s = (\exists ss.\ \text{eff}\ r\ s\ ss)$

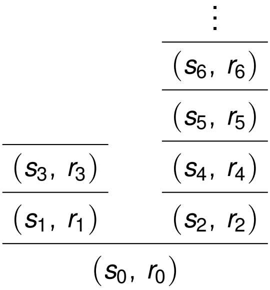
$$\dfrac{\dfrac{\dfrac{}{(s_3,\ r_3)}}{(s_1,\ r_1)} \qquad \dfrac{\dfrac{\dfrac{\dfrac{\vdots}{(s_6,\ r_6)}}{(s_5,\ r_5)}}{(s_4,\ r_4)}}{(s_2,\ r_2)}}{(s_0,\ r_0)}$$

Abstracting Away

eff : rule $\rightarrow$ state $\rightarrow$ state fset $\rightarrow$ bool   enabled $r\ s = (\exists ss.\ \text{eff}\ r\ s\ ss)$

Available $\forall s.\ \exists r.\ \text{enabled}\ r\ s$

Persistent $\forall s, r, r', s', ss.\ \text{enabled}\ r'\ s \wedge r' \neq r \wedge$
$\text{eff}\ r\ s\ ss \wedge s' \in \text{set}\ ss \Rightarrow \text{enabled}\ r'\ s'$



Abstracting Away

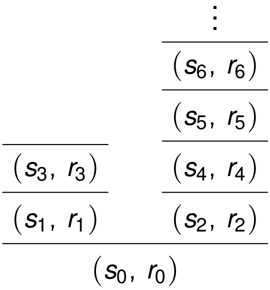
eff : rule $\rightarrow$ state $\rightarrow$ state fset $\rightarrow$ bool $\qquad$ enabled $r\ s = (\exists ss.\ \text{eff}\ r\ s\ ss)$

Available $\forall s.\ \exists r.\ \text{enabled}\ r\ s$

Persistent $\forall s, r, r', s', ss.\ \text{enabled}\ r'\ s \wedge r' \neq r \wedge$

$\text{eff}\ r\ s\ ss \wedge s' \in \text{set}\ ss \Rightarrow \text{enabled}\ r'\ s'$

codatatype tree $=$ Node (state $\times$ rule) (tree fset)



Abstracting Away

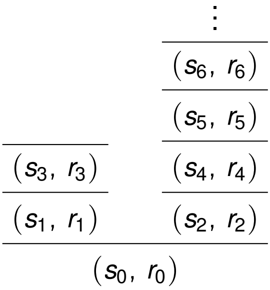
$\text{eff} : \text{rule} \to \text{state} \to \text{state fset} \to \text{bool} \qquad \text{enabled } r\ s = (\exists ss.\ \text{eff } r\ s\ ss)$

Available $\forall s.\ \exists r.\ \text{enabled } r\ s$

Persistent $\forall s, r, r', s', ss.\ \text{enabled } r'\ s \land r' \neq r \land$
$\text{eff } r\ s\ ss \land s' \in \text{set } ss \Rightarrow \text{enabled } r'\ s'$

codatatype tree = Node (state × rule) (tree fset)

$$\frac{\text{eff } r\ s\ ss \quad s' \in ss \quad \text{epath } (\text{SCons } (s', r')\ \sigma)}{\text{epath } (\text{SCons } (s, r)\ (\text{SCons } (s', r')\ \sigma))}$$



Abstracting Away

$\text{eff} : \text{rule} \to \text{state} \to \text{state fset} \to \text{bool} \qquad \text{enabled } r\ s = (\exists ss.\ \text{eff } r\ s\ ss)$

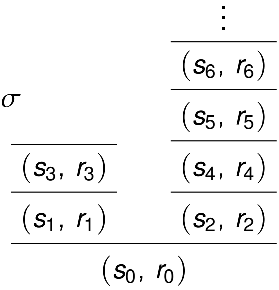
Available $\forall s.\ \exists r.\ \text{enabled } r\ s$

Persistent $\forall s, r, r', s', ss.\ \text{enabled } r'\ s \wedge r' \neq r \wedge$
$\text{eff } r\ s\ ss \wedge s' \in \text{set } ss \Rightarrow \text{enabled } r'\ s'$

codatatype tree $=$ Node $(\text{state} \times \text{rule})$ (tree fset)

$$\frac{\text{eff } r\ s\ ss \qquad s' \in ss \qquad \text{epath } (\text{SCons } (s', r')\ \sigma)}{\text{epath } (\text{SCons } (s, r)\ (\text{SCons } (s', r')\ \sigma))}$$

Fair $\forall r.\ \text{ev } (\text{alw enabledAt}_r)\ \sigma \Rightarrow \text{alw } (\text{ev taken}_r)\ \sigma$



Theorem (Abstract Completeness)

Assume that the effect relation is available and persistent.
Then each state admits either a finite proof tree or a fair escape path.

Proof Idea.

$$\text{mkTree} : \text{rule stream} \to \text{state} \to \text{tree}$$

□

Theorem (Abstract Completeness)

Assume that the effect relation is available and persistent.
Then each state admits either a finite proof tree or a fair escape path.

Proof Idea.

$$\begin{aligned}
&\mathsf{mkTree} : \mathsf{rule\ stream} \to \mathsf{state} \to \mathsf{tree}\\
&\mathsf{mkTree}\ \rho\ s = \mathbf{let}\ \ \mathsf{SCons}\ r\ \rho' = \mathsf{sdropWhile}\ (\lambda r.\ \neg\ \mathsf{enabled}\ r\ s)\ \rho\\
&\qquad\qquad\qquad\quad\ \ ss = (\varepsilon ss.\ \mathsf{eff}\ r\ s\ ss)\\
&\qquad\qquad\qquad\ \mathbf{in}\ \ \mathsf{Node}\ (s, r)\ (\mathsf{image}\ (\mathsf{mkTree}\ \rho')\ ss)
\end{aligned}$$

□

Our CASC-25 Submission

```
data Stream a = SCons a (Stream a)
newtype FSet a = FSet [a]
data Tree a = Node a (FSet (Tree a))

fmap f (FSet xs) = FSet (map f xs)

sdropWhile p (SCons a s) =
  if p a then sdropWhile p s else SCons a s

mkTree eff rho s =
  Node (s, r) (fmap (mkTree eff rho’) (fromJust (eff r s)))
  where SCons r rho’ = sdropWhile (\r -> not (isJust (eff r s))) rho
```

Our CASC-25 Submission

```haskell
data Stream a = SCons a (Stream a)
newtype FSet a = FSet [a]
data Tree a = Node a (FSet (Tree a))

fmap f (FSet xs) = FSet (map f xs)

sdropWhile p (SCons a s) =
  if p a then sdropWhile p s else SCons a s

mkTree eff rho s =
  Node (s, r) (fmap (mkTree eff rho’) (fromJust (eff r s)))
  where SCons r rho’ = sdropWhile (\r -> not (isJust (eff r s))) rho
```

Conclusion

Our complete abstract proof:

- 425 lines of Isabelle ($\approx$ 8 pages)
- Rigorous
- Formal (mechanized)
- Coinductive

Conclusion

Our complete abstract proof:

- 425 lines of Isabelle ($\approx$ 8 pages)
- Rigorous
- Formal (mechanized)
- Coinductive

A key part of “mechanized metatheory of Sledgehammer”

Conclusion

Our complete abstract proof:

- 425 lines of Isabelle ($\approx$ 8 pages)
- Rigorous
- Formal (mechanized)
- Coinductive

A key part of “mechanized metatheory of Sledgehammer”

Future work:

- Completeness for resolution, superposition, . . . ?

Evangelizing presupposes a desire in the Church to come out of herself. . . . When the Church does not come out of herself to evangelize, she becomes self-referential and then gets sick. The evils that, over time, happen in ecclesial institutions have their root in self-referentiality and a kind of theological narcissism.

Jorge Mario Bergoglio