HOL and Isabelle/HOL
— Warm-Up —

Christoph Benzmüller and Luca Pasetto

ESSLLI 2026

First steps in Isabelle/HOL using LLMs

— Demo —

Liars Street

— Demo —

CAN A COMPUTER CRACK THIS RIDDLE?

On Liar's Street everyone always lies; on Truthteller's Road everyone always tells the truth. From just what Nilda and Carla say, our software works out who lives where — by reasoning, not guessing.

THE RIDDLE

LIAR'S STREET
everyone here always lies

TRUTHTELLER'S ROAD
everyone here tells the truth

Nilda says: *"Carla lives on Liar's Street."*

Carla says: *"Neither of us lives on Liar's Street."*

WHAT THE COMPUTER ANSWERS

Nitpick searched every possible world — and found exactly ONE that fits:

TRUTHTELLER'S ROAD  **Nilda**

LIAR'S STREET  **Carla**

The computer can answer in three ways:

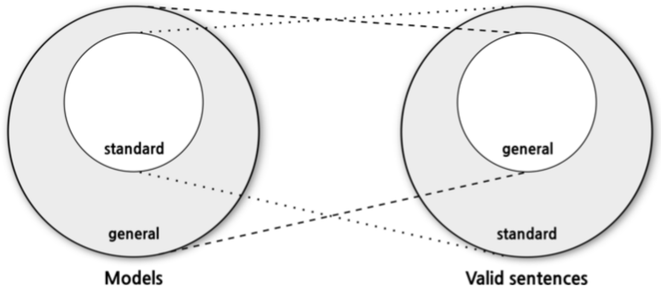
1 One answer — solved!	**∞** Many answers — ambiguous	**0** No answer — paradox

Surjective Cantor Theorem

— Demo —

HOL: Well Understood Semantics

HOL with standard semantics: incomplete
HOL with Henkin's general semantics: semi-decidable & compact



more model structures . . . fewer valid formulas

Important principles are still valid in Henkin's general models:

- Comprehension (type-restricted): $\forall G\, \exists F\, \forall \overline{X^n}\, F\overline{X^n} = G$
- Boolean Extensionality: $\forall P\, \forall Q\, ((P \leftrightarrow Q) \rightarrow P = Q)$
- Functional Extensionality: $\forall F\, \forall G\, ((\forall X\, F\ X = G\ X) \rightarrow F = G)$

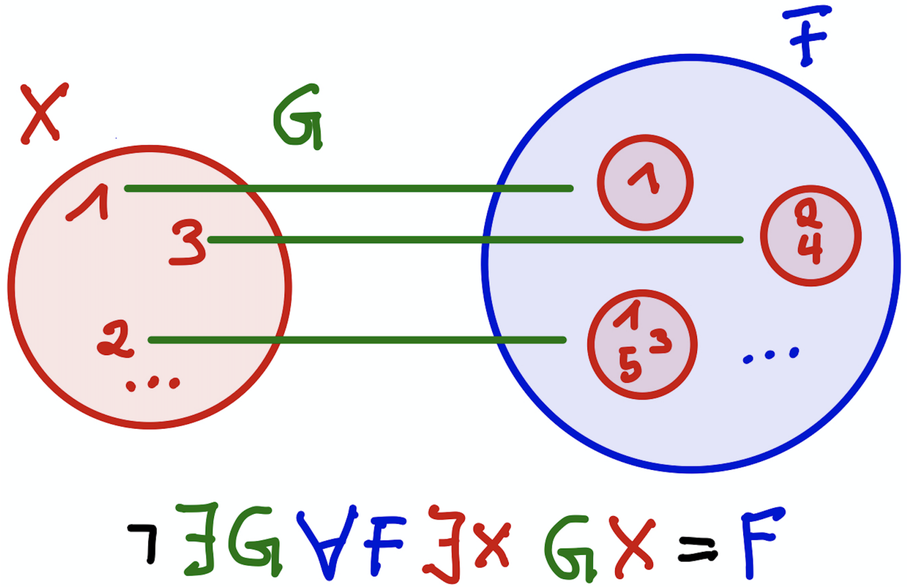
Note: Any "Henkin-valid" formula is also valid in standard semantics!

Suggested Reading

- Origin [Church, JSL, 1940]
- Henkin's general semantics: [Henkin, JSL, 1950] [Andrews, JSL, 1971, 1972]
- Extensionality&Intensionality: [BenzmüllerEtAl., JSL, 2004] [Muskens, JSL, 2007] [PhD thesis by Steen]

HOL: Expressivity Matters — Surjective Cantor Theorem —

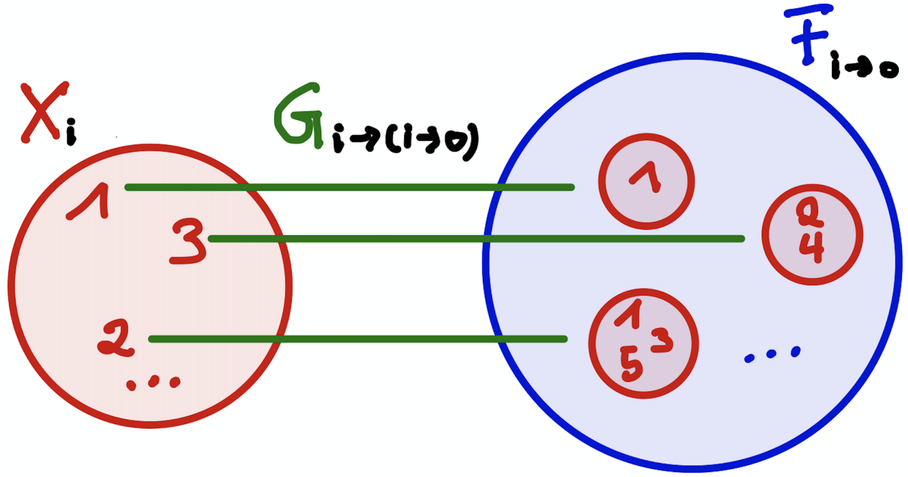
Theorem: *There is no surjective map from a set to its powerset.*



$$\neg \exists G\, \forall F\, \exists x\, G\, x = F$$

HOL: Expressivity Matters — Surjective Cantor Theorem —

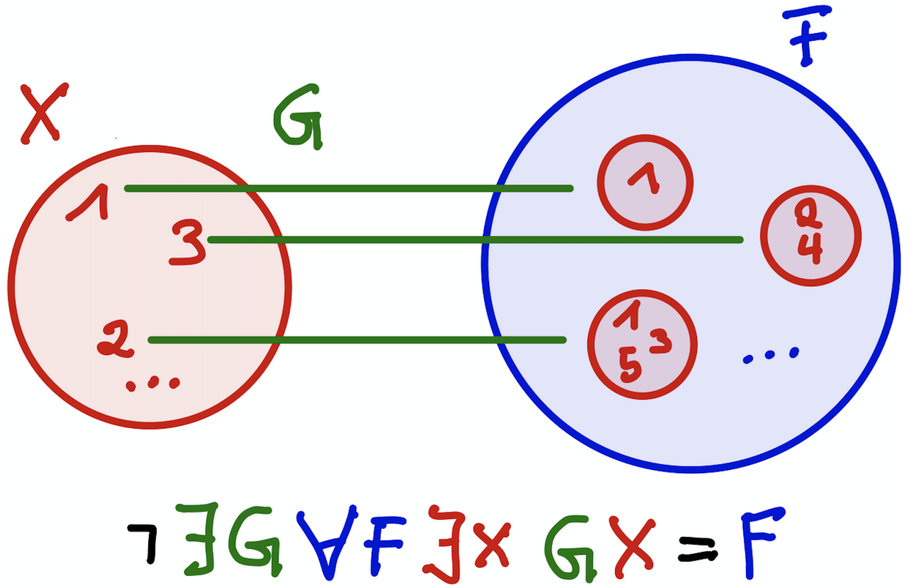
Theorem: *There is no surjective map from a set to its powerset.*



$$\neg\exists G_{i\to(i\to o)}\,\forall F_{i\to o}\,\exists X_i\; G\,X = F$$

HOL: Expressivity Matters — Surjective Cantor Theorem —

Theorem: *There is no surjective map from a set to its powerset.*



Proof (by contradiction):

Assume there is a surjective G.

Consider the following choice for F: $F := \{X \mid \neg X \in (G\ X)\}$

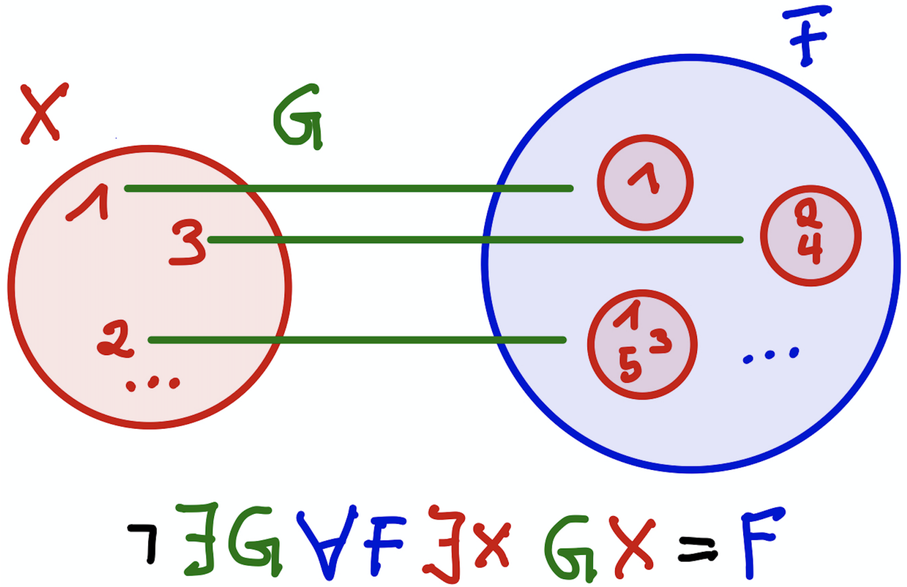
Then there is an a such that $(G\ a) = F$ (since G is surj. by ass.)

But then: $a \in (G\ a)$ iff $a \in F$ iff $\neg a \in (G\ a)$ (by def. of F)

Contradiction. □

HOL: Expressivity Matters — Surjective Cantor Theorem —

Theorem: *There is no surjective map from a set to its powerset.*



HOL ATPs can solve this problem very efficiently

In our **Leo provers** this works by a combination of (heuristic) **guessing** and straightforward **resolution and HO unification**:

"¬" in $F := \{X \mid \neg X \in (G\ X)\}$ is **guessed**, rest is straightforward

Further reading: [AndrewsEtAl., Automating higher-order logic, 1984]

HOL: Example Formula — Surjective Cantor Theorem —

Theorem: $\neg\exists G_{i\to(i\to o)}\forall F_{i\to o}\exists X_i\, G\ X = F$

represented in HOL (as defined on previous slide) as

$$\neg\neg\Pi(\lambda G_{i\to(i\to o)}\ \neg(\Pi(\lambda F_{i\to o}\ \neg\Pi(\lambda X_i\ \neg G\ X = F))))$$

This is e.g. also the *internal representation* in our Leo provers.

HOL: Example Formula — Surjective Cantor Theorem —

Theorem: $\neg\exists G_{i\to(i\to o)}\forall F_{i\to o}\exists X_i\, G\ X = F$

represented in HOL (as defined on previous slide) as

$$\neg\neg\Pi(\lambda G_{i\to(i\to o)}\ \neg(\Pi(\lambda F_{i\to o}\ \neg\Pi(\lambda X_i\ \neg G\ X = F))))$$

This is e.g. also the *internal representation* in our Leo provers.

However, *intuitive user representations* are supported in HOL provers (e.g. in the Isabelle/HOL system):

```
lemma SurjectiveCantor: "¬(∃G::i⇒i⇒bool.∀F.∃X. G X = F)"
```

HOL: Example Formula — Surjective Cantor Theorem —

Theorem: $\neg\exists G_{i\to(i\to o)}\forall F_{i\to o}\exists X_i\, G\, X = F$

represented in HOL (as defined on previous slide) as

$$\neg\neg\Pi(\lambda G_{i\to(i\to o)}\ \neg(\Pi(\lambda F_{i\to o}\ \neg\Pi(\lambda X_i\ \neg G\, X = F))))$$

This is e.g. also the *internal representation* in our Leo provers.

However, *intuitive user representations* are supported in HOL provers (e.g. in the Isabelle/HOL system):

```
lemma SurjectiveCantor: "¬(∃G::i⇒i⇒bool.∀F.∃X. G X = F)"

        proof
          assume  1: "∃G::i⇒i⇒bool.∀F.∃X. G X = F"
          obtain  g::"i⇒i⇒bool" where 2: "∀F.∃X. g X = F" using 1 by blast
          let ?F = "λX. ¬ g X X" (* choose F = {x|¬(x∈(g x))} *)
          have    3: "∃Y. ?F = g Y" using 2 by metis
          obtain a::i where 4: "?F = g a" using 3 by blast
          have    5: "g a a ⟷ ?F a" using 4 by metis
          have    6: "g a a ⟷ ¬g a a" using 5 by metis
          show False using 6 by blast
        qed
```

Boolos' Curious Inference

— Demo —

The Space of Proofs

On (hyper-)exponentially smaller/larger proofs:

When passing from first-order to higher-order logic (more generally: from n-th order to n+1-th order)

Classic references:

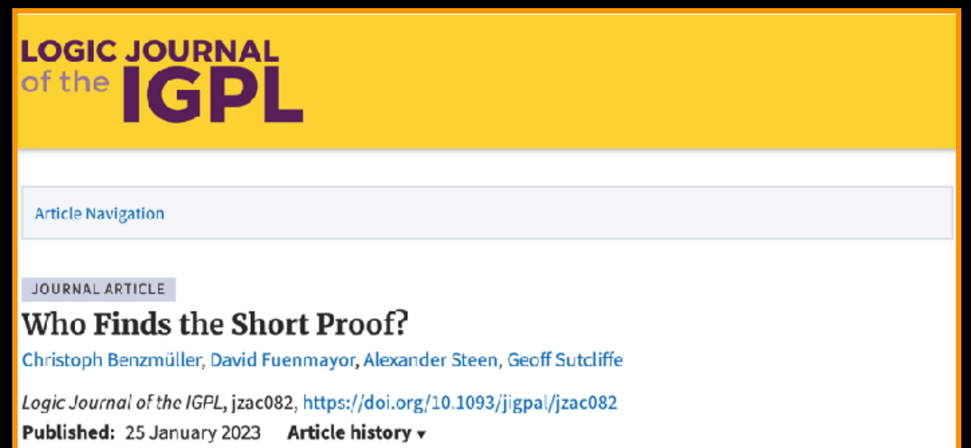
- - Gödel (1936) “Über die Länge von Beweisen” (claim)
- - Buss (1994-95) “On Gödel's theorems on lengths of proofs. I-II” (proof)
- … more

Who finds the short proof?

Theorem Proving in HOL is undecidable (semi-decidable for general semantics) and hence often considered unsuited for practical applications.

E.g. the KR&R community typically puts a strong focus on less expressive logics like FOL or decidable fragments of FOL.

With this work we wanted to challenge and attack this position with the help of a thought experiment.



Two Logicians at the Gates of Heaven

If you want to go to heaven, then better pick a HOL theorem prover

Who finds the short proof?

If you want to go to heaven, then better pick a HOL theorem prover

HOLly and FOLbert

- arrive at the gates of heaven and both want to enter
- but there is space only for one of them

ATP contest is set up to find a winner

- both are asked to choose a FOL problem an pose it to the other
- both are asked to select either a FOL ATP or a HOL ATP
- whose ATP solves its problem first wins
- FOLbert chooses a FOL ATP
- HOLly chooses a HOL ATP
- no winner by midnight: contest starts over next day

Only HOLly may win one day, provided

- she chooses a clever FOL problem for FOLbert to solve

HOL: Expressivity Matters — Boolos' Curius Inference —

JOURNAL ARTICLE

A Curious Inference

George Boolos

Journal of Philosophical Logic
Vol. 16, No. 1 (Feb., 1987), pp. 1-12 (12 pages)

Published by: Springer

1. $\forall n\, f(n,1) = s(1)$
2. $\forall x\, f(1,s(x)) = s(s(f(1,x)))$
3. $\forall n \forall x\, f(s(n),s(x)) = f(n,f(s(n),x))$
4. $D(1)$
5. $\forall x\, D(x) \rightarrow D(s(x))$

   $\therefore$
6. $D(f(s(s(s(s(1))))),s(s(s(s(1))))))$

Ackermann-Function f

f(5,5) is a very-big-num

D(1)
D(s(1))
D(s(s(1)))
D(s(s(s(1))))
…
D(very-big-num)

[George Boolos, A curious inference, J.Phil.Log., 16:1-12, 1987]

Proof in FOL? — yes, **but practically infeasible** number of proof steps

Proof in HOL? — yes, on **one single page**

- Key: powerful lemmata as instances of comprehension axioms

See our interactive verification of Boolos's proof:

[BenzmüllerBrown, The curious inference of Boolos in MIZAR and OMEGA, 2007]

HOL: Expressivity Matters — Boolos' Curius Inference —


JOURNAL ARTICLE

A Curious Inference

George Boolos

Journal of Philosophical Logic
Vol. 16, No. 1 (Feb., 1987), pp. 1-12 (12 pages)
Published by: Springer

1. $\forall n\, f(n,1) = s(1)$
2. $\forall x\, f(1,s(x)) = s(s(f(1,x)))$
3. $\forall n \forall x\, f(s(n),s(x)) = f(n,f(s(n),x))$
4. $D(1)$
5. $\forall x\, D(x) \rightarrow D(s(x))$

   $\therefore$
6. $D(f(s(s(s(s(1))))),s(s(s(s(1))))))$

Ackermann-Function f

f(5,5) is a very-big-num

D(1)
D(s(1))
D(s(s(1)))
D(s(s(s(1))))
…
D(very-big-num)

[George Boolos, A curious inference, J.Phil.Log., 16:1-12, 1987]

Proof in FOL? — yes, **but practically infeasible** number of proof steps

Proof in HOL? — yes, on **one single page**

- Key: powerful lemmata as instances of comprehension axioms

See our interactive verification of Boolos's proof:

[BenzmüllerBrown, The curious inference of Boolos in MIZAR and OMEGA, 2007]

Proof has recently been put into Isabelle/HOL: [Kettland, AFP, 2022]

Boolos Curious Inference

JOURNAL ARTICLE

A Curious Inference

George Boolos

Journal of Philosophical Logic
Vol. 16, No. 1 (Feb., 1987), pp. 1-12 (12 pages)

Published by: Springer

STUDIES IN LOGIC, GRAMMAR AND RHETORIC 10 (23) 2007

The Curious Inference of Boolos in Mizar and OMEGA*

Christoph E. Benzmüller[1,2] and Chad E. Brown[2]

[1]The University of Cambridge, Cambridge, UK
[2]Universität des Saarlandes, Saarbrücken, Germany
chris|cebrown@ags.uni-sb.de

To Andrzej Trybulec

Abstract. We examine Boolos' curious inference and formalize it in a system based on set theory (Mizar) and a system based on classical higher-order logic (OMEGA). The Boolos example is interesting because while it can in principle be proven using a complete first-order calculus, it is impractical to do so. In our case study we are interested in aspects such as how natural and at what level of granularity Boolos' short second-order proof sketch can be formalized in Mizar and OMEGA.

Proof has recently been put also into Isabelle/HOL:
[Kettland, AFP, 2022]

Abbreviations

We first present a sketch of a second-order derivation of (6) from (1)–(5) of which a complete formalization in any standard axiomatic formulation of second-order logic can easily be written out:

By the comprehension principle of second-order logic, $\exists N \forall z(Nz \leftrightarrow \forall X[X1 \,\&\, \forall y(Xy \rightarrow Xsy) \rightarrow Xz])$, and then for some N, $\exists E \forall z(Ez \leftrightarrow Nz \,\&\, Dz)$.

LEMMA 1: $N1$; $\forall y(Ny \rightarrow Nsy)$; $Nssss1$; $E1$; $\forall y(Ey \rightarrow Esy)$; $Es1$.

LEMMA 2: $\forall n(Nn \rightarrow \forall x(Nx \rightarrow Efnx))$.

Proof. By comprehension, $\exists M \forall n(Mn \leftrightarrow \forall x(Nx \rightarrow Efnx))$. We want $\forall n(Nn \rightarrow Mn)$. Enough to show $M1$ and $\forall n(Mn \rightarrow Msn)$, for then if Nn, Mn.

$M1$: Want $\forall x(Nx \rightarrow Ef1x)$. By comprehension, $\exists Q \forall x(Qx \leftrightarrow Ef1x)$. Want $\forall x(Nx \rightarrow Qx)$. Enough to show $Q1$ and $\forall x(Qx \rightarrow Qsx)$.

$Q1$: Want $Ef11$. But $f11 = s1$ by (1) and $Es1$ by Lemma 1.

$\forall x(Qx \rightarrow Qsx)$: Suppose Qx, i.e. $Ef1x$. By (2) $f1sx = ssf1x$; by Lemma 1 twice, $Ef1sx$. Thus Qsx and $M1$.

$\forall n(Mn \rightarrow Msn)$: Suppose Mn, i.e. $\forall x(Nx \rightarrow Efnx)$. Want Msn, i.e. $\forall x(Nx \rightarrow Efsnx)$. By comprehension, $\exists P \forall x(Px \leftrightarrow Efsnx)$. Want $\forall x(Nx \rightarrow Px)$. Enough to show $P1$ and $\forall x(Px \rightarrow Psx)$.

$P1$: Want $Efsn1$. But $fsn1 = s1$ by (1) and $Es1$ by Lemma 1.

$\forall x(Px \rightarrow Psx)$: Suppose Px, i.e. $Efsnx$; thus $Nfsnx$. Want $Efsnsx$. Since $Nfsnx$ and Mn, $Efnfsnx$. But by (3) $fnfsnx = fsnsx$; thus $Efsnsx$. By Lemma 1, $Nssss1$. By Lemma 2, $Efssss1ssss1$. Thus $Dfssss1ssss1$, as desired.

Screenshot

New(!): Automating Boolos Curious Inference

```
theory BoolosATP imports Main
begin

typedecl i
consts
 e :: "i"   (* one *)
 s :: "i⇒i"   (* successor function *)
 F :: "i⇒i⇒i"   (* binary function; axiomatised below as Ackermann function *)
 D :: "i⇒bool"   (* arbitrary uninterpreted unary predicate *)

axiomatization where
    A1: "∀n. F n e = s e"   (* Axiom for Ackermann function F *)
and A2: "∀y. F e (s y) = s (s (F e y))"   (* Axiom for Ackermann function F *)
and A3: "∀x y. F (s x) (s y) = F x (F (s x) y)"   (* Axiom for Ackermann function F *)
and A4: "D e"   (* D holds for one *)
and A5: "∀x. D x ⟶ D (s x)"   (* if D holds for x it also holds for the successor of x *)

lemma "D (F (s (s (s (s e)))) (s (s (s (s e)))))" sledgehammer oops (* no proof; hopeless! *)

```

LOGIC JOURNAL of the IGPL

Article Navigation

JOURNAL ARTICLE

Who Finds the Short Proof?

Christoph Benzmüller, David Fuenmayor, Alexander Steen, Geoff Sutcliffe

Logic Journal of the IGPL, jzac082, https://doi.org/10.1093/jigpal/jzac082

Published: 25 January 2023 **Article history**

Cite Permissions Share

Abstract

This paper reports on an exploration of Boolos' Curious Inference, using higher-order automated theorem provers (ATPs). Surprisingly, only suitable shorthand notations had to be provided by hand for ATPs to find a short proof. The higher-order lemmas required for constructing a short proof are automatically discovered by the ATPs. Given the observations and suggestions in this paper, full proof automation of Boolos' and related examples now seems to be within reach of higher-order ATPs.

Finding: only the invention of definitions is key — rest can be automated!

New(!): Automating Boolos Curious Inference

```
theory BoolosATP imports Main
begin

typedecl i
consts
 e :: "i"   (* one *)
 s :: "i⇒i"   (* successor function *)
 F :: "i⇒i⇒i"   (* binary function; axiomatised below as Ackermann function *)
 D :: "i⇒bool"   (* arbitrary uninterpreted unary predicate *)

axiomatization where
    A1: "∀n. F n e = s e"   (* Axiom for Ackermann function F *)
and A2: "∀y. F e (s y) = s (s (F e y))"   (* Axiom for Ackermann function F *)
and A3: "∀x y. F (s x) (s y) = F x (F (s x) y)"   (* Axiom for Ackermann function F *)
and A4: "D e"   (* D holds for one *)
and A5: "∀x. D x ⟶ D (s x)"   (* if D holds for x it also holds for the successor of x *)

lemma "D (F (s (s (s (s e)))) (s (s (s (s e)))))" sledgehammer oops (* no proof; hopeless! *)

definition isIndSet where "isIndSet X ≡ (X e) ∧ (∀x. X x ⟶ X (s x))"   (* X is inductive *)
definition N where "N x ≡ (∀X::i⇒bool. isIndSet X ⟶ X x)"   (* N is smallest inductive set *)
definition P where "P x y ≡ N (F x y)"   (* P(x,y) iff F(x,y) is in N *)

lemma "D (F (s (s (s (s e)))) (s (s (s (s e)))))"   (* ATPs can now proof this: using the Defs *)
  sledgehammer   (* proof found *)
  by (metis A1 A2 A3 A4 A5 N_def P_def isIndSet_def)   (* proof reconstruction succeeds *)
end
```

LOGIC JOURNAL of the IGPL

Article Navigation

JOURNAL ARTICLE

Who Finds the Short Proof?

Christoph Benzmüller, David Fuenmayor, Alexander Steen, Geoff Sutcliffe

Logic Journal of the IGPL, jzac082, https://doi.org/10.1093/jigpal/jzac082

Published: 25 January 2023 **Article history**

Cite Permissions Share

Abstract

This paper reports on an exploration of Boolos' Curious Inference, using higher-order automated theorem provers (ATPs). Surprisingly, only suitable shorthand notations had to be provided by hand for ATPs to find a short proof. The higher-order lemmas required for constructing a short proof are automatically discovered by the ATPs. Given the observations and suggestions in this paper, full proof automation of Boolos' and related examples now seems to be within reach of higher-order ATPs.

Short (and more or less legible) proofs were found by the ATPs:

cvc5 1.0,
E(hoh) 3.0,
Leo-III 1.7.0,
Vampire 4.7,
Zipperposition 2.1.,
Z3

Finding: only the invention of definitions is key — rest can be automated!

Automating Boolos Curious Inference

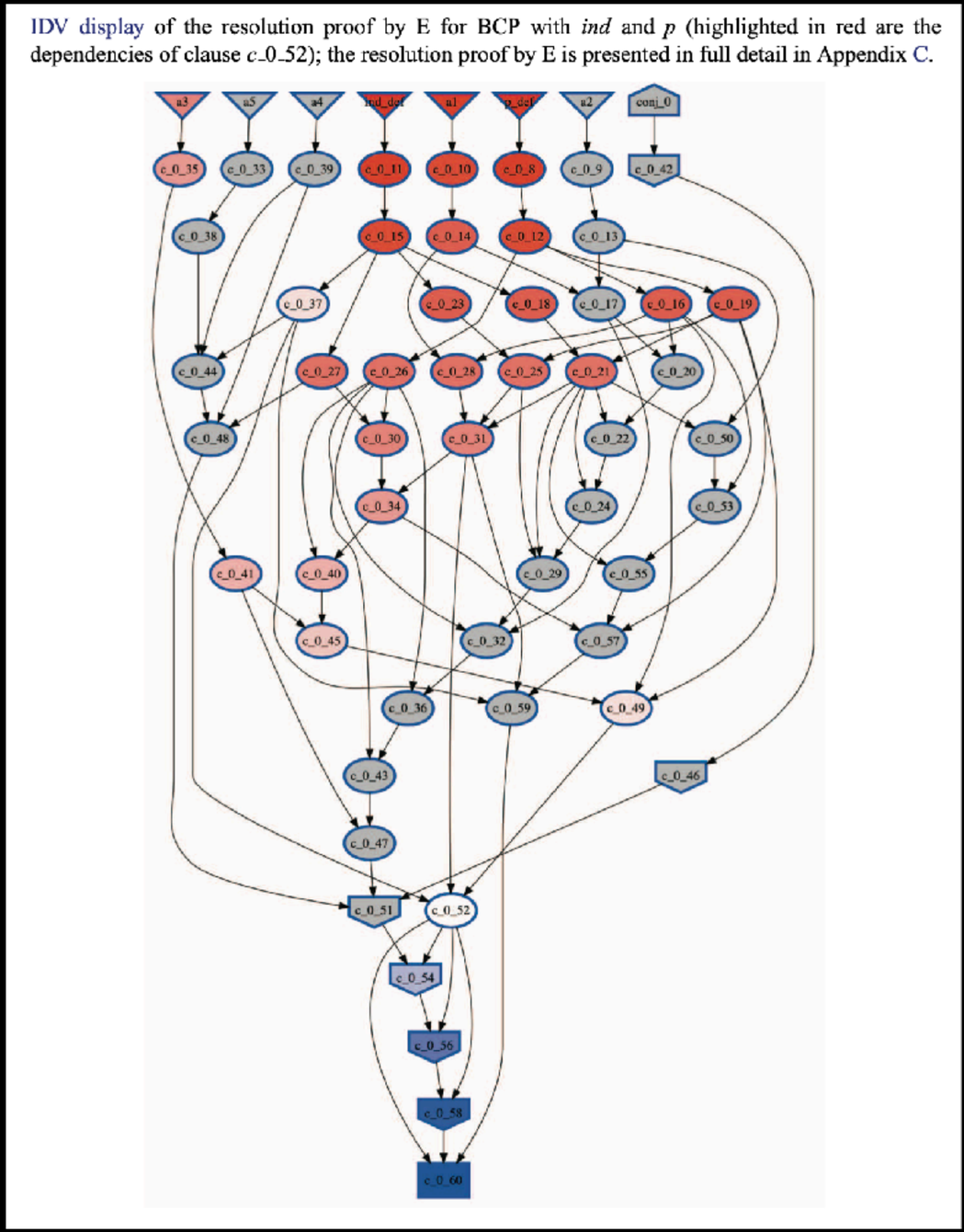
TPTP THF

https://github.com/cbenzmueller/BoolosCuriousInference-ATP/blob/main/Boolos1.p

```
% Declarations
thf(e_decl,type, e: $i ).
thf(s_decl,type, s: $i > $i ).
thf(d_decl,type, d: $i > $o ).
thf(f_decl,type, f: $i > $i > $i ).

% Auxiliary declarations
thf(ind_decl,type, ind: ( $i > $o ) > $o ).
thf(p_decl,type, p: $i > $i > $o ).

% Axioms
thf(a1,axiom,
    ! [N: $i] :
      ( ( f @ N @ e )
      = ( s @ e ) ) ).

thf(a2,axiom,
    ! [Y: $i] :
      ( ( f @ e @ ( s @ Y ) )
      = ( s @ ( s @ ( f @ e @ Y ) ) ) ) ).

thf(a3,axiom,
    ! [X: $i,Y: $i] :
      ( ( f @ ( s @ X ) @ ( s @ Y ) )
      = ( f @ X @ ( f @ ( s @ X ) @ Y ) ) ) ).

thf(a4,axiom,
    d @ e ).

thf(a5,axiom,
    ! [X: $i] :
      ( ( d @ X )
     => ( d @ ( s @ X ) ) ) ).

% Shorthand notations
thf(ind_def,axiom,
    ( ind
    = ( ^ [Q: $i > $o] :
          ( ( Q @ e )
          & ! [X: $i] :
              ( ( Q @ X )
             => ( Q @ ( s @ X ) ) ) ) ) ) ).

thf(p_def,axiom,
    ( p
    = ( ^ [X: $i,Y: $i] :
          ( ^ [X2: $i] :
              ! [X3: $i > $o] :
                ( ( ind @ X3 )
               => ( X3 @ X2 ) )
          @ ( f @ X @ Y ) ) ) ) ).

% Conjecture
thf(conj_0,conjecture,
    d @ ( f @ ( s @ ( s @ ( s @ ( s @ ( s @ e ) ) ) ) ) @ ( s @ ( s @ ( s @ ( s @ ( s @ e ) ) ) ) ) ) ).
```

Automating Boolos Curious Inference

IDV display of
proof by E 3.0



(E.g. clause C48 links to https://github.com/cbenzmueller/BoolosCuriousInference-ATP/blob/main/Boolos1.proof#L381-L384)

Part I

From A4, A5 and Def_*ind*, it follows that d is an inductive set:

$$C_{48} : ind\, d. \qquad \text{(L1a)}$$

From A1, Def_*ind* and Def_*p*, it follows that:

$$C_{31} : pxe \text{ for all } x. \qquad \text{(L1b)}$$

From A1, A2, Def_*ind* and Def_*p*, it follows that pe is an inductive set:

$$C_{59} : ind\, pe. \qquad \text{(L1c)}$$

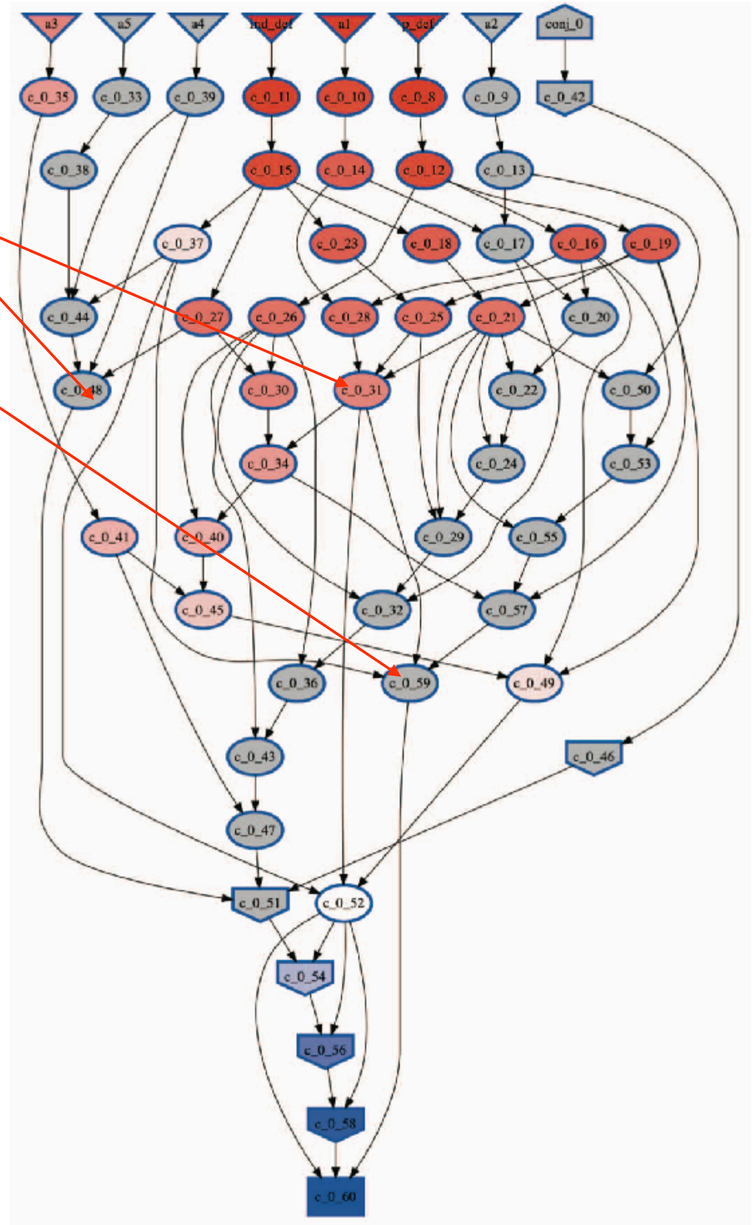
PROOF. Fully formal proofs are provided in Appendix C.

The proof of L1a (see the derivation of clause C_{48} from A4, A5 and Def_*ind*) is obvious.

Proof of L1b (see the derivation of clause C_{31} from axiom A1, Def_*ind* and Def_*p*): from Def_*ind* get $C_{23} : ind\, U \to Ue$ and $C_{18} : ind\, U \to (Ux \to Usx)$, for all U and x. Moreover, from Def_*p*, infer that for all x and y there exists a property qxy (depending on x and y) encoded by `epred1_2 @ X @ Y` in Appendix C, such that $C_{16} : qxy\,(fxy) \to pxy$ and $C_{19} : ind\, qxy \vee pxy$. From C_{19} and C_{23}, it follows $C_{25} : qxy\, e \vee pxy$, for all x and y. From C_{19} and C_{18}, for all x, y and z, get $C_{21} : (qxy\, z \to qxy\, sz) \vee pxy$. From A1 and C_{16}, get $C_{28} : qze\, se \to pze$, for all z. Finally, obtain L1b, i.e. clause C_{31}, from C_{28}, C_{25} and C_{21}.

Proof of L1c (see the derivation of clause C_{59} from A1, A2, Def_*ind* and Def_*p*): instantiating z with $s(fez')$ in C_{21} and applying the equation $fesz' = ssfez'$, which follows from A2, get $C_{50} : (qxy(sfez)' \to qxy(fesz')) \vee pxy$, for all z'. From this and C_{16}, obtain $C_{53} qesz'(sfez') \to pesz'$ and also $C_{55} : qesz'(fez') \to pesz'$, both for all z'. Now, from A1 and C_{16}, get $C_{28} : qze\, se \to pze$, for all z. Moreover, Def_*ind* implies that for all properties U, there exists some property kU (depending on U) encoded by `esk1_1 @ U` in Appendix C, such that $C_{27} : Ue \to (ind\, U \vee UkU)$ and $C_{37} : (Ue \wedge UskU) \to ind\, U$. From Def_*p*, it follows $C_{26} : (ind\, U \wedge pxy) \to Ufxy$, for all x, y and U. Using C_{26} and C_{27} get that for all x there exists $(k\, px)$ such that for all U holds $C_{30} : (ind\, U \wedge pxe) \to (ind\, px \vee Ufx(k\, px))$. Apply L1b to infer that for all x there exists $(k\, px)$ such that for all U holds $C_{34} : ind\, U \to (ind\, px \vee Ufx(k\, px))$. Using C_{55}, it then follows from C_{34} that $C_{57} : ind\, pe \vee pes(k\, pe)$. From C_{57} and C_{37} finally obtain L1c, i.e. clause C_{59}. □

IDV display of the resolution proof by E for BCP with *ind* and *p* (highlighted in red are the dependencies of clause *c*_0_52); the resolution proof by E is presented in full detail in Appendix C.



(E.g. clause C48 links to https://github.com/cbenzmueller/BoolosCuriousInference-ATP/blob/main/Boolos1.proof#L381-L384)

Part I

From A4, A5 and Def_*ind*, it follows that d is an inductive set:

$$C_{48} : ind\ d. \qquad \text{(L1a)}$$

From A1, Def_*ind* and Def_*p*, it follows that:

$$C_{31} : pxe \text{ for all } x. \qquad \text{(L1b)}$$

From A1, A2, Def_*ind* and Def_*p*, it follows that pe is an inductive set:

$$C_{59} : ind\ pe. \qquad \text{(L1c)}$$

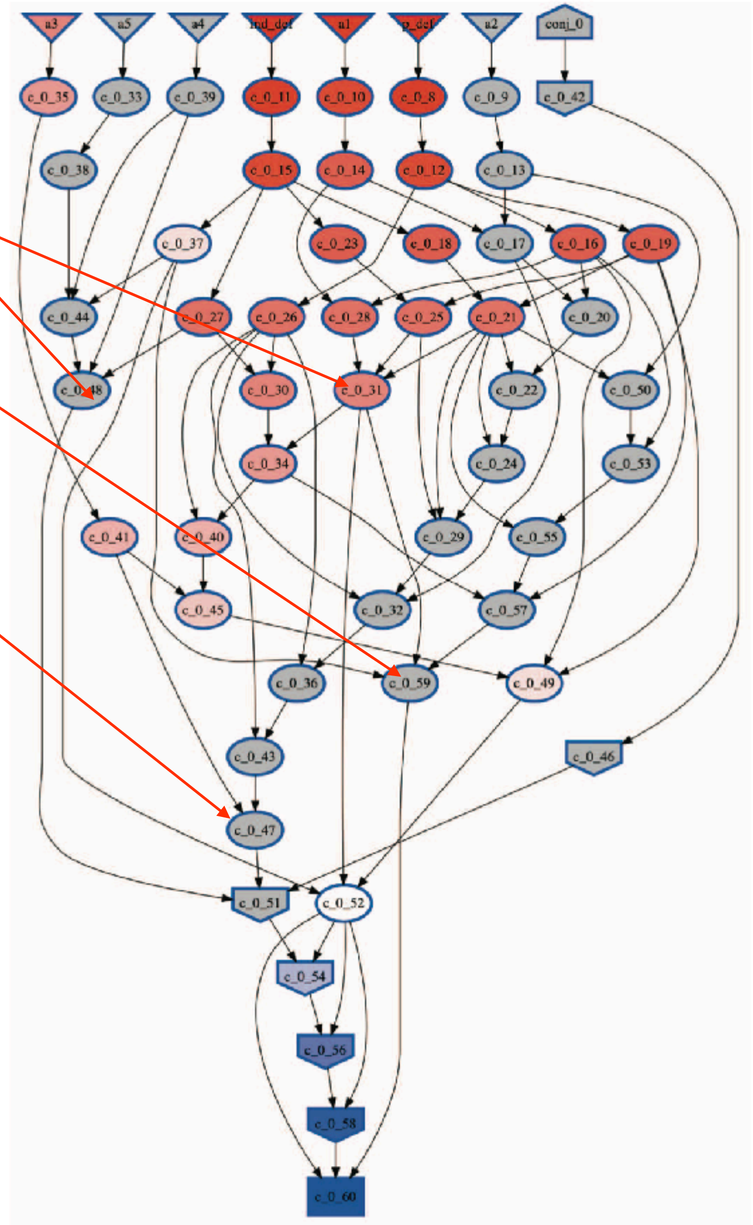
Part II

From A1, A2, A3, Def_*ind* and Def_*p*, it follows:

$$C_{47} : \forall x.\forall Y.\ (ind\ px \wedge ind\ psx \wedge ind\ Y) \rightarrow Y\ fsxsssse. \qquad \text{(L2)}$$

Informally, for all entities x and sets Y, if px, psx and Y are inductive sets (over e and s), then $f\ sx\ sssse$ is in Y.

IDV display of the resolution proof by E for BCP with *ind* and *p* (highlighted in red are the dependencies of clause *c*_0_52); the resolution proof by E is presented in full detail in Appendix C.



(E.g. clause C48 links to https://github.com/cbenzmueller/BoolosCuriousInference-ATP/blob/main/Boolos1.proof#L381-L384)

Part I
From A4, A5 and Def_*ind*, it follows that d is an inductive set:

$$C_{48} : ind\ d. \qquad \text{(L1a)}$$

From A1, Def_*ind* and Def_*p*, it follows that:

$$C_{31} : pxe \text{ for all } x. \qquad \text{(L1b)}$$

From A1, A2, Def_*ind* and Def_*p*, it follows that pe is an inductive set:

$$C_{59} : ind\ pe. \qquad \text{(L1c)}$$

Part II
From A1, A2, A3, Def_*ind* and Def_*p*, it follows:

$$C_{47} : \forall x.\forall Y.\ (ind\ px \wedge ind\ psx \wedge ind\ Y) \rightarrow Y\ fsxssssse. \qquad \text{(L2)}$$

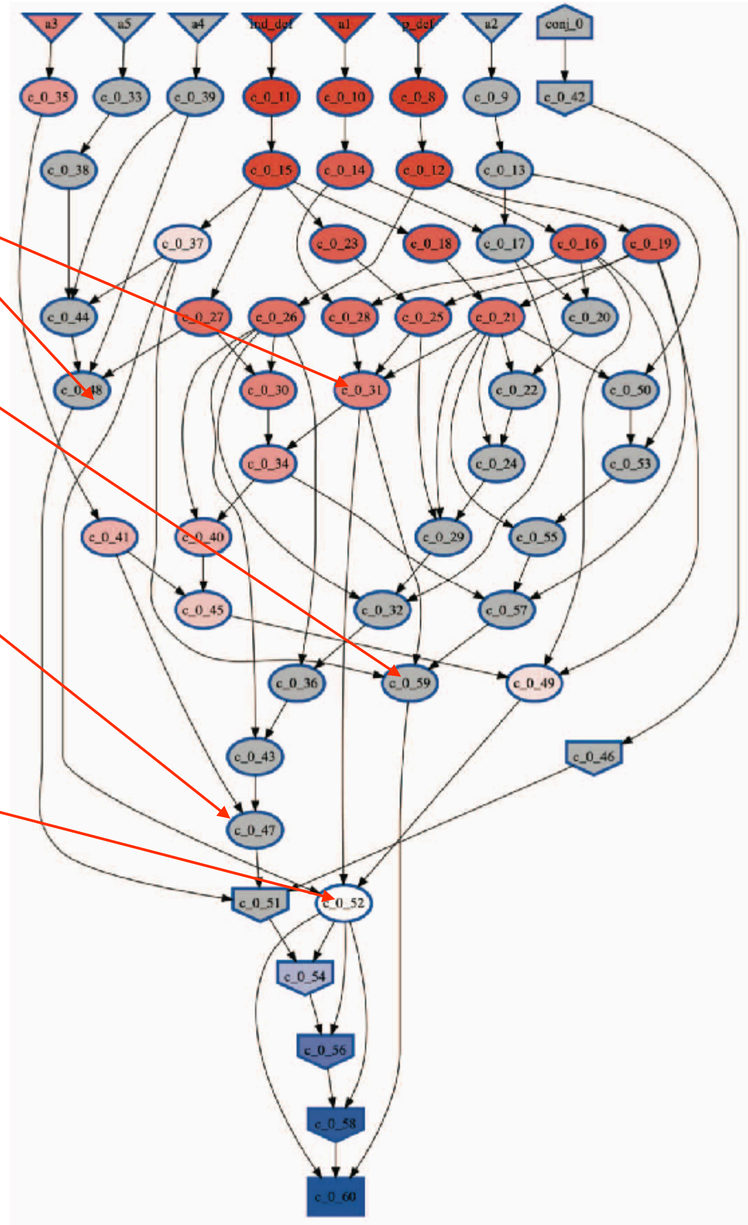
Informally, for all entities x and sets Y, if px, psx and Y are inductive sets (over e and s), then $f\ sx\ sssse$ is in Y.

Part III
From A1, A3, Def_*ind* and Def_*p*, it follows:

$$C_{52} : \forall x.\ ind\ px \rightarrow ind\ psx. \qquad \text{(L3)}$$

Informally, for any x: if px is an inductive set (over e and s), then so is psx.

IDV display of the resolution proof by E for BCP with *ind* and *p* (highlighted in red are the dependencies of clause *c*_0_52); the resolution proof by E is presented in full detail in Appendix C.



(E.g. clause C48 links to https://github.com/cbenzmueller/BoolosCuriousInference-ATP/blob/main/Boolos1.proof#L381-L384)

Part I
From A4, A5 and Def_*ind*, it follows that d is an inductive set:

$$C_{48} : ind\ d. \qquad \text{(L1a)}$$

From A1, Def_*ind* and Def_*p*, it follows that:

$$C_{31} : pxe \text{ for all } x. \qquad \text{(L1b)}$$

From A1, A2, Def_*ind* and Def_*p*, it follows that pe is an inductive set:

$$C_{59} : ind\ pe. \qquad \text{(L1c)}$$

Part II
From A1, A2, A3, Def_*ind* and Def_*p*, it follows:

$$C_{47} : \forall x.\forall Y.\ (ind\ px \wedge ind\ psx \wedge ind\ Y) \rightarrow Y\ fsxssssse. \qquad \text{(L2)}$$

Informally, for all entities x and sets Y, if px, psx and Y are inductive sets (over e and s), then $f\ sx\ sssse$ is in Y.

Part III
From A1, A3, Def_*ind* and Def_*p*, it follows:

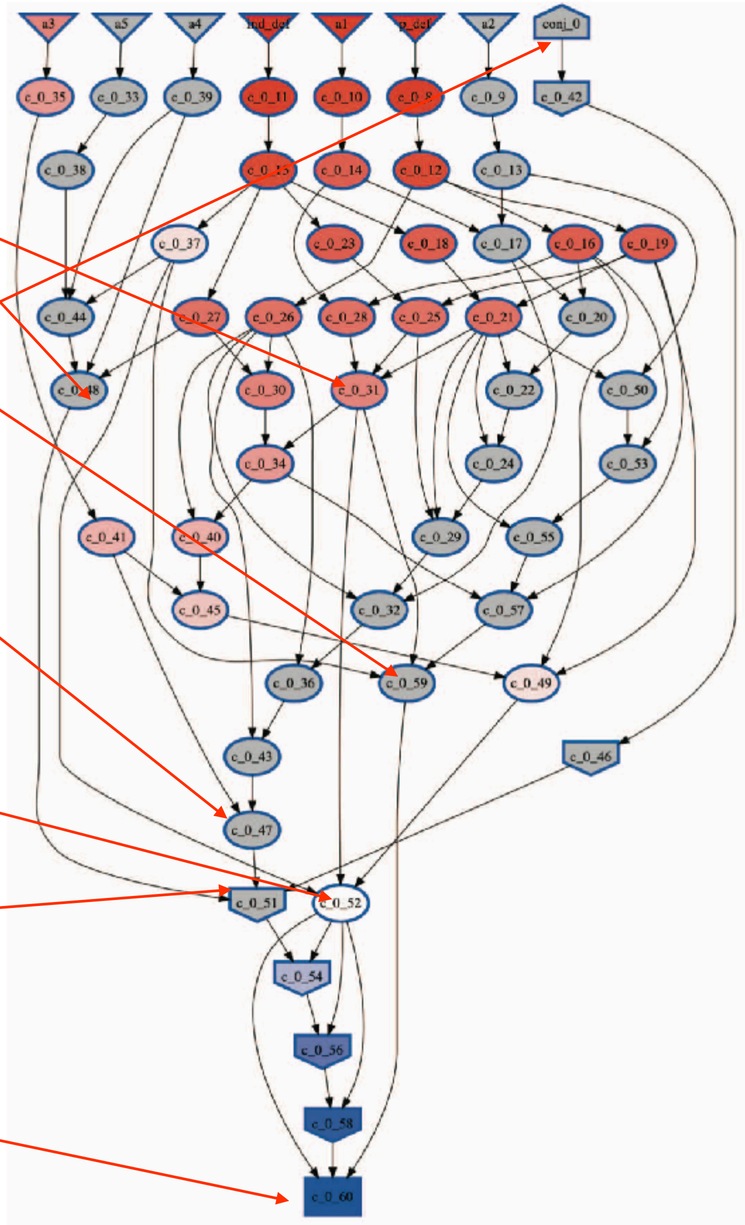
$$C_{52} : \forall x.\ ind\ px \rightarrow ind\ psx. \qquad \text{(L3)}$$

Informally, for any x: if px is an inductive set (over e and s), then so is psx.

Part IV: Proof of C by reductio ad absurdum
To prove C assume ¬C and derive a contradiction. Assume $\neg d\ fssssesssse$ (clauses C_{42} and C_{46}). From this and L1, obtain that $psssse$ is not an inductive set or that $pssse$ is not an inductive set, i.e. clause C_{51}. Use L3 to show that $pssse$, $psse$ and pse are not inductive sets (clauses C_{54}, C_{56} and C_{58}). From these, together with L1 and L3, derive a contradiction. By reductio ad absurdum C holds.

Part IV′: Constructive proof of C
The refutation argument IV can be converted into constructive argument IV′, avoiding reduction ad absurdum: from L1 and L3 pse is an inductive set. By further applications of L3, $psse$, $pssse$ and $psssse$ are inductive sets. Use L1a and L2 to conclude $d\ fssssesssse$.

IDV display of the resolution proof by E for BCP with *ind* and *p* (highlighted in red are the dependencies of clause *c*_0_52); the resolution proof by E is presented in full detail in Appendix C.



(E.g. clause C48 links to https://github.com/cbenzmueller/BoolosCuriousInference-ATP/blob/main/Boolos1.proof#L381-L384)

Part I
From A4, A5 and Def_*ind*, it follows that d is an inductive set:

$$C_{48} : ind\ d. \qquad \text{(L1a)}$$

From A1, Def_*ind* and Def_*p*, it follows that:

$$C_{31} : pxe \text{ for all } x. \qquad \text{(L1b)}$$

From A1, A2, Def_*ind* and Def_*p*, it follows that pe is an inductive set:

$$C_{59} : ind\ pe. \qquad \text{(L1c)}$$

Part II
From A1, A2, A3, Def_*ind* and Def_*p*, it follows:

$$C_{47} : \forall x.\forall Y.\ (ind\ px \wedge ind\ psx \wedge ind\ Y) \rightarrow Y\ fsxssssse. \qquad \text{(L2)}$$

Informally, for all entities x and sets Y, if px, psx and Y are inductive sets (over e and s), then $f\ sx\ sssse$ is in Y.

Part III
From A1, A3, Def_*ind* and Def_*p*, it follows:

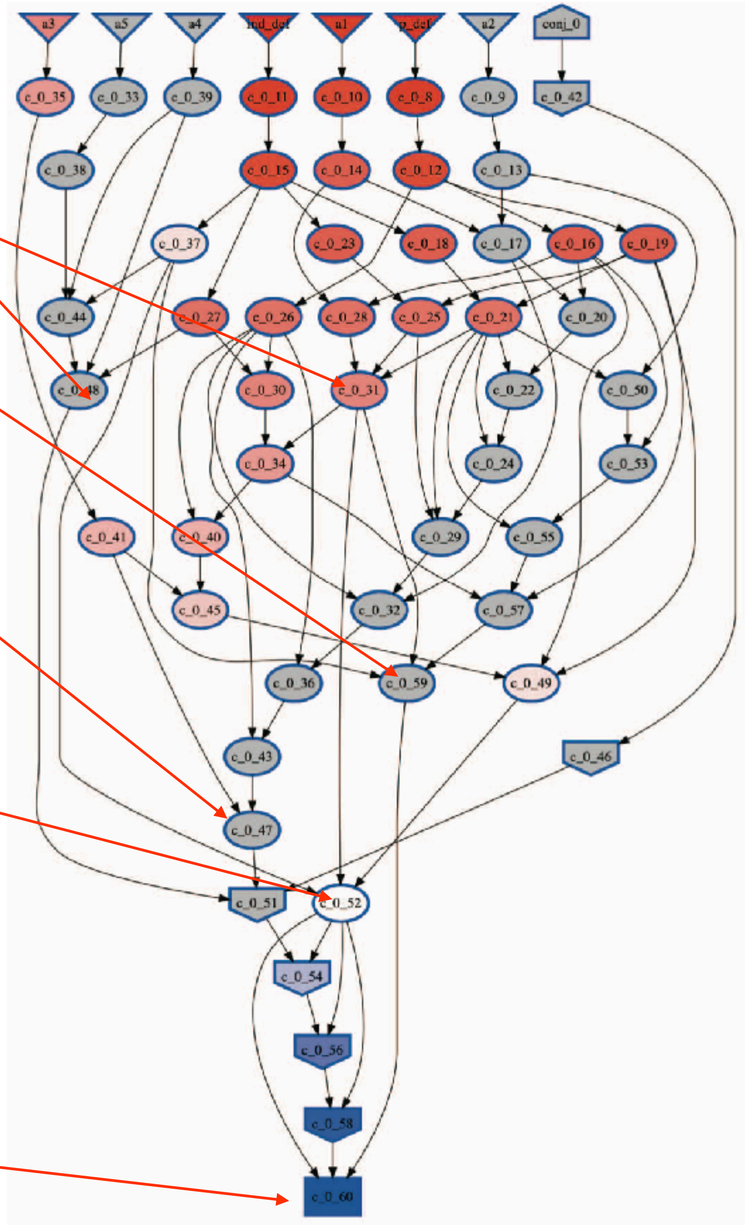
$$C_{52} : \forall x.\ ind\ px \rightarrow ind\ psx. \qquad \text{(L3)}$$

Informally, for any x: if px is an inductive set (over e and s), then so is psx.

Part IV: Proof of C by reductio ad absurdum
To prove C assume ¬C and derive a contradiction. Assume $\neg d\ fssssesssse$ (clauses C_{42} and C_{46}). From this and L1, obtain that $pssssse$ is not an inductive set or that $pssse$ is not an inductive set, i.e. clause C_{51}. Use L3 to show that $pssse$, $psse$ and pse are not inductive sets (clauses C_{54}, C_{56} and C_{58}). From these, together with L1 and L3, derive a contradiction. By reductio ad absurdum C holds.

Part IV′: Constructive proof of C
The refutation argument IV can be converted into constructive argument IV′, avoiding reduction ad absurdum: from L1 and L3 pse is an inductive set. By further applications of L3, $psse$, $pssse$ and $pssssse$ are inductive sets. Use L1a and L2 to conclude $d\ fssssesssse$.

IDV display of the resolution proof by E for BCP with *ind* and *p* (highlighted in red are the dependencies of clause *c*_0_52); the resolution proof by E is presented in full detail in Appendix C.



(E.g. clause C48 links to https://github.com/cbenzmueller/BoolosCuriousInference-ATP/blob/main/Boolos1.proof#L381-L384)

Challenge

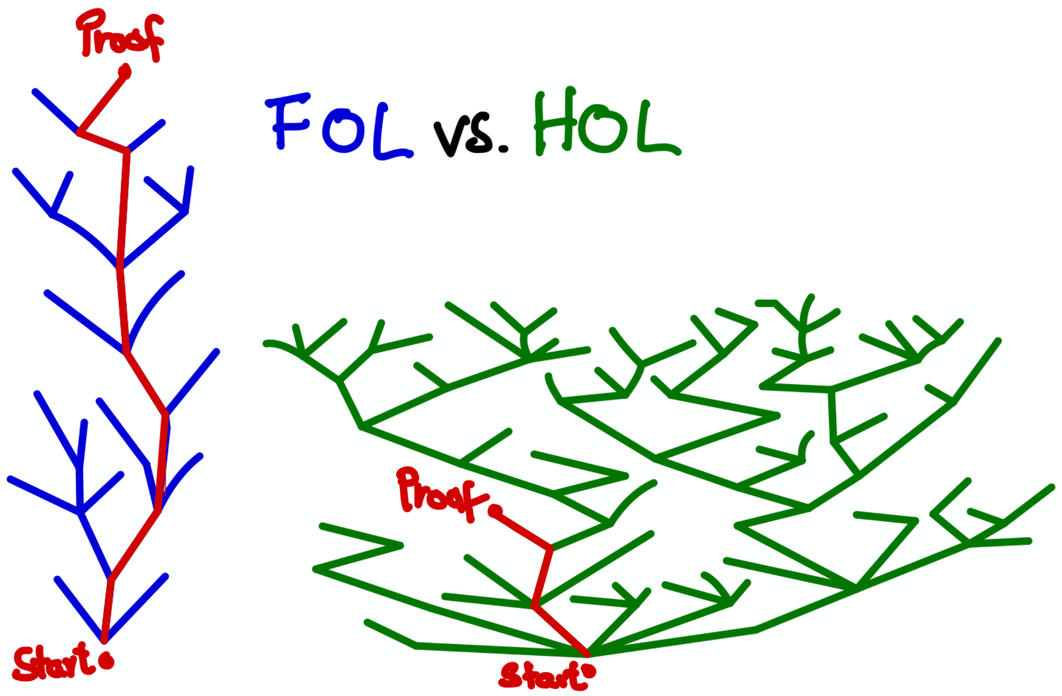
Automatically find the shorthand notations

https://github.com/cbenzmueller/BoolosCuriousInference-ATP/blob/main/Boolos1.p

```
% Declarations
thf(e_decl,type, e: $i ).
thf(s_decl,type, s: $i > $i ).
thf(d_decl,type, d: $i > $o ).
thf(f_decl,type, f: $i > $i > $i ).

% Auxiliary declarations
thf(ind_decl,type, ind: ( $i > $o ) > $o ).
thf(p_decl,type, p: $i > $i > $o ).

% Axioms
thf(a1,axiom,
    ! [N: $i] :
      ( ( f @ N @ e )
      = ( s @ e ) ) ).

thf(a2,axiom,
    ! [Y: $i] :
      ( ( f @ e @ ( s @ Y ) )
      = ( s @ ( s @ ( f @ e @ Y ) ) ) ) ).

thf(a3,axiom,
    ! [X: $i,Y: $i] :
      ( ( f @ ( s @ X ) @ ( s @ Y ) )
      = ( f @ X @ ( f @ ( s @ X ) @ Y ) ) ) ).

thf(a4,axiom,
    d @ e ).

thf(a5,axiom,
    ! [X: $i] :
      ( ( d @ X )
     => ( d @ ( s @ X ) ) ) ).

% Shorthand notations
thf(ind_def,axiom,
    ( ind
    = ( ^ [Q: $i > $o] :
          ( ( Q @ e )
          & ! [X: $i] :
              ( ( Q @ X )
             => ( Q @ ( s @ X ) ) ) ) ) ) ).

thf(p_def,axiom,
    ( p
    = ( ^ [X: $i,Y: $i] :
          ( ^ [X2: $i] :
              ! [X3: $i > $o] :
                ( ( ind @ X3 )
               => ( X3 @ X2 ) )
          @ ( f @ X @ Y ) ) ) ) ).

% Conjecture
thf(conj_0,conjecture,
    d @ ( f @ ( s @ ( s @ ( s @ ( s @ ( s @ e ) ) ) ) ) @ ( s @ ( s @ ( s @ ( s @ ( s @ e ) ) ) ) ) ) ).
```

Observations & Questions

HOL: Expressivity Matters — Boolos' Curius Inference —



Proofs in HOL can be short, elegant, intuitive

We need both

- Robust HOL ATP (cut-elimination)
- Clever HOL ATP (cut-introduction)

How to achieve the latter

- Machine Learning & AI?
- Proof Planning?
- Combinations of techniques?

Challenge: controlled reasoning with cut-strong principles