Automatizability and Interpolation

Paul Beame

University of Washington

PCMI 2000 Thurs July 27
A proof system for a language $L$ is a polynomial time algorithm $V$ s.t.

- for all inputs $x$
  - $x \in L$ iff there exists a string $P$ s.t. $V$ accepts input $(x, P)$

think of $P$ as a proof that $x$ is in $L$ and $V$ as a proof verifier
Complexity of proof systems

- **Defn:** The complexity of a proof system \( V \) is a function \( f: \mathbb{N} \rightarrow \mathbb{N} \) defined by

\[
f(n) = \max_{x \in L, |x| = n} \min_{P: V \text{ accepts } (x, P)} |P|
\]

- i.e. how large \( P \) has to be as a function of \( |x| \)
- \( V \) is polynomially-bounded if its complexity is a polynomial function of \( n \)

- Definition says **nothing** about how costly it is to find short proofs!
  - lower bounds are even stronger that way
Automatizability (sic)

**Defn:** Given a proof system $V$ for $L$ and a function $f : \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$ we say that $V$ is $f(n,S)$-automatizable iff there is an algorithm $A_V$ s.t.

- given any input $x$ with $|x| = n$, if $x \in L$, $A$ outputs a proof $P$ in $V$ of this fact in time at most $f(n,S)$ where $S$ is the size of the shortest proof in $V$ that $x$ is in $L$.

We say that $V$ is automatizable iff it is $f(n,S)$-automatizable for some $f$ that is $n^{O(1)}S^{O(1)}$ i.e., can find a proof in time polynomial in the size of the smallest one.
Width & Automatizability

- **Theorem [BW]:** Every Davis-Putnam (DLL)/tree-like resolution proof of $F$ of size $S$ can be converted to one of width $\left\lceil \log_2 S \right\rceil + w(F)$.

- **Corollary [CEI][BP][BW]:** Tree-like resolution is $S^{O(\log n)}$-automatizable.

- **Proof:** There are only $2^{\log S \left( \frac{n}{\log S} \right)} = n^{O(\log S)}$ clauses of length at most $\log S$. Run breadth-first resolution only deriving clauses of width $\log S$. Can keep space requirements down by making it a recursive search.
Width, Resolution, and PCR

Theorem [BW] Every resolution proof of $F$ of size $S$ can be converted to one of width $O(\sqrt{n \log S} + w(F))$.

Corollary: General resolution is $2^{O(\sqrt{n \log S} \log n)}$-automatizable.

Theorem: Tree-PCR and PCR are $S^{O(\log n)}$-automatizable and $2^{O(\sqrt{n \log S} \log n)}$-automatizable respectively.

There are roughly $n^d$ monomials of degree at most $d$ & Groebner-basis like algorithm does linear algebra in that basis.
Interpolation

* Given formulas
  * $A(x, z)$ in variables $x$ and $z$
  * $B(y, z)$ in variables $y$ and $z$

* Defn: If $A(x, z) \lor B(y, z)$ is a tautology then an interpolant $C$ is a function s.t.
  * for any truth assignment $\zeta$ to $z$
    * $C(\zeta) = 0$ implies $A(x, \zeta)$ is a tautology
    * $C(\zeta) = 1$ implies $B(y, \zeta)$ is a tautology

* Also dual form if $A(x, z) \land B(y, z)$ is unsatisfiable
Interpolation - origin of the name

- **Given formulas**
  - $A(x,z)$ in free variables $x$ and $z$
  - $B(y,z)$ in free variables $y$ and $z$

- **Theorem: [Craig]** If $A(x,z) \rightarrow B(y,z)$ is a tautology then there is an interpolant $C$ with only free variables $z$ such that $A(x,z) \rightarrow C(z)$ and $C(z) \rightarrow B(y,z)$.

  - i.e. given $\neg A(x,z) \lor B(y,z)$: $C(z) \rightarrow B(y,z)$, $\neg C(z) \rightarrow \neg A(x,z)$
**Feasible Interpolation**

**Defn:** Given a propositional proof system \( V \) and a function \( f: \mathbb{N} \rightarrow \mathbb{N} \) we say that \( V \) has *f-interpolation* iff given an unsatisfiable formula of the form \( A(x,z) \land B(y,z) \) with proof size \( S \) in \( V \) there is a circuit of size at most \( f(S) \) computing an interpolant \( C \) for \( A(x,z) \land B(y,z) \); i.e. that says which of \( A(x,z) \) or \( B(y,z) \) is false.

- \( V \) has feasible interpolation iff \( f \) is polynomial.

- \( V \) has monotone f-interpolation iff whenever the variables \( z \) occur only negatively in \( B \) and only positively in \( A \), the circuit \( C \) is a monotone circuit.
Automatizability & Interpolation

**Lemma:** [Impagliazzo, BPR] If $V$ is automatizable then $V$ has feasible interpolation

**Proof:** Let $f$ be the polynomial function such that $V$ is $f$-automatizable and $A_V$ be the associated algorithm.

Given unsatisfiable $A(x, z) \land B(y, z)$ and an assignment $\zeta$ to $z$: 
Automatizability & Interpolation

**Lemma:** [Impagliazzo, BPR] If $V$ is automatizable then $V$ has feasible interpolation

**Proof:** Let $f$ be the polynomial function such that $V$ is $f$-automatizable and $A_V$ be the associated algorithm.

Given unsatisfiable $A(x, z) \land B(y, z)$ and an assignment $\zeta$ to $z$:

1. Run $A_V$ on input $A(x, z) \land B(y, z)$ to a proof $P$ of size $S' \leq f(S)$ where $S$ is the size of its optimal proof in $V$
**Lemma:** [Impagliazzo, BPR] If $V$ is automatizable then $V$ has feasible interpolation

**Proof:** Let $f$ be the polynomial function such that $V$ is $f$-automatizable and $A_V$ be the associated algorithm.

Given unsatisfiable $A(x, z) \land B(y, z)$ and an assignment $\zeta$ to $z$:

- Run $A_V$ on input $A(x, z) \land B(y, z)$ to a proof $P$ of size $S' \leq f(S)$ where $S$ is the size of its optimal proof in $V$
- Run $A_V$ on input $A(x, \zeta)$ for $f(S')$ steps
  - if it finds a proof output 0
  - else output 1
Automatizability & Interpolation

Lemma: [Impagliazzo, BPR] If $V$ is automatizable then $V$ has feasible interpolation.

Proof: Let $f$ be the polynomial function such that $V$ is $f$-automatizable and $A_V$ be the associated algorithm.

Given unsatisfiable $A(x, z) \land B(y, z)$ and an assignment $\zeta$ to $z$:

- Run $A_V$ on input $A(x, z) \land B(y, z)$ to a proof $P$ of size $S' \leq f(S)$ where $S$ is the size of its optimal proof in $V$.
- Run $A_V$ on input $A(x, \zeta)$ for $f(S')$ steps:
  - if it finds a proof output 0
  - else output 1

Note that if $B(y, \zeta)$ has satisfying assignment $\sigma$ then plugging $\sigma, \zeta$ into the proof $P$ yields a proof of size $S'$ of unsatisfiability of $A(x, \zeta) \land B(\sigma, \zeta)$ which is $A(x, \zeta) \land 1$. 
Interpolation and Resolution

- **Theorem:** [Krajicek] Resolution has feasible (monotone) interpolation.

- **Proof idea:** structure of proof allows one to decide easily which clauses cause unsatisfiability
Interpolation for Resolution

\( A(x,z) \) \( B(y,z) \)

- \( \neg x_1 \)
- \( \neg x_2 \)
- \( x_1 \lor x_2 \lor z \)
- \( y_1 \lor y_2 \lor \neg z \)
- \( \neg y_1 \)
- \( \neg y_2 \)
- \( x_1 \lor x_2 \lor y_1 \lor y_2 \)
- \( y_1 \lor y_2 \)
- \( y_2 \)
- \( \Lambda \)
Interpolation for Resolution

\[ A(x,z) \quad B(y,z) \]

\[ \neg x_1 \quad \neg x_2 \quad x_1 \lor x_2 \lor z \]

\[ x_1 \lor x_2 \lor y_1 \lor y_2 \]

\[ x_1 \lor y_1 \lor y_2 \]

\[ y_1 \lor y_2 \]

\[ \Lambda \]
Interpolation for Resolution

\[ A(x, z) \quad B(y, z) \]

\[ \neg x_1 \quad \neg x_2 \quad x_1 \lor x_2 \lor z \quad y_1 \lor y_2 \lor \neg z \]

\[ x_1 \lor x_2 \lor y_1 \lor y_2 \]

\[ y_1 \lor y_2 \]

\[ y_2 \]

\[ \Lambda \]

\[ z \leftarrow 1 \]
Interpolation for Resolution

A(x, z)

\neg x_1 \quad \neg x_2 \quad x_1 \lor x_2 \lor z

\neg y_1 \quad \neg y_2

B(y, 1)

y_1 \lor y_2

\neg y_1 \quad y_2

\neg y_1 \lor y_2

\neg x_1 \lor x_2 \lor y_1 \lor y_2

\neg y_1 \lor y_2

\neg y_2

\lor

\Lambda

y_2

z \leftarrow 1
Interpolation for Resolution

$$\neg x_1 \quad \neg x_2 \quad x_1 \lor x_2 \lor 1$$

$$y_1 \lor y_2$$

$$\neg y_1 \quad \neg y_2$$

$$y_2$$

$$\Lambda$$
Interpolation for Resolution

\[
A(x, 1) \quad B(y, 1)
\]

\[
\neg x_1 \quad \neg x_2 \quad 1 \quad y_1 \lor y_2 \quad \neg y_1 \quad \neg y_2
\]

\[
x_1 \lor y_1 \lor y_2 \quad y_1 \lor y_2 \quad y_2
\]

\[
\Lambda
\]

z \leftarrow 1
Interpolation for Resolution

\[ A(x,1) \]

\[ \neg x_1 \]
\[ \neg x_2 \]

\[ \neg x_1 \lor y_1 \lor y_2 \]
\[ x_1 \lor y_1 \lor y_2 \]
\[ y_1 \lor y_2 \]
\[ \Lambda \]

\[ B(y,1) \]
\[ y_1 \]
\[ y_2 \]
\[ z \leftarrow 1 \]
Interpolation for Resolution

\[ A(x,1) \rightarrow \neg x \rightarrow \neg x \rightarrow y \rightarrow y \rightarrow y \rightarrow y \rightarrow R \rightarrow 1 \]

\[ B(y,1) \rightarrow y \rightarrow y \rightarrow y \rightarrow y \rightarrow y \rightarrow y \rightarrow y \rightarrow y \rightarrow R \rightarrow 1 \]
Interpolation for Resolution

\[ A(x, 1) \]

\[ \neg x_1 \quad \neg x_2 \]

\[ B(y, 1) \]

\[ y_1 \lor y_2 \quad \neg y_1 \quad \neg y_2 \]

\[ x \lor y \]

\[ z \leftarrow 1 \]

Obtain a refutation of \( B(y, 1) \)

Easy to find given original proof
Interpolation and Lower Bounds

General proof strategy:

- Given
  - a class of circuits for which one has lower bounds
  - a proof system whose interpolants are in the class

- Build
  - a formula whose interpolant will be a circuit for a hard problem in the circuit class
Interpolation and Lower Bounds

- **Theorem:** If proof system $V$ has feasible interpolation and $\mathsf{NP} \not\subseteq \mathsf{P/poly}$ then $V$ is not polynomially bounded.

- **Theorem:** [BPR] Any proof system $V$ that has monotone feasible interpolation is not polynomially bounded.
Interpolation & NP vs P/poly

Proof sketch: Suppose that $V$ has feasible interpolation and is polynomially bounded with bound $p$

Consider formula $A(x, z) \land B(y, z)$ where

- $z$ represents a CNF formula
- $A(x, z)$ says that assignment $x$ satisfies $z$
- $B(y, z)$ says that $y$ - of length $p(|x|)$ - is a proof in $V$ that $z$ is unsatisfiable

Feasible interpolation for this formula will give a polysize circuit for deciding satisfiability
Clique-coloring formulas

- Formula $A(x, z) \land B(y, z)$ where
  - $z$ are the $n(n-1)/2$ variables representing an $n$ node graph $G(z)$
  - $A(x, z)$ is the statement that $G(z)$ has a $k$-clique
    $$(V_v x_{iv}) \land (\neg x_{iv} \lor \neg x_{ju} \lor z_{uv})$$
    $$(\neg x_{iv} \lor \neg x_{jv}) \land (\neg x_{iu} \lor \neg x_{iv})$$
  - $B(y, z)$ is the statement that $G(z)$ is $(k-1)$-colorable
    $$(V_i y_{vi}) \land (\neg z_{uv} \lor \neg y_{ui} \lor \neg y_{vi})$$
    $$(\neg y_{vi} \lor \neg y_{vj})$$
Interpolation examples

- **Theorem:** [Krajicek] Resolution has feasible (monotone) interpolation.

- **Theorem:** [Pudlak 95] Cutting Planes has feasible (monotone) interpolation where the interpolants are circuits over the real numbers
  - Also extended monotone lower bounds for clique to real circuits

- **Corollary:** Any Cutting Planes proofs of clique-coloring formulas are exponential

- **Theorem:** Polynomial calculus has feasible interpolation
Limitations of Interpolation

- **Theorem:** [KP] If one-way functions exist then Frege systems do not have feasible interpolation.

- **Theorem:** [BPR, Bonet et al] If factoring Blum integers is hard then any proof system that can polynomially simulate TC\(^0\)-Frege, or even AC\(^0\)-Frege does not have feasible interpolation.
Proof idea

- Suppose one has a method of **key agreement**
  - Given two people, one with $x$ and another with $y$
  - via exchanging messages, they agree on a secret key $\text{key}(x, y)$ so that even listening to their conversation without knowing $x$ or $y$ it is hard to figure out what even a bit of $\text{key}(x, y)$ is

- The common variables $z$ will represent the transcript of their conversation
  - $A(x, z)$ will say that the player with $x$ correctly computed its side of the conversation and the last bit of $\text{key}(x, y)$ is 0
  - $B(y, z)$ will say that the player with $y$ correctly computed its side of the conversation and the last bit of $\text{key}(x, y)$ is 1
Connections with proof systems

- Must encode the computation of each player in such a way that the proof system can prove given $x$ and $z$ what the value of the bit is.
- Can extend $x$ by helper extension variables to make the task easier.

  - The actual proof uses Diffie-Hellman secret key exchange which is as hard as factoring.
  - That requires powering which is not in $TC^0$ but the extension variables make it easy enough to prove.
The Interpolation Line

- Extended Frege
- Frege
- $TC^0$-Frege
- $AC^0$-Frege
- Cutting Planes
- Resolution
- Davis-Putnam
- Truth Tables
- PCR
- Polynomial Calculus
- Nullstellensatz