Computational complexity of grammars for proofs with induction

Gabriel Ebner

First International Workshop on
Proof Theory for Automated Deduction,
Automated Deduction for Proof Theory

2019-10-24

Technische Universität Wien / Vrije Universiteit Amsterdam

Introduction

Grammars

Complexity of decision problems

More tractable subclasses

Back to proofs

Conclusion

Herbrand's theorem

Theorem (special case of Herbrand 1930)

Let $\varphi(x)$ be a quantifier-free first-order formula.

Then $\exists x\, \varphi(x)$ is valid iff there exist terms $t_1, \ldots, t_n$ such that $\varphi(t_1) \vee \cdots \vee \varphi(t_n)$ is a tautology.

Grammar-based proof analysis (Hetzl 2012)

$$\begin{tikzcd}
 & \text{cut-free proof} \arrow[d, dotted, no head] \\
\pi \arrow[r, mapsto] & \pi^* \arrow[d, mapsto] \\
 & H(\pi^*) \arrow[d, dotted, no head] \\
 & \text{set of terms} \equiv \text{Herbrand sequent}
\end{tikzcd}$$

Grammar-based proof analysis (Hetzl 2012)

$$
\begin{tikzcd}
 & \text{cut-free proof} \arrow[d, dash, dotted] \\
\pi \arrow[r, mapsto] \arrow[d, mapsto] & \pi^* \arrow[d, mapsto] \\
G(\pi) \arrow[r, mapsto] \arrow[d, dash, dotted] & H(\pi^*) \arrow[d, dash, dotted] \\
\text{grammar} & \text{set of terms} \equiv \text{Herbrand sequent}
\end{tikzcd}
$$

- Assign (tree) grammar $G(\pi)$ to proof π
  such that the language $L(G(\pi))$ is a Herbrand sequent

Grammar-based proof analysis (Hetzl 2012)

$$
\begin{tikzcd}
 & & \text{cut-free proof} \arrow[d, dotted, no head] \\
\pi \arrow[rr, mapsto] \arrow[d, mapsto] & & \pi^* \arrow[d, mapsto] \\
G(\pi) \arrow[r, mapsto] \arrow[d, dotted, no head] & L(G(\pi)) \supseteq & H(\pi^*) \arrow[d, dotted, no head] \\
\text{grammar} & & \text{set of terms} \equiv \text{Herbrand sequent}
\end{tikzcd}
$$

- Assign (tree) grammar $G(\pi)$ to proof π
  such that the language $L(G(\pi))$ is a Herbrand sequent

Applications

- Invert cut-elimination: given a cut-free proof π, find a proof π' with cuts.
    1. Find grammar G such that $L(G) \supseteq H(\pi)$
    2. Compute cut-formulas

  $\rightarrow$ Lemma generation
  $\rightarrow$ Invariants for inductive proofs

- Working with grammars reduces bureaucracy:
    - e.g. prove uncompressibility result for grammars, then lift to proofs (Eberhard, Hetzl 2018)

VTRATGs

Vectorial totally rigid acyclic tree grammars.

Grammars for proofs with purely universally quantified cuts.

- Start symbol: A
- Nonterminal vectors: $A, \overline{B}, \overline{C}, \overline{D}, \ldots$
  where $\overline{B} = (B_1, \ldots, B_n)$, etc.
- (Acyclic) productions: $\overline{B} \to \overline{t}[\overline{C}, \overline{D}, \ldots]$

Rigid derivations: $A[A \backslash t_1][\overline{B} \backslash \overline{t_2}][\overline{C} \backslash \overline{t_3}] \cdots$

(Finite!) language $L(G)$ consists of all derivable terms

VTRATG example

$$A \rightarrow f(B_1, B_1, B_2) \mid g(B_1, B_1, B_2)$$
$$\overline{B} \rightarrow (c, e) \mid (d, f)$$

VTRATG example

$$
\begin{aligned}
A &\rightarrow f(B_1, B_1, B_2) \mid g(B_1, B_1, B_2) \\
\overline{B} &\rightarrow (c, e) \mid (d, f)
\end{aligned}
$$

$$L(G) = \{f(c, c, e), f(d, d, f), g(c, c, e), g(d, d, f)\}$$

Cut-reduction to language generation

$L(G(\pi))$ contains the formulas in a Herbrand sequent of π

$G(\pi)$ consists of:

- Nonterminals: eigenvariables from cuts + start symbol A
- Productions $\alpha \to t$ for weak quantifier inferences on cut formulas:

$$\dfrac{\dfrac{\vdash \varphi(\alpha)}{\vdash \forall x\, \varphi(x)}\,\forall_r \qquad \dfrac{\dfrac{\varphi(t) \vdash}{}\,\forall_l \quad \vdots}{\forall x\, \varphi(x) \vdash}}{}\,\mathrm{cut}$$

- Productions $\mathrm{A} \to \varphi(t)$ for instances of formulas end-sequent.

Simple induction proofs (Eberhard, Hetzl 2015)

$$\dfrac{\dfrac{\overset{(\pi_1)}{\Gamma \vdash \forall x\, \psi(0, \bar{x})} \qquad \overset{(\pi_2)}{\Gamma, \forall x\, \psi(s(\nu), \bar{x}) \vdash \forall \bar{x}\, \psi(\nu, \bar{x})}}{\Gamma \vdash \forall \bar{x}\, \psi(\alpha, \bar{x})}\,\text{ind} \qquad \overset{(\pi_3)}{\Gamma, \forall \bar{x}\, \psi(\alpha, \bar{x}) \vdash \varphi(\alpha)}}{\Gamma \vdash \varphi(\alpha)}$$

where π_1, π_2, π_3 are cut-free
and ψ, φ are quantifier-free

$\rightarrow$ study induced Herbrand sequents of $\Gamma \vdash \varphi(n)$ for numerals n.

Grammar assignment to simple induction proofs (Eberhard, Hetzl 2015)

for each numeral n

$$
\begin{tikzcd}
\pi \arrow[r, mapsto, "\text{induction-elim.}"] \arrow[d, mapsto] & \pi_n \arrow[rr, mapsto, "\text{cut-elim.}"] & & \pi_n^* \arrow[d, mapsto] \\
G(\pi) \arrow[r, mapsto, "\text{instantiates}"] \arrow[d, dotted, no head] & I(G(\pi), n) \arrow[r, mapsto, "\text{gen.}"] \arrow[d, dotted, no head] & L(G(\pi), n) \arrow[r, phantom, "\supseteq"] & H(\pi_n^*) \arrow[d, dotted, no head] \\
\text{ind. grammar} & \text{VTRATG} & & \text{H-sequent}
\end{tikzcd}
$$

Induction grammars

- Two kinds of (cyclic!) productions:
    - $\tau \to t[\overline{\gamma}, \alpha, \nu]$
    - $\overline{\gamma} \to t[\overline{\gamma}, \alpha, \nu]$
- Instantiation: for each numeral n, set $L(G, n) = L(I(G, n))$ for VTRATG $I(G, n)$:
    - Nonterminals $\tau, \overline{\gamma_0}, \ldots, \overline{\gamma_n}$.
    - $\tau \to t[\alpha, \nu, \overline{\gamma}] \quad \rightsquigarrow \quad \tau \to t[n, k, \overline{\gamma_{s(k)}}] \quad$ for $s(k) < n$
    - $\overline{\gamma} \to \bar{t}[\alpha] \quad \rightsquigarrow \quad \overline{\gamma_k} \to \bar{t}[n] \quad$ for $k < n$
    - $\overline{\gamma} \to \bar{t}[\alpha, \nu, \overline{\gamma}] \quad \rightsquigarrow \quad \overline{\gamma_k} \to \bar{t}[n, k, \overline{\gamma_{s(k)}}] \quad$ for $s(k) < n$
    - (corresponds to unrolling of induction inference)

Induction grammar example

$$\tau \to r(\gamma_1, \gamma_1) \qquad\qquad \overline{\gamma} \to (f(\gamma_2), g(\gamma_1)) \mid (c, d)$$

Instantiates to $I(G, 2)$:

$$\begin{aligned} \tau &\to r(\gamma_{0,1}, \gamma_{0,1}) \mid r(\gamma_{1,1}, \gamma_{1,1}) \mid r(\gamma_{2,1}, \gamma_{2,1}) \\ (\gamma_{0,1}, \gamma_{0,2}) &\to (f(\gamma_{1,2}), g(\gamma_{1,1})) \mid (c, d) \\ (\gamma_{1,1}, \gamma_{1,2}) &\to (f(\gamma_{2,2}), g(\gamma_{2,1})) \mid (c, d) \\ (\gamma_{2,1}, \gamma_{2,2}) &\to (c, d) \end{aligned}$$

And $L(G, 2) = \{r(f(g(c)), f(g(c))), r(f(d), f(d)), r(c, c)\}$

Minimal cover

Problem (TRATG-COVER)

Input: set of terms T and a number k.

Output: is there a TRATG G with at most k productions such that $L(G) \supseteq T$?

Surprisingly hard. We only know that it is in NP.

Decision problems on VTRATGs (Eberhard, E, Hetzl 2018)

VTRATG-Membership:	$t \in L(G)$	NP-complete
VTRATG-Emptiness:	$L(G) = \emptyset$	coNP-complete
VTRATG-Containment:	$L(G_1) \subseteq L(G_2)$	Π_2^P-complete
VTRATG-Disjointness:	$L(G_1) \cap L(G_2) = \emptyset$	coNP-complete
VTRATG-Equivalence:	$L(G_1) = L(G_2)$	Π_2^P-complete

Decision problems on induction grammars

Ind-Membership:	$t \in L(G, n)$	NP-complete
Ind-Emptiness:	$\forall n \; L(G, n) = \emptyset$	**PSPACE-complete**
Ind-Containment:	$\forall n \; L(G_1, n) \subseteq L(G_2, n)$	**undecidable**
Ind-Disjointness:	$\forall n \; L(G_1, n) \cap L(G_2, n) = \emptyset$	**undecidable**
Ind-Equivalence:	$\forall n \; L(G_1, n) = L(G_2, n)$	**undecidable**

Post Correspondence Problem

Problem (PCP)

Input: two finite lists of words $w_1, \dots, w_n$ and $v_1, \dots, v_n$

Output: is there a sequence of indices $i_1, \dots, i_k$ with $k > 0$ such that $w_{i_1} \dots w_{i_k} = v_{i_1} \dots v_{i_k}$?

Undecidable (Post 1946).

Disjointness

Theorem
Ind-Disjointness is undecidable.

Proof.
Reduce PCP to Ind-Disjointness.

Construct two induction grammars Image_P and $\mathrm{Equal}_{\Sigma,l}$:

- Image_P generates all pairs $(w_{i_1} \cdots w_{i_l}, v_{i_1} \cdots v_{i_l})$
- $\mathrm{Equal}_{\Sigma,l}$ generates all pairs (w, w)

A word $a_1 a_2 \ldots a_n$ is encoded as a unary term $a_1(a_2(\ldots a_n(\epsilon)))$.

Then the PCP instance has *no* solution iff:

$$\forall i \quad L(\mathrm{Image}_P, i) \cap L(\mathrm{Equal}_{\Sigma,l}, i) = \emptyset \qquad \square$$

Disjointness of induction grammars

Let $w = a_1 a_2 \dots a_k$, then $w \cdot \gamma = a_1(a_2(\dots a_k(\gamma)))$.

Definition

The induction grammar $\mathrm{Equal}_{\Sigma,l}$ has the following productions:

$$\begin{aligned} \tau &\to r(\gamma, \gamma) \\ \gamma &\to w \cdot \gamma \mid w \quad \text{where } |w| \leq l \end{aligned}$$

Lemma

$L(\mathrm{Equal}_{\Sigma,l}, k) = \{r(w, w) \mid w \in \Sigma^*, |w| \leq l(k+1)\}$

Disjointness of induction grammars

Definition

The induction grammar Image_P has the following productions:

$$\begin{aligned} \tau &\to r(\gamma_1, \gamma_2) \\ (\gamma_1, \gamma_2) &\to (w_1 \cdot \gamma_1, v_1 \cdot \gamma_2) \mid \cdots \mid (w_n \cdot \gamma_1, v_n \cdot \gamma_2) \\ (\gamma_1, \gamma_2) &\to (w_1, v_1) \mid \cdots \mid (w_n, v_n) \end{aligned}$$

Lemma

$L(\mathrm{Image}_P, k) = \{r(w_{i_1} \cdots w_{i_l}, v_{i_1} \cdots v_{i_l}) \mid 1 \leq l \leq k+1\}$

Containment of induction grammars

Theorem
IND-CONTAINMENT is undecidable.

Proof.
Similar to IND-DISJOINTNESS.

Construct two induction grammars Image_P and $\mathrm{Diff}_{\Sigma,l}$:

- Image_P as before
- $\mathrm{Diff}_{\Sigma,l}$ generates all pairs of different words

Then the PCP instance has *no* solution iff:

$$\forall i \quad L(\mathrm{Image}_P, i) \subseteq L(\mathrm{Diff}_{\Sigma,l}, i) \qquad \square$$

Containment of induction grammars

Definition

The induction grammar $\mathrm{Diff}_{\Sigma,l}$ has the following productions:

$$\tau \to r(\gamma_1, \gamma_2)$$

$$\overline{\gamma} \to (t \cdot \gamma_1, u \cdot \gamma_2, v \cdot \gamma_3, w \cdot \gamma_4) \quad \text{where } |t| = |u| \leq l \wedge \max(|v|, |w|) \leq l$$

$$\overline{\gamma} \to (t \cdot \gamma_3, u \cdot \gamma_4, v \cdot \gamma_3, w \cdot \gamma_4) \quad \text{where } |t| = |u| \leq l \wedge \max(|v|, |w|) \leq l \wedge t \neq u$$

$$\overline{\gamma} \to (t, u, v, w) \quad \text{where } \max(|t|, |u|, |v|, |w|) \leq l \wedge t \neq u$$

where $t, u, v, w \in \Sigma^*$ and $\overline{\gamma} = (\gamma_1, \gamma_2, \gamma_3, \gamma_4)$

Lemma

$$L(\mathrm{Diff}_{\Sigma,l}, k) = \{r(v, w) \mid v \neq w \in \Sigma^* \wedge \max(|v|, |w|) \leq l(k+1)\}$$

Dependency graphs of VTRATGs

Hardness results on VTRATGs require complicated grammars.
Typical grammars (such as $I(G, n)$) are much simpler.

Dependency graphs of VTRATGs

Hardness results on VTRATGs require complicated grammars.
Typical grammars (such as $I(G, n)$) are much simpler.

This VTRATG G is almost "linear":
$A \rightarrow f(B) \quad B \rightarrow f(C) \quad C \rightarrow g(D) \quad D \rightarrow c$

Dependency graphs of VTRATGs

Hardness results on VTRATGs require complicated grammars.
Typical grammars (such as $I(G, n)$) are much simpler.

This VTRATG G is almost "linear":
$A \to f(B) \quad B \to f(C) \quad C \to g(D) \quad D \to c$

Assign dependency graph $D(G)$:
$A - B - C - D$

Dependency graphs of VTRATGs

Hardness results on VTRATGs require complicated grammars.
Typical grammars (such as $I(G, n)$) are much simpler.

This VTRATG G is almost "linear":
$A \to f(B) \quad B \to f(C) \quad C \to g(D) \quad D \to c$

Assign dependency graph $D(G)$:
$A - B - C - D$

The treewidth $\mathrm{tw}(D(G))$ measures how close it is to a tree:

- Let G be a connected graph with at least two vertices, then $\mathrm{tw}(G) = 1$ iff G is a tree

Dependency graphs of VTRATGs

Hardness results on VTRATGs require complicated grammars.
Typical grammars (such as $I(G, n)$) are much simpler.

This VTRATG G is almost “linear”:
$A \to f(B) \quad B \to f(C) \quad C \to g(D) \quad D \to c$

Assign dependency graph $D(G)$:
$A - B - C - D$

The treewidth $\mathrm{tw}(D(G))$ measures how close it is to a tree:

- Let G be a connected graph with at least two vertices, then $\mathrm{tw}(G) = 1$ iff G is a tree

We have $\mathrm{tw}(D(I(G, n))) \leq 2|\overline{\gamma}|$!

Treewidth-bounded dependency graphs

(tw $\leq k$)-Membership:	$t \in L(G)$	P
(tw $\leq k$)-Emptiness:	$L(G) = \emptyset$	P
(tw $\leq k$)-Containment:	$L(G_1) \subseteq L(G_2)$	coNP-complete
(tw $\leq k$)-Disjointness:	$L(G_1) \cap L(G_2) = \emptyset$	coNP-complete
(tw $\leq k$)-Equivalence:	$L(G_1) = L(G_2)$	coNP-complete

These complexity results apply to $I(G, n)$ since we have $\mathrm{tw}(D(I(G, n))) \leq 2|\overline{\gamma}|$.

Induction grammars with bounded $|\overline{\gamma}|$

$(\lvert\overline{\gamma}\rvert \leq k)$-Ind-Membership:	$t \in L(G)$	P
$(\lvert\overline{\gamma}\rvert \leq k)$-Ind-Emptiness:	$\forall n\, L(G, n) = \emptyset$	P
$(\lvert\overline{\gamma}\rvert \leq k)$-Ind-Containment:	$\forall n\, L(G_1, n) \subseteq L(G_2, n)$	undec.
$(\lvert\overline{\gamma}\rvert \leq k)$-Ind-Disjointness:	$\forall n\, L(G_1, n) \cap L(G_2, n) = \emptyset$	undec.
$(\lvert\overline{\gamma}\rvert \leq k)$-Ind-Equivalence:	$\forall n\, L(G_1, n) = L(G_2, n)$	undec.

Introduction

Grammars

Complexity of decision problems

More tractable subclasses

Back to proofs

Conclusion

Problems on proofs

- Complexity results also transfer to proofs.
- Want to find simple induction proofs $\pi(G)$ such that e.g.:

$$L(G(\pi(G_1)), n) \subseteq L(G(\pi(G_2)), n) \leftrightarrow H(\pi(G_1)^*_n) \subseteq H(\pi(G_2)^*_n)$$

- Main technical challenge: weakening inferences.
  - in general $L(G(\pi), n) \supset H(\pi^*_n)$!

Non-erasing cut-reduction

Theorem (Hetzl, Straßburger 2012)

- *For every Gentzen cut-reduction sequence $\pi \leadsto \pi'$, we have $L(G(\pi)) \supseteq L(G(\pi'))$.*
- *If we did not perform grade reduction on weakenings, then $L(G(\pi)) = L(G(\pi'))$.*

Let $\overset{ne}{\leadsto}$ be the *non-erasing* Gentzen cut-reduction relation, i.e. where we do not reduce weakenings.

We can still define $H(\cdot)$ on $\overset{ne}{\leadsto}$-NFs.

Decision problems on proofs

Problem (SIP-CONTAINMENT).

Input: simple induction proofs π, π'.

Let $\pi_n^*, \pi_n'^*$ be $\overset{ne}{\rightsquigarrow}$-NFs such that $\pi_n \overset{ne}{\rightsquigarrow} \pi_n^*$ and $\pi_n' \overset{ne}{\rightsquigarrow} \pi_n'^*$.

Output: is $H(\pi_n^*) \subseteq H(\pi_n'^*)$ for all n?

Theorem
SIP-CONTAINMENT is undecidable.

Conclusion

- Decision problems on induction grammars are generally infeasible.
  - Even restricting the size of the vectors.
- **Open problem**: how complex is IND-COVER? (or TRATG-COVER, resp.?)

  Given a finite family of sets of terms $(L_n)_{n \in I}$ and $K \geq 0$, is there an induction grammar G with at most K productions such that $L(G, n) \supseteq L_n$ for all n?