Short Proofs are Hard to Find

Ian Mertz

University of Toronto

Joint work w/ Toniann Pitassi, Hao Wei

ICALP, July 10, 2019
Proof propositional complexity

Proof propositional complexity


How long is the shortest $P$-proof of $\tau$?
Proof propositional complexity

How long is the shortest $P$-proof of $\tau$?

Can we find short $P$-proofs of $\tau$?

Resolution

One of the simplest and most important proof systems
Resolution

One of the simplest and most important proof systems

- SAT solvers ([Davis-Putnam-Logemann-Loveland], [Pipatsrisawat-Darwiche])
- automated theorem proving
- model checking
- planning/inference
Resolution

Axioms:

\[ \overline{a} \lor d \\
\overline{b} \\
a \lor b \lor c \\
a \lor \overline{c} \]

\[ \phi = a \]

\[ \overline{a} \]

\[ b \]

\[ \overline{b} \]

\[ \overline{a} \lor b \]

\[ b \lor \overline{d} \]

\[ a \lor b \lor c \]

\[ a \lor \overline{c} \]

size = 11

width = 3
Automatizability [Bonet-Pitassi-Raz]

A proof system \( \mathcal{P} \) is \( f \)-automatizable if there exists an algorithm \( A : \text{UNSAT} \rightarrow \mathcal{P} \) that takes as input \( \tau \) and returns a \( \mathcal{P} \)-refutation of \( \tau \) in time \( f(n, S_\mathcal{P}(\tau)) \), where \( S_\mathcal{P}(\tau) \) is the size of the shortest \( \mathcal{P} \)-refutation of \( \tau \).
Automatizability [Bonet-Pitassi-Raz]

A proof system $\mathcal{P}$ is $f$-automatizable if there exists an algorithm $A : \text{UNSAT} \rightarrow \mathcal{P}$ that takes as input $\tau$ and returns a $\mathcal{P}$-refutation of $\tau$ in time $f(n, S_\mathcal{P}(\tau))$, where $S_\mathcal{P}(\tau)$ is the size of the shortest $\mathcal{P}$-refutation of $\tau$.

Automatizability is connected to many problems in computer science...

- theorem proving and SAT solvers
- algorithms for PAC learning ([Kothari-Livni], [Alekhnovich-Braverman-Feldman-Klivans-Pitassi])
- algorithms for unsupervised learning ([Bhattiprolu-Guruswami-Lee])
- approximation algorithms (many works...
Known automatizability lower bounds

General results and results for strong systems

- approximating $S_{\mathcal{P}}(\tau)$ to within $2^{\log^{1-o(1)} n}$ is NP-hard for all "reasonable" $\mathcal{P}$ ([Alekhnovich-Buss-Moran-Pitassi])
Known automatizability lower bounds

General results and results for strong systems

- approximating $S_P(\tau)$ to within $2^{\log^{1-o(1)} n}$ is NP-hard for all “reasonable” $P$ ([Alekhnovich-Buss-Moran-Pitassi])
- lower bounds against different Frege systems under cryptographic assumptions
  ([Bonet-Domingo-Gavalda-Maciel-Pitassi],[BPR],[Krajíček-Pudlák])
Known automatizability lower bounds

Results for weak systems

- first lower bounds against automatizability for Res, TreeRes by [Alekhnovich-Razborov]
Known automatizability lower bounds

Results for weak systems

- first lower bounds against automatizability for Res, TreeRes by [Alekhnovich-Razborov]
- extended to Nullsatz, PC by [Galesi-Lauria]
Known automatizability lower bounds

Results for weak systems

- first lower bounds against automatizability for Res, TreeRes by [Alekhnovich-Razborov]
- extended to Nullsatz, PC by [Galesi-Lauria]

Rest of this talk: a new version of [AR] + [GL]

- simplified construction and proofs
- stronger lower bounds via ETH assumption
- results also hold for Res(r)
Our results

**Theorem (Main Theorem)**

*Assuming ETH, \( \mathcal{P} \) is not \( n^{\tilde{o}(\log \log S_{\mathcal{P}}(\tau))} \)-automatizable for \( \mathcal{P} = \text{Res}, \text{TreeRes}, \text{Nullsatz}, \text{PC} \).*
Our results

**Theorem (Main Theorem)**

Assuming ETH, $\mathcal{P}$ is not $n^{\tilde{o}(\log \log S_{\mathcal{P}}(\tau))}$-automatizable for $\mathcal{P} = \text{Res}, \text{TreeRes}, \text{Nullsatz}, \text{PC}$.

**Theorem (Main Theorem for Res(r))**

Assuming ETH, Res(r) is not $n^{\tilde{o}(\log \log S_{\mathcal{P}}(\tau)/\exp(r^2))}$-automatizable for $r \leq \tilde{O}(\log \log \log n)$.
Our results

Theorem (Main Theorem)

Assuming ETH, \( \mathcal{P} \) is not \( n^{\tilde{\omega}(\log \log S_\mathcal{P}(\tau))} \)-automatizable for \( \mathcal{P} = \text{Res, TreeRes, Nullsatz, PC} \).

Theorem (Atserias-Muller’19)

Assuming \( \mathcal{P} \neq \text{NP} \), Res is not automatizable.
Assuming ETH, Res is not automatizable in subexponential time.
Our results

Theorem (Main Theorem)
Assuming ETH, \( \mathcal{P} \) is not \( n^{o(\log \log S_{\mathcal{P}}(\tau))} \)-automatizable for \( \mathcal{P} = \text{Res}, \text{TreeRes}, \text{Nullsatz}, \text{PC} \).

Theorem (Atserias-Muller’19)
Assuming \( P \neq \text{NP} \), \( \text{Res} \) is not automatizable.
Assuming ETH, \( \text{Res} \) is not automatizable in subexponential time.

Theorem (Bonet-Pitassi; Ben-Sasson-Wigderson)
TreeRes is \( n^{O(\log S_{\mathcal{P}}(\tau))} \)-automatizable.
Res is \( n^{O(\sqrt{n \log S_{\mathcal{P}}(\tau)})} \)-automatizable.
### Getting an automatizability lower bound

**Recipe:**

1. Hard gap problem $G$
2. Turn an instance of $G$ into a tautology $\tau$ such that
   - “yes” instances have small proofs
   - “no” instances have no small proofs
3. Run automatizing algorithm $Aut$ on $\tau$ and see how long the output is
Gap hitting set

$S = \{S_1 \ldots S_n\}$ over $[n]$
**Our results**

### Overview

**Gap hitting set**

- \( S = \{S_1 \ldots S_n\} \) over \([n]\)
- **hitting set**: \( H \subseteq [n] \) s.t. \( H \cap S_i \neq \emptyset \) for all \( i \in [n] \)

\( \gamma(S) \) is the size of the smallest gap hitting set: given \( S \),

distinguish whether \( \gamma(S) \leq k \) or \( \gamma(S) > k \)

**Theorem (Chen-Lin)**

Assuming ETH the gap hitting set problem cannot be solved in time \( n^{o(k)} \) for \( k = \tilde{O}(\log \log n) \)

---

Ian Mertz (U. of Toronto)

**Short Proofs are Hard to Find**

ICALP, July 10, 2019


**Gap hitting set**

- $S = \{S_1 \ldots S_n\}$ over $[n]$
- **hitting set:** $H \subseteq [n]$ s.t. $H \cap S_i \neq \emptyset$ for all $i \in [n]$
- $\gamma(S)$ is the size of the smallest $H$
- **Gap hitting set:** given $S$, distinguish whether $\gamma(S) \leq k$ or $\gamma(S) > k^2$

---

Theorem (Chen-Lin)

Assuming ETH the gap hitting set problem cannot be solved in time $n^{o(k)}$ for $k = \tilde{O}(\log \log n)$.
Our results

Overview

Gap hitting set

- \( S = \{S_1 \ldots S_n\} \) over \([n]\)
- **hitting set**: \( H \subseteq [n] \) s.t. \( H \cap S_i \neq \emptyset \) for all \( i \in [n] \)
- \( \gamma(S) \) is the size of the smallest \( H \)
- **Gap hitting set**: given \( S \), distinguish whether \( \gamma(S) \leq k \) or \( \gamma(S) > k^2 \)

**Theorem (Chen-Lin)**

*Assuming ETH the gap hitting set problem cannot be solved in time \( n^{\omega(k)} \) for \( k = \tilde{O}(\log \log n) \)*
From gap hitting set to automatizability

Theorem (Main Technical Lemma)

For $k = \tilde{O}(\log \log n)$, there exists a polytime algorithm mapping $S$ to $\tau_S$ s.t.

- if $\gamma(S) \leq k$ then $S_{\mathcal{P}}(\tau_S) \leq n^{O(1)}$
- if $\gamma(S) > k^2$ then $S_{\mathcal{P}}(\tau_S) \geq n^{\Omega(k)}$

where $\mathcal{P} \in \{\text{TreeRes}, \text{Res}, \text{Nullsatz}, \text{PC}\}$. 
Proof sketch of main theorem

Theorem (Main Theorem)

Assuming ETH, $\mathcal{P}$ is not $n^{\tilde{O}(\log \log \mathcal{S}_\mathcal{P}(\tau))}$-automatizable.

Proof: Let $Aut$ be the automatizing algorithm for $\mathcal{P}$ running in time $f(n, S) = n^{\tilde{O}(\log \log S)}$, and let $k = \tilde{O}(\log \log n)$. 
Proof sketch of main theorem

**Theorem (Main Theorem)**

Assuming ETH, $\mathcal{P}$ is not $n^{\tilde{O}(\log \log S_{\mathcal{P}}(\tau))}$-automatizable.

**Proof:** Let $\text{Aut}$ be the automatizing algorithm for $\mathcal{P}$ running in time $f(n, S) = n^{\tilde{O}(\log \log S)}$, and let $k = \tilde{\Theta}(\log \log n)$. 

\[ S \xrightarrow{\text{Main Technical Lemma}} T \xrightarrow{\text{run Aut for } n^{\tilde{O}(k)} \text{ steps}} \text{check if Aut outputs a valid } P\text{-ref of } T \]
Proof sketch of main theorem

Theorem (Main Theorem)
Assuming ETH, \( P \) is not \( n^{\tilde{o}(\log \log S_{P}(\tau))} \)-automatizable.

Proof: Let \( Aut \) be the automatizing algorithm for \( P \) running in time
\[ f(n, S) = n^{\tilde{o}(\log \log S)} \]
and let \( k = \tilde{\Theta}(\log \log n) \).

Theorem (Main Technical Lemma)
- If \( \gamma(S) \leq k \) then \( S_{P}(\tau) \leq n^{O(1)} \)
- If \( \gamma(S) > k^2 \) then \( S_{P}(\tau) \geq n^{\Omega(k)} \)
Proof sketch of main theorem

Theorem (Main Theorem)

Assuming ETH, $\mathcal{P}$ is not $n^{\tilde{o}(\log \log S_{\mathcal{P}}(\tau))}$-automatizable.

**Proof:** Let $\text{Aut}$ be the automatizing algorithm for $\mathcal{P}$ running in time $f(n, S) = n^{\tilde{o}(\log \log n)} = n^{o(k)}$, and let $k = \tilde{\Theta}(\log \log n)$.

Theorem (Main Technical Lemma)

- if $\gamma(S) \leq k$ then $S_{\mathcal{P}}(\tau) \leq n^{O(1)}$
- if $\gamma(S) > k^2$ then $S_{\mathcal{P}}(\tau) \geq n^{\Omega(k)}$
Proof sketch of main theorem

**Theorem (Main Theorem)**

Assuming ETH, $\mathcal{P}$ is not $n^{\tilde{o}(\log \log S_{\mathcal{P}}(\tau))}$-automatizable.

**Proof:** Let $Aut$ be the automatizing algorithm for $\mathcal{P}$ running in time $f(n, S) = n^{\tilde{o}(\log \log S)}$, and let $k = \tilde{\Theta}(\log \log n)$.

**Theorem (Main Technical Lemma)**

1. If $\gamma(S) \leq k$ then $S_{\mathcal{P}}(\tau) \leq n^{O(1)}$
2. If $\gamma(S) > k^2$ then $S_{\mathcal{P}}(\tau) \geq n^{\Omega(k)}$
Detour: universal sets

- $A_{m \times m}$ is $(m, q)$-universal if for all $I \subseteq [m]$, $|I| \leq q$, all $2^{|I|}$ possible column vectors appear in $A$ restricted to the rows $I$. 

![Diagram of a universal set matrix]
Detour: universal sets

- $A_{m \times m}$ is $(m, q)$-universal if for all $I \subseteq [m]$, $|I| \leq q$, all $2^{|I|}$ possible column vectors appear in $A$ restricted to the rows $I$

- $A_{m \times m}$ is $(m, q)$-dual universal if for all $J \subseteq [m]$, $|J| \leq q$, all $2^{|J|}$ possible row vectors appear in $A$ restricted to the columns $J$
Detour: universal sets

- $A_{m \times m}$ is $(m, q)$-universal if for all $I \subseteq [m], |I| \leq q$, all $2^{|I|}$ possible column vectors appear in $A$ restricted to the rows $I$.
- $A_{m \times m}$ is $(m, q)$-dual universal if for all $J \subseteq [m], |J| \leq q$, all $2^{|J|}$ possible row vectors appear in $A$ restricted to the columns $J$.
- constructions like the Paley graph work for $q = \frac{\log m}{4}$.
Defining $\tau_S$

Variables of $\tau_S$ will implicitly define two matrices using $A$ and $S$.
Defining $\tau_S$

Variables of $\tau_S$ will implicitly define two matrices using $A$ and $S$.
Defining $\tau_S$

Variables of $\tau_S$ will implicitly define two matrices using $A$ and $S$
Defining $\tau_S$

$\tau_S$ will state that there exist $\vec{\alpha}, \vec{\beta}$ such that there is no $i, j$ where $Q[i, j] = R[i, j] = 1$
Defining $\tau_S$

$\tau_S$ will state that there exist $\vec{\alpha}, \vec{\beta}$ such that there is no $i, j$ where $Q[i,j] = R[i,j] = 1$
Upper bound on $S_P(\tau_S)$

Lemma (Upper bound on $S_P(\tau_S)$)

If $\gamma(S) \leq k \leq \frac{\log m}{4}$, then $\tau_S$ is unsatisfiable and $S(\tau_S) \leq m^k n$ for TreeRes.

High-level idea: the universal property of $A$ guarantees some column of $Q$ will be a hitting set.
Upper bound on $S_P(\tau_S)$

Lemma (Upper bound on $S_P(\tau_S)$)

If $\gamma(S) \leq k \leq \log m$, then $\tau_S$ is unsatisfiable and $S(\tau_S) \leq m^k n$ for TreeRes.

High-level idea: the universal property of $A$ guarantees some column of $Q$ will be a hitting set.
Upper bound on $S_P(\tau_S)$

Lemma (Upper bound on $S_P(\tau_S)$)

If $\gamma(S) \leq k \leq \frac{\log m}{4}$, then $\tau_S$ is unsatisfiable and $S(\tau_S) \leq m^k n$ for TreeRes.

High-level idea: the universal property of $A$ guarantees some column of $Q$ will be a hitting set.
Upper bound on $S_P(\tau_S)$

Lemma (Upper bound on $S_P(\tau_S)$)

If $\gamma(S) \leq k \leq \frac{\log m}{4}$, then $\tau_S$ is unsatisfiable and $S(\tau_S) \leq m^k n$ for TreeRes.

High-level idea: the universal property of $A$ guarantees some column of $Q$ will be a hitting set.
Upper bound on $S_P(\tau_S)$

Lemma (Upper bound on $S_P(\tau_S)$)

If $\gamma(S) \leq k \leq \frac{\log m}{4}$, then $\tau_S$ is unsatisfiable and $S(\tau_S) \leq m^k n$ for TreeRes.

**High-level idea:** the universal property of $A$ guarantees some column of $Q$ will be a hitting set.

Size of the proof: $m^k n$
Upper bound on $S_P(\tau_S)$

Lemma (Upper bound on $S_P(\tau_S)$)

If $\gamma(S) \leq k \leq \frac{\log m}{4}$, then $\tau_S$ is unsatisfiable and $S(\tau_S) \leq n^2$ for TreeRes.

**High-level idea:** the universal property of $A$ guarantees some column of $Q$ will be a hitting set.

Size of the proof: $m^k n = n^2$ for $m = n^{1/k}$
Lower bound on $S_P(\tau_S)$

High-level idea 1: any proof $\pi$ must query all rows in some hitting set
Lower bound on $S_P(\tau_S)$

**High-level idea 1:** any proof $\pi$ must query all rows in some hitting set

- $\text{Res}/\text{TreeRes}$ - prover-delayer game [Pudlák, Atserias-Lauria-Nordström]
Lower bound on $S_P(\tau_S)$

**High-level idea 1:** any proof $\pi$ must query all rows in some hitting set

- Res/TreeRes - prover-delayer game [Pudlák, Atserias-Lauria-Nordström]
- Nullsatz/PC - linear operator [Galesi-Lauria]
Lower bound on $S_P(\tau_S)$

**High-level idea 1:** any proof $\pi$ must query all rows in some hitting set

- Res/TreeRes - prover-delayer game [Pudlák, Atserias-Lauria-Nordström]
- Nullsatz/PC - linear operator [Galesi-Lauria]
- Res(k) - switching lemma [Buss-Impagliazzo-Segerlend]
Lower bound on $S_P(\tau_S)$

**High-level idea 1:** any proof $\pi$ must query all rows in some hitting set

- Res/TreeRes - prover-delayer game [Pudlák, Atserias-Lauria-Nordström]
- Nullsatz/PC - linear operator [Galesi-Lauria]
- Res($k$) - switching lemma [Buss-Impagliazzo-Segerlend]
- TreeCP - lifting [upcoming work]
Lower bound on $S_P(\tau_S)$

**High-level idea 2:** $\pi$ knows nothing about a row or column without setting lots of variables
Lower bound on $S_P(\tau_S)$

**High-level idea 2:** $\pi$ knows nothing about a row or column without setting lots of variables

**Error-correcting codes**

- $x_i \in \{0, 1\}^{6 \log m}$,
  $y_j \in \{0, 1\}^{6 \log n}$
- $f_x : \{0, 1\}^{6 \log m} \rightarrow [m]$
- $f_y : \{0, 1\}^{6 \log n} \rightarrow [n]$
Open problems

Better hard $k$ in gap hitting set $\rightarrow$ better non-automatizability result
Open problems

Better hard $k$ in gap hitting set $\rightarrow$ better non-automatizability result

Theorem (Chen-Lin)

Assuming ETH the gap hitting set problem cannot be solved in time $n^{o(k)}$ for $k = O(\log^{1/7-o(1)} \log n)$

Theorem (Main Technical Lemma)

For $k = O(\sqrt{\log n})$, there exists a polytime algorithm mapping $S$ to $\tau_S$ . . .
Thank you!