

Arithmetical theories and the automation of induction

Stefan Hetzl Jannik Vierling

Institute of Discrete Mathematics and Geometry
TU Wien, Austria

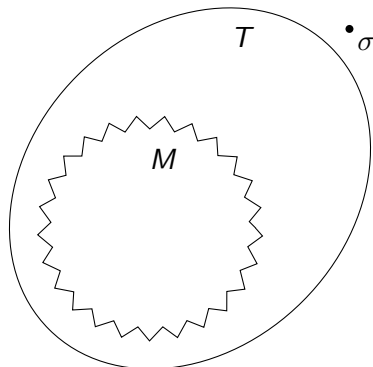
42ème Journées sur les Arithmétiques Faibles
Karlovasi, Greece
September 25, 2023

Motivation: two areas

- Mathematical logic: theories of arithmetic
- Computer science: automated inductive theorem proving
 - Find proof by induction automatically
History in computer science dating back to the 1970ies
 - Difference to first-order theorem proving (validity):
No cut-elimination theorem
 - Methods: recursion analysis, term rewriting, rippling, extensions of saturation-based provers, cyclic proofs, theory exploration, ...
 - Typical problems:
 $\forall x \forall y \ x + y = y + x$
 $\forall l \ \text{len}(\text{rev}(l)) = \text{len}(l)$
 - Empirical evaluation of implementations

Motivation: building a connection

- Use mathematical logic in order to:
 1. classify strength of methods
 2. obtain independence results
- Straightforward results, e.g., $T = \text{PA}$, $\sigma = \text{Con}_{\text{PA}}$
- **Practically meaningful** independence results
- ▶ Weak arithmetical theories



Specifics of this setting

- Often: M and T parametric in language / basic axioms
- Coding does not play a role
- Not only numbers, also: lists, trees, etc.
- Bounded quantifiers are not distinguished
- Sometimes: idiosyncracies of method M reflected in T

Outline

- 1 Introduction
- 2 Induction on literals**
- 3 Open Induction
- 4 Existential Induction
- 5 Conclusion

- **Definition.** A *literal* is an atom or a negated atom.
- **Definition.** The induction axiom $I_x\varphi(x, \bar{z})$ is

$$\varphi(0, \bar{z}) \wedge \forall x (\varphi(x, \bar{z}) \rightarrow \varphi(s(x), \bar{z})) \rightarrow \forall x \varphi(x, \bar{z}).$$

- **Definition.** For a set of formulas Γ define

$$\Gamma\text{-IND} = \{I_x\varphi(x, \bar{z}) \mid \varphi(x, \bar{z}) \in \Gamma\}$$

- In particular: Literal-IND
- **Observation.** For natural axioms A in language $L = \{0, s, p, +\}$:

$$A + \text{Literal-IND} \equiv A + \text{Open-IND}.$$

Saturation theorem proving

- Standard technique for automated theorem proving in FOL
- **Definition.** *Clause* is a formula $\bigvee_{i=1}^n L_i$ where L_i literal.
- **Definition.** *Saturation system* \mathcal{S} is a set of rules.
- **Example.** The *resolution rule* is

$$\frac{C \vee L \quad L' \vee D}{(C \vee D)\sigma}$$

where σ is most general unifier of L and $\overline{L'}$.

- **Definition.** Clause set \mathcal{C} *closed* under \mathcal{S} if for all n -ary rules $\rho \in \mathcal{S}$:
 $C_1, \dots, C_n \in \mathcal{C}$ implies $\rho(C_1, \dots, C_n) \in \mathcal{C}$
Given \mathcal{C} , compute closure by $\mathcal{C}^0 = \mathcal{C}, \mathcal{C}^1, \mathcal{C}^2, \dots \longrightarrow \mathcal{C}^\omega$.
- **Definition.** \mathcal{S} *sound* if $C \in \mathcal{C}^\omega$ implies $C \models \mathcal{C}$
- **Definition.** \mathcal{S} *refutationally complete* if $\mathcal{C} \models \perp$ implies $\perp \in \mathcal{C}^\omega$

Adding explicit induction axioms

- Adding induction:

$$\overline{\text{CNF}(\text{sk}^{\exists}(I_x\varphi(x)))}$$

- Vampire prover [Voronkov et al. '20]: single clause induction

$$\frac{\overline{L(a)} \vee C}{\text{CNF}(\text{sk}^{\exists}(I_xL(x)))} \text{ SCIND}$$

a constant symbol, $L(x)$ literal, $L(a)$ variable-free

- **Example.** $\mathcal{S} + \text{SCIND}$ refutes

$$\{x + 0 = 0, x + s(y) = s(x + y), c + (c + c) \neq (c + c) + c\}$$

Characterisation and independence result

- **Theorem.** \mathcal{S} sound saturation system, $T \exists_2$ theory. If $\mathcal{S} + \text{SCIND}$ refutes $\text{CNF}(\text{sk}^\exists(T))$ then $T + \text{Literal-IND}$ is inconsistent.

- **Definition.**

Let $T = \{ 0 \neq s(x), s(x) = s(y) \rightarrow x = y, \\ E(0), E(x) \rightarrow O(s(x)), O(x) \rightarrow E(s(x)), \neg E(c) \wedge \neg O(c) \}.$

- **Theorem.** $T + \text{Literal-IND}$ is consistent.
- **Corollary.** If \mathcal{S} sound saturation system, then $\mathcal{S} + \text{SCIND}$ does not refute T .

Outline

- 1 Introduction
- 2 Induction on literals
- 3 Open Induction**
- 4 Existential Induction
- 5 Conclusion

- **Theorem.** [Shoenfield '58] Over

$$s(x) \neq 0$$

$$p(0) = 0$$

$$p(s(x)) = x$$

$$x + 0 = x$$

$$x + s(y) = s(x + y)$$

open induction is equivalent to

$$x + y = y + x$$

$$(x + y) + z = x + (y + z)$$

$$x = 0 \vee x = s(p(x))$$

$$x + y = x + z \rightarrow y = z$$

- Shepherdson '60ies: systematic study

- **Definition.** Let $L = \{\text{nil}, \text{cons}, \frown\}$, let $T =$

$$\begin{aligned} & \text{nil} \neq \text{cons}(x, X) \\ \text{cons}(x, X) = \text{cons}(y, Y) & \rightarrow x = y \wedge X = Y \\ \text{nil} \frown Y & = Y \\ \text{cons}(x, X) \frown Y & = \text{cons}(x, X \frown Y) \end{aligned}$$

- **Theorem.** [H, Vierling]

$$T + \text{Open-IND} \not\vdash Y \frown X = Z \frown X \rightarrow Y = Z.$$

- **Project.** Systematic picture of subsystems of open induction
 - atomic, literal, clause, dual clause, open induction
 - for numbers (in various signatures)
 - for lists, trees, ...

Outline

- 1 Introduction
- 2 Induction on literals
- 3 Open Induction
- 4 Existential Induction**
- 5 Conclusion

- Abstraction of n -clause calculus [Kersani, Peltier '13; Kersani '14]
- **Definition.** An $L \cup \{\eta\}$ clause set \mathcal{C} is a *clause set cycle (CSC)* if $\mathcal{C}(s(\eta)) \models \mathcal{C}(\eta)$ and $\mathcal{C}(0) \models \perp$. An $L \cup \{\eta\}$ clause set $\mathcal{D}(\eta)$ is refuted by a CSC $\mathcal{C}(\eta)$ if $\mathcal{D}(\eta) \models \mathcal{C}(\eta)$.
- Variants
- **Example.** CSC refutes Even/Odd example

- **Definition.** Γ set of formulas, define

$$\frac{\varphi(0) \quad \varphi(x) \rightarrow \varphi(s(x))}{\varphi(\eta)} \Gamma\text{-IND}_{\eta}^{\text{R-}}$$

where $\varphi(x) \in \Gamma$.

- **Definition.** T theory, R inference rule, define

$$[T, R] = T + \{\varphi \mid T \vdash \Gamma, \Gamma/\varphi \in R\}.$$

- **Theorem.** \mathcal{D} is refuted by a CSC iff $\mathcal{D} + [\emptyset, \exists_1\text{-IND}_{\eta}^{\text{R-}}] \vdash \perp$.

- **Definition.** $L_{LA} = \{0, s, p, +\}$

$$T = \{ s(0) \neq 0, p(0) = 0, p(s(x)) = x, \\ x + 0 = x, x + s(y) = s(x + y), \\ x + y = y + x, x + (y + z) = (x + y) + z \}$$

- **Definition.** Let $k, n, m \in \mathbb{N}$ with $0 < n < m$, define $E_{k,n,m}$ as:

$$n \cdot x + \overline{(m-n)k} = m \cdot x \rightarrow x = \bar{k}$$

- **Example.** $x + 0 = x + x \rightarrow x = 0$ ($E_{0,1,2}$)

- **Theorem.** $T + \exists_1\text{-IND}^- \not\vdash E_{k,n,m}$

- **Corollary.** $\mathcal{E}_{k,n,m}(\eta)$ is not refuted by an L_{LA} clause set cycle.

Independence result: proof

Theorem

$$T + \exists_1\text{-IND}^- \not\vdash E_{k,n,m}$$

Proof.

Countermodel \mathcal{M} , domain $\{(i, n) \in \mathbb{N} \times \mathbb{Z} \mid i = 0 \text{ implies } n \in \mathbb{N}\}$

$$0^{\mathcal{M}} = (0, 0)$$

$$p^{\mathcal{M}}((0, n)) = (0, n \dot{-} 1)$$

$$s^{\mathcal{M}}(i, n) = (i, n + 1)$$

$$p^{\mathcal{M}}((i, n)) = (i, n - 1) \text{ if } i > 0$$

$$(i, n) +^{\mathcal{M}} (j, m) = (\max(i, j), n + m)$$



- Iterations of CSCs ... nested applications of $\exists_1\text{-IND}_\eta^{\text{R-}}$
- **Theorem.** For every $k \in \mathbb{N}$ there is a clause set \mathcal{C}_k which is refuted by $[\emptyset, \exists_1\text{-IND}^{\text{R-}}]_{k+1}$ but not by $[\emptyset, \exists_1\text{-IND}^{\text{R-}}]_k$
Proof Sketch. Totality of a suitable function f_k .
- **Question.** Is $T + \exists_1\text{-IND}^- \forall_2\text{-conservative}$ over $T + \exists_1\text{-IND}^{\text{R-}}$?
- **Definition.** Let $T_0 = \{ s(0) \neq 0, p(0) = 0, p(s(x)) = x, x + 0 = x, x + s(y) = s(x + y) \}$.
- **Conjecture.** $T_0 + \exists_1\text{-IND}^- \not\vdash x + (x + x) = (x + x) + x$.
(Note that $T_0 + \text{Literal-IND} \vdash x + (x + x) = (x + x) + x$.)
(Would yield corollary on CSCs)

Conclusion and future work

- Strategy for analysing method M :
 - Find upper bound T for strength of M
 - Independence result for T
- ▶ Logical foundations of automated inductive theorem proving
 - ▶ Clarify relationship between methods, challenge problems
 - ▶ New questions / problems about (weak) arithmetical theories

Future Work:

- Analyse further methods: term rewriting, theory exploration, ...
- Towards a systematic picture: which method solves which problem?
- (Weak) theories of inductive datatypes: lists, trees, etc.
- Analyticity

Thank you!



Stefan Hetzl and Jannik Vierling.

Unprovability results for clause set cycles.

Theoretical Computer Science, 935, 21–46, 2022.



Stefan Hetzl and Jannik Vierling.

Induction and Skolemization in saturation theorem proving.

Annals of Pure and Applied Logic, 174(1):103167, 2023.



Jannik Vierling.

The limits of automated inductive theorem provers.

Ph.D. thesis, TU Wien, Austria.