An Introduction to Proof Complexity, Part II.

Pavel Pudlák
Mathematical Institute, Academy of Sciences, Prague
and
Charles University, Prague

Computability in Europe 2009, Heidelberg

Overview

Part I.

- A lower bound for the propositional Pigeon-Hole Principle.
- Theories and complexity classes.
- Conditional and relativized separation of theories.

Part II.

- Propositional proof systems.
- Feasible interpolation.
- Unprovability of circuit lower bounds.
- Total search problems. (not presented at CiE)
- Feasible incompleteness. (not presented at CiE)

Propositional Proof Systems

The idea of a general propositional proof system

1. it is sound;
2. it is complete;
3. the relation *'D is a proof of tautology ϕ"* is decidable in *polynomial time.*

Definition (Cook, 1975)

Let *TAUT* be a set of tautologies. A *proof system* for *TAUT* is any polynomial time computable function f that maps the set of all binary strings $\{0,1\}^*$ onto *TAUT*.

Meaning:
Every string is a proof.
$f(\bar{a})$ is the formula of which $\bar{a}$ is a proof.

Say that *a proof system is polynomially bounded*, if every tautology has a proof of polynomial length.

Fact

There exists a polynomially bounded proof system iff **NP** $=$ **coNP**.

Meaning:
Every string is a proof.
$f(\bar{a})$ is the formula of which $\bar{a}$ is a proof.

Definition

A proof system f_1 *polynomially simulates* a proof system f_2, if there exists a polynomial time computable function g such that for all $\bar{a} \in \{0,1\}^*$, $f_1(g(\bar{a})) = f_2(\bar{a})$.

Meaning:
Given a proof $\bar{a}$ of $f_2(\bar{a})$ in the second system, we can construct a proof $g(\bar{a})$ of the same tautology in the first system in polynomial time.

Meaning:
Every string is a proof.
$f(\bar{a})$ is the formula of which $\bar{a}$ is a proof.

Definition

A proof system f_1 *polynomially simulates* a proof system f_2, if there exists a polynomial time computable function g such that for all $\bar{a} \in \{0,1\}^*$, $f_1(g(\bar{a})) = f_2(\bar{a})$.

Meaning:
Given a proof $\bar{a}$ of $f_2(\bar{a})$ in the second system, we can construct a proof $g(\bar{a})$ of the same tautology in the first system in polynomial time.

Theorem (Cook-Reckhow)

Frege systems polynomially simulate each other.

Extended Frege proof systems

Extension Rule: Whenever needed, introduce formula

$$p \equiv \phi,$$

where p is a new variable that does not occur in the conclusion nor in ϕ and ϕ is otherwise arbitrary.

An *Extended Frege* proof system is a Frege system augmented with the extension rule.

Extended Frege systems polynomially simulate each other.

Extended Frege systems are polynomially equivalent to

1. *Substitution Frege* systems—Frege systems with the substitution rule.

2. *Circuit Frege* systems—Frege systems that use boolean circuits instead of formulas.

Extended Frege proof systems are the strongest proof systems whose soundness $\Theta_{\mathbf{P}}$ proves.

Theorem (Cook, 1975)

If P is a proof system whose soundness is provable in $\Theta_{\mathbf{P}}$, then Extended Frege proof systems polynomially simulate P.

In principle it is possible to prove $\Theta_{\mathbf{P}} \neq \Theta_{\mathbf{NP}}$ by proving superpolynomial lower bounds on Extended Frege proofs of explicitly defined tautologies.

Corollary

Suppose T is a polynomial time decidable set of tautologies such that tautologies from T do not have polynomial size proofs in an Extended Frege proof system. Further suppose that $\Theta_{\mathbf{NP}}$ proves that every element of T is a tautology. Then $\Theta_{\mathbf{P}} \neq \Theta_{\mathbf{NP}}$.

Theorem

If Extended Frege proof systems are not polynomially bounded, then $\Theta_{\mathbf{P}} \nvdash \mathbf{P} = \mathbf{NP}$.

More presicely, we have the following stronger theorem:

If there exists no polynomial time algorithm that for a given tautology constructs an extended frege proof, then for no Π^b_1 *formula* $\phi(x)$, $\Theta_{\mathbf{P}} \vdash \forall x\ Sat(x) \equiv \phi(x)$.

Proof. Suppose $\Theta_{\mathbf{P}} \vdash \forall x\ Sat(x) \equiv \phi(x)$. Then

$$\Theta_{\mathbf{P}} \vdash \forall x\ Sat(x) \vee \neg\phi(x).$$

Since $Sat(x) \vee \neg\phi(x)$ is a Σ^b_1 formula, by Buss's Theorem the existential quantifiers can be witnessed by a polynomial time computable function.

Since $\neg\phi(\neg x) \equiv \neg Sat(\neg x)$, the formula $\neg\phi(\neg x)$ defines a plynomially bounded proof system.

By Cook's theorem, this proof system is polynomially simulated by an Extended Frege system.

Hence proofs in an Extenden Frege system can be found in polynomial time. □

Some propositional proof systems

- Resolution
- depth 2 Frege systems
- depth 3 Frege systems
- $\vdots$

.... $\uparrow$ exp. lower bounds

- Frege systems
- Extended Frege systems

Feasible interpolation

Theorem (Craig)

Let $\Phi(\bar{p},\bar{q}) \rightarrow \Psi(\bar{p},\bar{r})$ be a tautology, where $\bar{p},\bar{q},\bar{r}$ are disjoint sets of propositional variables, there exists a formula $I(\bar{p})$, which contains only the common variables $\bar{p}$, such that both $\Phi(\bar{p},\bar{q}) \rightarrow I(\bar{p})$ and $I(\bar{p}) \rightarrow \Psi(\bar{p},\bar{r})$ are tautologies.

Krajíček's Idea: If we have a short proof of $\Phi(\bar{p},\bar{q}) \rightarrow \Psi(\bar{p},\bar{r})$, we can bound the complexity of $I(\bar{p})$.
Thus we could reduce proving lower bounds on the lengths of proofs to lower bounds on circuit complexity.

This idea does not work for all proof systems. If it works, we say that the proof system has the *feasible interpolation property.*

$$\Phi(\bar{p},\bar{q}) \to I(\bar{p}) \text{ and } I(\bar{p}) \to \Psi(\bar{p},\bar{r})$$

is equivalent to

$$\Phi(\bar{p},\bar{q}) \to I(\bar{p}) \text{ and } \neg\Psi(\bar{p},\bar{r}) \to I(\bar{p}),$$

hence also to

$$\exists\bar{q}\ \Phi(\bar{p},\bar{q}) \to I(\bar{p}) \text{ and } \exists\bar{r}\ \neg\Psi(\bar{p},\bar{r}) \to \neg I(\bar{p}).$$

Also

$$\Phi(\bar{p},\bar{q}) \to \Psi(\bar{p},\bar{r})$$

is equivalent to

$$\neg\exists\bar{p}\ (\exists\bar{q}\ \Phi(\bar{p},\bar{q}) \wedge \exists\bar{r}\ \neg\Psi(\bar{p},\bar{r})).$$

Theorem (informal statement)

*If $A, B \in$ **NP** such that Resolution proves $A \cap B = \emptyset$ using proofs of polynomial size, then A and B can be separated by a set $C \in$ **P**/**poly**.*

Resolution is the proof system which uses elementary disjunctions i.e., disjunctions of literals, as formulas, and the cut rule as the only rule

$$\frac{\Gamma \vee p,\;\; \Delta \vee \neg p}{\Gamma \vee \Delta},$$

(where Γ, Δ are elementary disjunctions).
A *literal* is a variable or a negated variable.
Elementary disjunctions are called *clauses.*

Resolution is the proof system which uses elementary disjunctions i.e., disjunctions of literals, as formulas, and the cut rule as the only rule

$$\frac{\Gamma \vee p,\ \ \Delta \vee \neg p}{\Gamma \vee \Delta},$$

(where Γ, Δ are elementary disjunctions).
A *literal* is a variable or a negated variable.
Elementary disjunctions are called *clauses.*

The ternary connective *sel* (selector) is defined by $sel(0, x, y) = x$ and $sel(1, x, y) = y$.

Theorem (essentially - Krajíček, 1994)

Let P be a resolution proof of the empty clause from clauses $A_i(\bar{p}, \bar{q}), i \in I$, $B_j(\bar{p}, \bar{r}), j \in J$ where $\bar{p}, \bar{q}, \bar{r}$ are disjoint sets of propositional variables. Then there exists a circuit $C(\bar{p})$ such that for every 0-1 assignment $\bar{a}$ for $\bar{p}$

$C(\bar{a}) = 0 \quad \Rightarrow \quad A_i(\bar{a}, \bar{q}), i \in I$ *are unsatisfiable, and*

$C(\bar{a}) = 1 \quad \Rightarrow \quad B_j(\bar{a}, \bar{r}), j \in J$ *are unsatisfiable;*

the circuit C is in basis $\{0, 1, \wedge, \vee, sel\}$ and its underlying graph is the graph of the proof P.

Theorem (essentially - Krajíček, 1994)

Let P be a resolution proof of the empty clause from clauses $A_i(\bar{p}, \bar{q}), i \in I$, $B_j(\bar{p}, \bar{r}), j \in J$ where $\bar{p}, \bar{q}, \bar{r}$ are disjoint sets of propositional variables. Then there exists a circuit $C(\bar{p})$ such that for every 0-1 assignment $\bar{a}$ for $\bar{p}$

$C(\bar{a}) = 0 \quad \Rightarrow \quad A_i(\bar{a}, \bar{q}), i \in I$ *are unsatisfiable, and*

$C(\bar{a}) = 1 \quad \Rightarrow \quad B_j(\bar{a}, \bar{r}), j \in J$ *are unsatisfiable;*

the circuit C is in basis $\{0, 1, \wedge, \vee, sel\}$ and its underlying graph is the graph of the proof P.
Moreover, we can construct in polynomial time a resolution proof of the empty clause from clauses $A_i(\bar{a}, \bar{q}), i \in I$ if $C(\bar{a}) = 0$, respectively $B_j(\bar{a}, \bar{r}), j \in J$ if $C(\bar{a}) = 1$; the length of this proof is less than or equal to the length of P.

The idea of the proof.

$$\begin{array}{c} \ldots A_i(\bar{p},\bar{q})\ldots \quad \ldots B_j(\bar{p},\bar{r})\ldots \\ \ldots\ \ldots \\ \ldots \\ \emptyset \end{array}$$

Given an assignment $\bar{a}$ for $\bar{p}$, follow the proof but instead of resolving on variables p_i just substitute for them.

Thus the proof splits into two parts: one with only variables $\bar{q}$ the other with variables $\bar{r}$.

One of the parts contains the empty clause—this is the refutation of the corresponding part of initial clauses. E.g.

$$\begin{array}{cc} \ldots A_i(\bar{a},\bar{q})\ldots & \ldots B_j(\bar{a},\bar{r})\ldots \\ \ldots & \ldots \\ \ldots & \ldots \\ \emptyset & \end{array}$$

Thus we have a *polynomial time algorithm* for deciding which set of clauses is inconsistent.

Proof.

First we describe the transformation of the proof for a given assignment $\bar{p} \mapsto \bar{a}$. Once we get an assignment for $\bar{p}$, we can eliminate all these variables from the proof, but we shall do it in two stages, as we shall need more information about this process when constructing the circuit explicitly.

Let us call a clause *q-clause,* resp. *r-clause,* if it contains only variables $\bar{p}, \bar{q}$ resp. $\bar{p}, \bar{r}$. We shall call a clause q-clause, resp. r-clause also in the case that the clause contains only variables $\bar{p}$ or is empty, but its ancestors are q-clauses, resp. r-clauses.

1. In the first stage we replace each clause of P by a subclause so that each clause in the proof is either q-clause or r-clause. We start with the initial clauses, which are left unchanged and continue along the derivation P.

Case 1.

$$\frac{\Gamma \vee p_k, \ \Delta \vee \neg p_k}{\Gamma \vee \Delta}$$

and we have replaced $\Gamma \vee p_k$ by Γ' and $\Delta \vee \neg p_k$ by Δ'. Then we replace $\Gamma \vee \Delta$ by Γ' if $p_k \mapsto 0$ and by Δ' if $p_k \mapsto 1$.

Case 2.

$$\frac{\Gamma \vee q_k,\ \Delta \vee \neg q_k}{\Gamma \vee \Delta}$$

and we have replaced $\Gamma \vee q_k$ by Γ' and $\Delta \vee \neg q_k$ by Δ'. If one of $\Gamma',\ \Delta'$ is an r-clause, then it does not contain q_k, and we replace $\Gamma \vee \Delta$ by this clause. If both Γ' and Δ' are q-clauses, we resolve along q_k, or take one, which does not contain q_k.

Case 3.

$$\frac{\Gamma \vee r_k,\ \Delta \vee \neg r_k}{\Gamma \vee \Delta}.$$

This is the dual case to case 2.

2. Now delete the clauses which contain a $\bar{p}$ literal with value 1, and remove all $\bar{p}$ literals from the remaining clauses. Thus we get a valid derivation of the final empty clause from the reduced initial clauses. If this final clause is a q-clause, the proof contains a subproof using only the reduced clauses $A_i, i \in I$; if it is an r-clause, the proof contains a subproof using only the reduced clauses $B_j, j \in J$.

3. The circuit C is constructed so that the value computed at a gate corresponding to a clause Γ will determine if it is transformed into a q-clause or an r-clause. We assign 0 to q-clauses and 1 to r-clauses.
Thus the circuit is constructed as follows. Put constant 0 gates on clauses $A_i, i \in I$ and constant 1 gates on clauses $B_j, j \in J$. Now consider three cases as above.

Case 1. If the gate on $\Gamma \vee p_k$ gets value x and the gate on $\Delta \vee \neg p_k$ gets value y, then the gate on $\Gamma \vee \Delta$ should get the value $z = sel(p_k, x, y)$. Thus we place the *sel* gate on $\Gamma \vee \Delta$.

Case 2. If the gate on $\Gamma \vee q_k$ gets value x and the gate on $\Delta \vee \neg q_k$ gets value y, then the gate on $\Gamma \vee \Delta$ should get the value $z = x \vee y$. Thus we place the $\vee$ gate on $\Gamma \vee \Delta$.

Case 3. This is dual to case 2, so we place the $\wedge$ gate on $\Gamma \vee \Delta$. $\square$

We cannot prove lower bounds using this theorem without using unproven assumptions, because we are not able to prove lower bounds on the size of boolean circuits.

Monotone feasible interpolation for Resolution

Theorem

Let P be a resolution proof of the empty clause from clauses $A_i(\bar{p}, \bar{q}), i \in I$, $B_j(\bar{p}, \bar{r}), j \in J$ where $\bar{p}, \bar{q}, \bar{r}$ are disjoint sets of propositional variables.
Suppose moreover that either all variables $\bar{p}$ occur in $A_i(\bar{p}, \bar{q}), i \in I$ only positively or all variables $\bar{p}$ occur in $B_i(\bar{p}, \bar{r}), j \in J$ only negatively.
Then there exists a monotone circuit $C(\bar{p})$ such that for every 0-1 assignment $\bar{a}$ for $\bar{p}$

$$C(\bar{a}) = 0 \quad \Rightarrow \quad A_i(\bar{a}, \bar{q}), i \in I \text{ are unsatisfiable, and}$$

$$C(\bar{a}) = 1 \quad \Rightarrow \quad B_j(\bar{a}, \bar{r}), j \in J \text{ are unsatisfiable;}$$

the size of the circuit C is at most 3-times the size of the proof P.

Proof
Assume that all $\bar{p}$'s are positive in clauses $A_i, i \in I$. This property is then inherited to the transformed q-clauses. Hence in case 1, if Δ' is a q-clause, it cannot contain $\neg p_k$ and we can take it for $\Gamma \vee \Delta$, even if $p_k \mapsto 0$. Thus we can replace $sel(p_k, x, y)$ by $(p_k \vee x) \wedge y$ which is monotone and differs from the selector exactly on one input $(p_k = 0, x = 1, y = 0)$ which corresponds to the above situation.

Application — an exponential lower bound on Resolution proofs

A. Haken, 1985, an exponential lower bound on Resolution proofs of PHP.
A.A. Razborov, 1985, an exponential lower bound on monotone circuits.

Theorem (Razborov, Boppana-Alon)

Let $m = \binom{n}{2}$, let $k = n^{1/4}$. Think of $\bar{a} \in \{0,1\}^m$ as graphs on n vertices. Let C be a monotone *boolean circuit with m input variables such that*

$\bar{a}$ encodes a graph with a clique of size $k \Rightarrow C(\bar{a}) = 1$,

$\bar{a}$ encodes a $k-1$ colorable graph $\Rightarrow C(\bar{a}) = 0$.

Then the size of C is at least $2^{n^{\epsilon}}$.

Corollary

*Let $\Phi(\bar{p}, \bar{q})$ express "$\bar{q}$ is not a clique of size k in graph $\bar{p}$",
Let $\Psi(\bar{p}, \bar{r})$ express "$\bar{r}$ is not a $k-1$-coloring of graph $\bar{p}$".
Then every Resolution proof of the tautology $\Phi(\bar{p}, \bar{q}) \vee \Psi(\bar{p}, \bar{r})$ has size at least $\frac{1}{3}2^{n^{\epsilon}}$.*

Definition

A proof system P has the *feasible interpolation property*, if there is a polynomial time algorithm such that given an implication of the form $\phi(\bar{p},\bar{q}) \rightarrow \psi(\bar{p},\bar{r})$ and its proof d in the proof system P, the algorithm constructs a circuit $C(\bar{p})$ that interpolates the implication. In particular, the size of the circuit C is bounded by a polynomial in the length of the proof d.

Definition

A proof system P has the *feasible interpolation property*, if there is a polynomial time algorithm such that given an implication of the form $\phi(\bar{p}, \bar{q}) \rightarrow \psi(\bar{p}, \bar{r})$ and its proof d in the proof system P, the algorithm constructs a circuit $C(\bar{p})$ that interpolates the implication. In particular, the size of the circuit C is bounded by a polynomial in the length of the proof d.

Proof systems with the feasible interpolation property

- Polynomial Calculus
- Cutting Planes
- Lovasz-Schrijver

Definition

A proof system P has the *feasible interpolation property*, if there is a polynomial time algorithm such that given an implication of the form $\phi(\bar{p}, \bar{q}) \to \psi(\bar{p}, \bar{r})$ and its proof d in the proof system P, the algorithm constructs a circuit $C(\bar{p})$ that interpolates the implication. In particular, the size of the circuit C is bounded by a polynomial in the length of the proof d.

Proof systems with the feasible interpolation property

- Polynomial Calculus
- Cutting Planes
- Lovasz-Schrijver

Proof systems that do not have the feasible interpolation property

Assuming that factoring is hard:

- bounded depth Frege and stronger systems

Monotone interpolation for Cutting Planes

Cutting planes proof system

Propositional variables $p_1, p_2, \ldots$ with the interpretation $0 = \textit{false}$ and $1 = \textit{true}$.
A proof line is an inequality

$$\sum_k c_k p_k \geq C,$$

where c_k and C are integers.

The *axioms* are $p_k \geq 0$ and $-p_k \geq -1$ (i.e. $0 \leq p_k \leq 1$) for every propositional variable p_k.
The *rules* are

1. **addition:** from $\sum_k c_k p_k \geq C$ and $\sum_k d_k p_k \geq D$ derive $\sum_k (c_k + d_k) p_k \geq C + D$;
2. **multiplication:** from $\sum_k c_k p_k \geq C$ derive $\sum_k d c_k p_k \geq dC$, where d is an arbitrary positive integer;
3. **division:** from $\sum_k c_k p_k \geq C$ derive $\sum_k \frac{c_k}{d} p_k \geq \left\lceil \frac{C}{d} \right\rceil$, provided that $d > 0$ is an integer which divides each c_k.

Definition

A *monotone real circuit* is a circuit which computes with real numbers and uses arbitrary nondecreasing real unary and binary functions as gates.
We say that a monotone real circuit computes a boolean function, if for all inputs of 0's and 1's the circuit outputs 0 or 1.

Examples of monotone real gates: $\max, \min, +, e^x$; also $\times$ on $\mathbb{R}^+$.

Theorem (Pudlák 1997)

Let P be a cutting plane proof of the contradiction $0 \geq 1$ from inequalities

$$\sum_k c_{i,k} p_k + \sum_l b_{i,l} q_l \geq A_i, i \in I,$$

$$\sum_k c'_{j,k} p_k + \sum_m d_{j,m} r_m \geq B_j, j \in J.$$

Suppose that all the coefficients $c_{i,k}$ are nonnegative, or all the coefficients $c'_{i,k}$ are nonpositive, then one can construct a real monotone *interpolating circuit C (computing a boolean function) whose size is bounded by a linear function in the number of variables and the number of inequalities of the proof P.*

Theorem (Pudlák 1997)

Let $m = \binom{n}{2}$, *let* $k = n^{1/4}$.
Let C *be a* monotone real circuit *with* m *input variables such that*

$\bar{a}$ *encodes a graph with a clique of size* $k \Rightarrow C(\bar{a}) = 1$,

$\bar{a}$ *encodes a* $k - 1$ *colorable graph* $\Rightarrow C(\bar{a}) = 0$.

Then the size of C *is at least* 2^{n^δ}.

Corollary

Let $\Phi(\bar{p}, \bar{q})$ *express "* $\bar{q}$ *is not a clique of size* k *in graph* $\bar{p}$ *",*
Let $\Psi(\bar{p}, \bar{r})$ *express "* $\bar{r}$ *is not a* $k - 1$*-coloring of graph* $\bar{p}$ *".*
Then every Cutting-Planes proof of the tautology $\Phi(\bar{p}, \bar{q}) \vee \Psi(\bar{p}, \bar{r})$ *has size at least* 2^{n^δ}.

Independently, Cook and Haken proved similar bounds for Broken Mosquito Screen pair of disjoint **NP** sets.

Corollary

No algorithm based only on cutting planes can solve the Integer Linear Programing problem in subexponential time.

Corollary

No algorithm based only on cutting planes can solve the Integer Linear Programing problem in subexponential time.

Corollary (of the lower bounds on tree-like resolution proofs)

No algorithm based Davis-Putnam procedure can solve $3 - SAT$ in time less than 2^{ϵ} for some $\epsilon > 0$.

Monotone boolean circuits

type of circuit	lower bounds	proof system
monotone circuits with $\wedge, \vee$	yes	Resolution
real monotone circuits	yes	Cutting Planes
monotone span programs	yes	Polynomial Calculus
monotone linear programs	no	Lovasz-Schrijver
???		???

A monotone LP programs P:

$$\sum_j a_{ij} z_j \leq \sum_k b_{ik} x_k + c_i$$

$a_{ij}, b_{i,k}, c_i \in \mathbb{R}$ constants; $z_j \in \mathbb{R}^+$, $x_k \in \{0,1\}$ variables

P computes boolean function $f(\bar{x})$, if for every assignment to $\bar{x}$

$$P \text{ has a solution } \bar{z} \quad \Leftrightarrow \quad f(\bar{x}) = 1$$

Monotone LP polynomially simulate monotone circuits and monotone span programs over reals.

Problem

Why it is easier to prove lower bounds for monotone circuits (of various types)?

Problem

Can one find a type of monotone circuit for every proof system that has the feasible interpolation property? And can one prove a superpolynomial lower bound?

Problem

Prove a superpolynomial lower bound on monotone LP.

Unprovability of circuit lower bounds

Theorem (Razborov 1995, Krajíček)

Assuming PRG-Conjecture the following is true. If P is a propositional proof system with the feasible interpolation property, then for every boolean function f it is hard to prove in P an exponential lower bound on the circuit complexity of f.

PRG-Conjecture

There exists an $\epsilon > 0$ and a polynomial time computable function (the PRG generator) $G_n : \{0,1\}^n \to \{0,1\}^{2n}$ such that for every circuit C of size $\leq 2^{n^\epsilon}$

$$|Prob(C(G_n(\mathbf{x})) = 1 - Prob(C(\mathbf{y}) = 1)| \geq 2^{-n^\epsilon}.$$

Unprovability of circuit lower bounds

Theorem (Razborov 1995, Krajíček)

Assuming PRG-Conjecture the following is true. If P is a propositional proof system with the feasible interpolation property, then for every boolean function f it is hard to prove in P an exponential lower bound on the circuit complexity of f.

PRG-Conjecture

There exists an $\epsilon > 0$ and a polynomial time computable function (the PRG generator) $G_n : \{0,1\}^n \to \{0,1\}^{2n}$ such that for every circuit C of size $\leq 2^{n^\epsilon}$

$$|Prob(C(G_n(\mathbf{x})) = 1 - Prob(C(\mathbf{y}) = 1)| \geq 2^{-n^\epsilon}.$$

Theorem (Razborov-Rudich,1994)

PRG-Conjecture implies that there is no polynomial time computable property of boolean functions that implies superpolynomial lower bounds and is satisfied by $\geq 1/2$ of all boolean functions.

View a boolean function of n variables as a 0-1 string of length 2^n.

PRG-Conjecture $\Rightarrow$ there exists a pseudorandom *function* generator.

Idea

Let $N = 2^n$. Think of $\bar{x} \in \{0,1\}^N$ as boolean functions of n variables.
Let $\alpha_M(\bar{p}, \bar{q})$ be a boolean formula expressing:

"the function coded by $\bar{p}$ has a circuit of size M (coded by $\bar{q}$)"

Thus $|\bar{p}| = N$ and $|\bar{q}| \approx M \log M$.

Suppose P is a proof system with the feasible interpolation property.
Suppose for some function $\bar{a} \in \{0,1\}^N$, there is a poly-size P-proof of the circuit lower bound $2M+1$, i.e., of the tautology $\neg\alpha_{2M+1}(\bar{a}, \bar{q})$.
Then we also have a poly-size proof of

$$\neg\alpha_M(\bar{p}, \bar{q}) \vee \neg\alpha_M(\bar{p} \oplus \bar{a}, \bar{r}).$$

Idea

Let $N = 2^n$. Think of $\bar{x} \in \{0,1\}^N$ as boolean functions of n variables.
Let $\alpha_M(\bar{p}, \bar{q})$ be a boolean formula expressing:

"the function coded by $\bar{p}$ has a circuit of size M (coded by $\bar{q}$)"

Thus $|\bar{p}| = N$ and $|\bar{q}| \approx M \log M$.

Suppose P is a proof system with the feasible interpolation property.
Suppose for some function $\bar{a} \in \{0,1\}^N$, there is a poly-size P-proof of the circuit lower bound $2M+1$, i.e., of the tautology $\neg\alpha_{2M+1}(\bar{a}, \bar{q})$.
Then we also have a poly-size proof of

$$\neg\alpha_M(\bar{p}, \bar{q}) \vee \neg\alpha_M(\bar{p} \oplus \bar{a}, \bar{r}).$$

By feasible interpolation, there exists a poly-size circuit C such that for all $\bar{b} \in \{0,1\}^N$

$$C(\bar{b}) = 0 \quad \Rightarrow \quad \neg\alpha_M(\bar{b}, \bar{q})$$

$$C(\bar{b}) = 1 \quad \Rightarrow \quad \neg\alpha_M(\bar{b} \oplus \bar{a}, \bar{r})$$

$$C(\bar{b}) = 0 \quad \Rightarrow \quad \neg\alpha_M(\bar{b}, \bar{q})$$
$$C(\bar{b}) = 1 \quad \Rightarrow \quad \neg\alpha_M(\bar{b} \oplus \bar{a}, \bar{r})$$

Hence:

- if $|\{\bar{b};\ C(\bar{b}) = 0\}| \geq \frac{1}{2}2^N$, then "$C(\bar{x}) = 0$" defines a property of functions of complexity $\geq M$ satisfied by $1/2$ of all functions,
- otherwise "$C(\bar{x} \oplus \bar{a}) = 1$" is such a property.

Assuming the PRG-Conjecture, if M is superpolynomial in n (i.e., $M = n^{\omega(1)}$), then this is not possible.
So it is not possible to prove a superpolynomial lower bound using a polynomial size proof (i.e. of size $N^c = 2^{cn}$ for c constant).

Theorem

Assume the PRG-Conjecture. If a proof system P has the feasible interpolation property, then there are no polynomial size P-proofs of superpolynomial circuit lower bounds.

Theorem

Assume the PRG-Conjecture. If a proof system P has the feasible interpolation property, then there are no polynomial size P-proofs of superpolynomial circuit lower bounds.

Theorem (Krajíček 2004)

Superpolynomial circuit lower bounds are hard for Resolution.

Without PRG-Conjecture!

Theorem

Assume the PRG-Conjecture. If a proof system P has the feasible interpolation property, then there are no polynomial size P-proofs of superpolynomial circuit lower bounds.

Theorem (Krajíček 2004)

Superpolynomial circuit lower bounds are hard for Resolution.

Without PRG-Conjecture!

Problem

Feasible interpolation holds only for weak propositional proof systems. Can one prove such lower bounds for other propositional proof systems?

Likely for bounded depth Frege systems.

Unprovability of lower bounds in first order theories.

Formalization in $\Theta_{\mathbf{NP}}[R]$:

$N = 2^n$ the length of the truth table;
$t = t(n) = n^{\omega(1)}$ a superpolynomial lower bound;
N and bounded formula $\sigma(x)$ defines a (the truth table of) boolean function $f : \{0,1\}^n \to \{0,1\}$ by: $f(\bar{x}) = 1 \Leftrightarrow \sigma(\bar{x})$;
$R|_{t \log t}$ encodes a circuit C of size t;
$LB(t, \sigma, R)$ is the formalization of: *"the circuit coded by R does not compute the boolean function coded by σ."*

Corollary (Razborov 1995)

Assuming the PRG-Conjecture, no superpolynomial lower bound on circuit complexity is provable in $\Theta_{\mathbf{NP}}[R]$.

Unprovability of lower bounds in first order theories.

Formalization in $\Theta_{\mathbf{NP}}[R]$:

$N = 2^n$ the length of the truth table;
$t = t(n) = n^{\omega(1)}$ a superpolynomial lower bound;
N and bounded formula $\sigma(x)$ defines a (the truth table of) boolean function $f : \{0,1\}^n \to \{0,1\}$ by: $f(\bar{x}) = 1 \Leftrightarrow \sigma(\bar{x})$;
$R|_{t \log t}$ encodes a circuit C of size t;
$LB(t, \sigma, R)$ is the formalization of: *"the circuit coded by R does not compute the boolean function coded by σ."*

Corollary (Razborov 1995)

Assuming the PRG-Conjecture, no superpolynomial lower bound on circuit complexity is provable in $\Theta_{\mathbf{NP}}[R]$.

Krajíček observed that in fact it is easy to prove this theorem by showing that it is consistent with $\Theta_{\mathbf{NP}}$ that some formula defines a mapping from N to n^c. In such a model every function is computable by a polynomial size DNF.
Razborov also considered *"split theories"* in particular:

$$\Theta_{\mathbf{PH}}[R] + \Theta_{\mathbf{PH}}[S] + \Theta_{\mathbf{NP}}[R, S]$$

where PH stands for Polynomial Hierarchy.

Razborov also considered *"split theories"* in particular:

$$\Theta_{\mathbf{PH}}[R] + \Theta_{\mathbf{PH}}[S] + \Theta_{\mathbf{NP}}[R, S]$$

where PH stands for Polynomial Hierarchy.
In this theory one can prove exponential lower bounds for bounded depth circuit computing the parity. (To this end Razborov invented his proof of the switching lemma.)

Theorem (Razborov 1995)

Assuming the PRG-Conjecture, no superpolynomial lower bound on circuit complexity is provable in the theory above.

Formalization of a circuit C in split theories:

$$C = C_1 \oplus C_2,$$

where C_1 is coded by R and C_2 is coded by S.

Total NP search problems **TFNP**

Definition

A search problem is given by a relation R such that

1. $R(x, y) \in \mathbf{P}$;
2. there is a polynomial p such that $R(x, y)$ implies $|y| \leq p(|x|)$;
3. $\forall x \exists y R(x, y)$.

The problem is: given input x, find y such that $R(x, y)$.

Why they are important:

- classification of functional problems;
- characterization of $\forall\Sigma_1$ theorems of bounded arithmetical theories.

Basic facts

1. **P** $=$ **NP** implies that every **TFNP** can be solved in polynomial time.
2. If every **TFNP** can be solved in polynomial time, then **NP** $\cap$ **coNP** $=$ **P**.

Reductions between TFNP search problems

Let S_i be a search problem determined by $R_i(x, y)$. Then S_1 is

1. *Turing reducible* to S_2 if there exists a polynomial time algorithm that queries S_2 for solutions and solves S_1;
2. *many-one reducible* to S_2 if there exist polynomial time computable functions f and g such that given x, f computes some string $f(x) = x'$ such that if $R_2(x', y')$ for some y', then $R_1(x, g(x, y'))$ ($\Leftrightarrow$ Turing reducible using a single query).

Conjecture

There is no complete **TFNP** *search problem.*

Classes of natural search problems

Johnson, Papadimitriou, Yannakakis 1988: several classes defined by combinatorial properties.

FP – search problems solvable in polynomial time.

PPAD – *Polynomial Parity Argument Directed*

1. A directed graph on $\{0,1\}^n$ with indegrees and outdegrees ≤ 1 is given by: *two polynomial time computable functions, one for the predecessor and one for the successor of a vertex.*
2. $\bar{0}$ is a source.
3. Find a sink in the graph.

PPAD contains a number of important search problems:

- Brower's Fixedpoint Theorem
- Sperner Lemma
- Nash Equilibrium

An application

Theorem (Daskalakis, Goldberg, Papadimitriou, Chen, Deng, 2005)

Finding a Nash Equilibrium (with exponential precision) is a **PPAD** *complete search problem.*

Computing the minimax (i.e., the equilibrium in a zero-sum game) is an LP problem, hence can be done in *polynomial time.*

Can a Nash equilibrium in general games be found in polynomial time?

An application

Theorem (Daskalakis, Goldberg, Papadimitriou, Chen, Deng, 2005)

Finding a Nash Equilibrium (with exponential precision) is a **PPAD** *complete search problem.*

Computing the minimax (i.e., the equilibrium in a zero-sum game) is an LP problem, hence can be done in *polynomial time.*

Can a Nash equilibrium in general games be found in polynomial time?

We conjecture NO. We believe that **PPAD** are not solvable in polynomial time, because there exists an oracle A such that $\mathbf{PPAD}^A \not\subseteq \mathbf{FP}^A$ and Nash Equilibrium is complete in **PPAD**.

PLS – *Polynomial Local Search*

This class can be defined as the search problems reducible to *ITERATION*, which is defined by:

1. Given a polynomial time computable function f *defined on* $[0, N]$ *such that* $f(0) > 0$ *and* $f(x) \geq x$ *for all* $x \in [0, N]$*,*
2. find an $x \in [0, N]$ *such that* $x < f(x)$ *and* $f(x) = f(f(x))$*.*

PLS – *Polynomial Local Search*

This class can be defined as the search problems reducible to *ITERATION*, which is defined by:

1. Given a polynomial time computable function f *defined on* $[0, N]$ *such that* $f(0) > 0$ *and* $f(x) \geq x$ *for all* $x \in [0, N]$,
2. find an $x \in [0, N]$ *such that* $x < f(x)$ *and* $f(x) = f(f(x))$.

There is a natural exponential time algorithm to find a solution:

```
x:=0
do $x := f(x)$ while $f(x) \neq f(f(x))$
```

PPP – *Polynomial Pigeonhole Principle*

This class can be defined as the search problems reducible to *PHP*, *Pigeon Hole Principle*, which is defined by:

1. Given a polynomial time computable function $f : [0, N+1] \to [0, N]$,
2. find $x, y \in [0, N+1]$, $x \neq y$ *such that* $f(x) = f(y)$.

The only algorithm we know is to search all pairs (x, y).

Relativizations

If a search problem is defined by a polynomial time structure, then there is a natural way to relativize it to an oracle A —*replace 'polynomial time computable' by 'polynomial time computable using oracle A'.*

If the problem is given by a first order structure $(X; R_1, \dots, R_k, F_1, \dots, F_l)$ we can take just the relations R_i and the functions F_j as oracles.

Thus reductions are (type 2) *polynomial time functionals*. We shall call such reductions *generic*.

To show that such a search problem S^A is not solvable in polynomial time for some oracle A (ie., to show $S^A \notin \mathbf{FP}^A$) is usually easy—by diagonalization.

Theorem (Riis 1993)

Let a search problem S be defined by an existential sentence $\exists y \phi$ in the language $(R_1, \ldots, R_k, F_1, \ldots, F_l)$ where all R_i and F_j are generic relations and functions (thus we do not allow $\leq$ etc.).

If $\neg\exists y \phi$ has an infinite model, then $S^A \notin \mathbf{PLS}^A$,

for some oracle A.

To show that such a search problem S^A is not solvable in polynomial time for some oracle A (ie., to show $S^A \notin \mathbf{FP}^A$) is usually easy—by diagonalization.

Theorem (Riis 1993)

Let a search problem S be defined by an existential sentence $\exists y \phi$ in the language $(R_1, \ldots, R_k, F_1, \ldots, F_l)$ where all R_i and F_j are generic relations and functions (thus we do not allow $\leq$ etc.).

If $\neg\exists y \phi$ has an infinite model, then $S^A \notin \mathbf{PLS}^A$,

for some oracle A.

Example. $\mathbf{PPP}^A \not\subseteq \mathbf{PLS}^A$
— there exists a model in which PHP fails

Let P_i, $i = 1, 2$ be two search problems defined by existential formulas. Let $\{\alpha_n\}$, resp. $\{\beta_n\}$ be a sequence of propositions (tautologies) expressing the totality of problem P_1, resp. P_2.

Theorem (Buresh-Oppenheim, Morioka)

If there exists a generic reduction of P_1 to P_2, then tautologies $\{\alpha_n\}$ have polynomial size bounded depth propositional proofs from tautologies $\{\beta_n\}$.

Thus we can show relativized separations by proving lower bounds on the lengths of propositional proofs.

Provably total search problems in theories corresponding to complexity classes

Theorem (Buss 1986)

FP *is exactly the class of search problems provably total in* $\Theta_{\mathbf{P}}$.

Provably total search problems in theories corresponding to complexity classes

Theorem (Buss 1986)

FP *is exactly the class of search problems provably total in* $\Theta_{\mathbf{P}}$.

Theorem (Buss, Krajíček 1994)

PLS *is exactly the class of search problems provably total in* $\Theta_{\mathbf{NP}}$.

Provably total search problems in theories corresponding to complexity classes

Theorem (Buss 1986)

FP *is exactly the class of search problems provably total in* $\Theta_{\mathbf{P}}$.

Theorem (Buss, Krajíček 1994)

PLS *is exactly the class of search problems provably total in* $\Theta_{\mathbf{NP}}$.

Theorem (Krajíček, Skelley, Thapen 2006)

CPLS *is exactly the class of search problems provably total in* $\Theta_{\mathbf{\Sigma_2}}$.

More recently, Skelley and Thapen characterized search problems for all $\Theta_{\mathbf{\Sigma_k}}$.

A characterization of provably total search problems of Θ_{Σ_3}.

Let $c(p,x,y,z)$ (cost) and $g_1(p,x), g_2(p,y), g_3(p,z)$ (strategy) be polynomial time computable functions.
Given a, find u,v,w such that

$$c(a,u,g_2(a,u,v),w) \geq c(a,g_1(a,u),v,g_3(a,v,w)).$$

Interpretation

- $c(p,x,y,z)$ defines the payoff in a 3-step games parameterized by p;
- two players A and B play two copies of the game;
- B is given by the strategy $g_1(p,x), g_2(p,x,y), g_3(p,y,z)$;
- our goal is to find moves for A such that he gains at least as much in the first copy as B gets in the second.

$$\begin{tikzcd}
A \arrow[d, "g_1"] & B \arrow[r] & A \arrow[d, "g_3"] \\
B \arrow[r] & A \arrow[u, "g_2"] & B
\end{tikzcd}$$

$$\begin{tikzcd}
A \arrow[d, "g_1"] & B \arrow[r] & A \arrow[d, "g_3"] & : & c(u, g_2(u, v), w) \arrow[d, phantom, "\geq"] \\
B \arrow[r] & A \arrow[u, "g_2"] & B & : & c(g_1(u), v, g_3(v, w))
\end{tikzcd}$$

Let

$$C := \max_x \min_y \max_z c(x, y, z),$$

then

$$\exists x \forall y \exists z \; c(x, y, z) \geq C \quad \text{and} \quad \forall x \exists y \forall z \; C \leq c(x, y, z).$$

Therefore

$$\mathbb{N} \models \exists u \exists v \exists w \; c(u, g_2(u, v), w) \geq c(g_1(u), v, g_3(v, w)).$$

But note that finding C is Σ_3^p-hard.

Feasible incompleteness

The Feasible Incompleteness Thesis.
The phenomenon of incompleteness manifests itself at the level of polynomial time computations.

1. The feasible consistency problem.
2. Incompleteness caused by complexity.

The feasible consistency problem

Assume that all theories are axiomatized by a finite, or polynomial time computable set of axioms.

$Con_T(n) \equiv_{df}$ "there is no T-proof of contradiction of length n."

Given theories S and T,

1. do the sentences $Con_T(n)$ have S-proofs of length at most $p(n)$ for some polynomial p?
2. Is there a polynomial time algorithm for finding S-proofs of $Con_T(n)$?

Theorem

1. For every consistent and finitely axiomatized theory T, there exists $\varepsilon > 0$ such that, for every n, every T-proof of $Con_T(n)$ has size at least n^ε. [Friedman, 1979]
2. For every consistent and finitely axiomatized theory T, there exists C such that, for every n, there exists T-proof of $Con_T(n)$ of size $\leq Cn$. [Pudlák, 1984]

Conjecture (C1)

For every consistent finitely axiomatized theory S, there exists a consistent finitely axiomatized theory T such that the sentences $Con_T(n)$ do not have polynomially bounded S-proofs.

Call a propositional proof system P *length-optimal* if, for every propositional proof system Q, there exists a polynomial p such that for every tautology τ, if there is a proof of length n of τ in Q, then there is a proof of length at most $p(n)$ of τ in P.

Conjecture (C1')

There exists no length-optimal proof system.

C1≡C1'

Conjecture

Let S and T be finitely axiomatized consistent theories. If T proves the consistency of S, then the shortest S-proofs of $Con_T(n)$ have exponential lengths.

Incompleteness caused by complexity

Conjecture (C2)

For every finitely axiomatized consistent theory T, there exists a total polynomial search problem Q which is strictly stronger than all search problems provably total in T.

Conjecture (C2')

There exists no complete problem among total polynomial search problems, that is, no problem to which total polynomial search problems are reducible.

C2≡C2'

A connection between the two parts of the thesis

Conjecture (C3)

For every consistent finitely axiomatized theory S there exists a consistent finitely axiomatized theory T such that S-proofs of sentences $Con_T(n)$ cannot be produced by an algorithm in time $p(n)$, for p a polynomial.

Define that a *propositional proof system P is optimal* if it polynomially simulates every propositional proof system.

Conjecture (C3')

Optimal propositional proof systems do not exist.

Conjecture (C3")

For every finitely axiomatized consistent theory T, there exists a propositional proof system P such that T does not prove the soundness of any formalization of P.

C3≡C3'≡C3"