Almost transparent short proofs for $\mathrm{NP}_{\mathbb{R}}$

Klaus Meer

Brandenburgische Technische Universität, Cottbus, Germany

From Dynamics to Complexity:
A conference celebrating the work
of Mike Shub
Toronto, May 10, 2012

Supported by DFG under GZ:ME 1424/7-1

Outline

1. Introduction
2. Complexity and PCPs over $\mathbb{R}$
3. Results
4. Proofs

1. Introduction

One of most important results since 1990 in Theoretical Computer Science:

PCP Theorem by

Arora & Lund & Motwani & Sudan & Szegedy (1992, 1998)
Arora & Safra (1992, 1998)
Dinur (2005)

Probabilistically Checkable Proofs

- give other characterization of NP through verifiers, i.e., particular randomized algorithms
- allow to stabilize verification proofs for NP problems
- have tremendous applications to (non-) approximability results in combinatorial optimization.

So far not much work on PCP for algebraic computation models:

- PCPs over the reals: Blum-Shub-Smale BSS model
- Approximation of optimization problems over $\mathbb{R}$

So far not much work on PCP for algebraic computation models:

- PCPs over the reals: Blum-Shub-Smale BSS model
- Approximation of optimization problems over $\mathbb{R}$

Motivation: Does immense importance of PCP notion in the Turing model transfer to BSS model?

So far not much work on PCP for algebraic computation models:

- PCPs over the reals: Blum-Shub-Smale BSS model
- Approximation of optimization problems over $\mathbb{R}$

Motivation: Does immense importance of PCP notion in the Turing model transfer to BSS model?

First approach into that direction:
Transparent long proofs for $\mathrm{NP}_{\mathbb{R}}$ (M., 2005)

Now: Almost transparent short proofs for $\mathrm{NP}_{\mathbb{R}}$

2. Complexity and PCPs over $\mathbb{R}$

Computational model by Blum, Shub, and Smale over the reals:

Operations:	$+, -, *, :, x \geq 0?$
Size of a problem:	number of reals specifying input
Cost of an algorithm:	number of operations

2. Complexity and PCPs over $\mathbb{R}$

Computational model by Blum, Shub, and Smale over the reals:

Operations:	$+, -, *, :, x \geq 0?$
Size of a problem:	number of reals specifying input
Cost of an algorithm:	number of operations

Complexity classes for real number decision problems:

- $P_{\mathbb{R}}$: polynomially decidable
- $NP_{\mathbb{R}}$: polynomially verifiable (uncountable search space)
- $NP_{\mathbb{R}}$-complete problems: universal complexity within $NP_{\mathbb{R}}$

Main problem: Is $P_{\mathbb{R}} = NP_{\mathbb{R}}$?

Example

Quadratic Polynomial Systems QPS

Input: $n, m \in \mathbb{N}$, real polynomials in n variables $p_1, \ldots, p_m \in \mathbb{R}[x_1, \ldots, x_n]$ of degree at most 2; each p_i depending on at most 3 variables.

Question: Is there a common real solution $a \in \mathbb{R}^n$ such that

$$p_1(a) = 0 \ , \ \ldots \ , \ p_m(a) = 0 \ \ ?$$

QPS is $\mathrm{NP}_{\mathbb{R}}$-complete (Blum & Shub & Smale '89)

Example

Quadratic Polynomial Systems QPS

Input: $n, m \in \mathbb{N}$, real polynomials in n variables $p_1, \ldots, p_m \in \mathbb{R}[x_1, \ldots, x_n]$ of degree at most 2; each p_i depending on at most 3 variables.

Question: Is there a common real solution $a \in \mathbb{R}^n$ such that

$$p_1(a) = 0 \ , \ \ldots \ , \ p_m(a) = 0 \ \ ?$$

QPS is $\mathrm{NP}_\mathbb{R}$-complete (Blum & Shub & Smale '89)

Interesting for us is membership proof $\in \mathrm{NP}_\mathbb{R}$:
For verification guess solution $y \in \mathbb{R}^n$ and evaluate all $p_i(y)$
Clear: Result depends on all components y_i

PCP-question makes perfect sense:

Can we stabilize the proof and detect faults by inspecting considerably less many (real) components than n?

Formalization through real verifiers: probabilistic BSS machines using coin toss

Definition

Let $r, q : \mathbb{N} \mapsto \mathbb{N}$; a real verifier $V(r, q)$ is a polynomial time probabilistic BSS machine working as follows: V gets as input:

- a string $x \in \mathbb{R}^n$ (the problem instance);
- and a $y \in \mathbb{R}^s$ (the verification proof);

i) V produces non-adaptively $r(n)$ random bits (uniform distribution);

ii) from x and the $r(n)$ random bits V determines $q(n)$ many components of y;

iii) using x, the random bits and these $q(n)$ components of y V deterministically produces its result (accept or reject)

Acceptance condition for a language $L \subseteq \mathbb{R}^* := \bigcup_{n \geq 1} \mathbb{R}^n$:

A real verifier V accepts a language L iff

- for all $x \in L$ there exists a guess y such that

$$\Pr_{\rho \in \{0,1\}^{r(n)}} \{V(x, y, \rho) = \text{'accept'}\} = 1$$

- for all $x \notin L$ and for all y

$$\Pr_{\rho \in \{0,1\}^{r(n)}} \{V(x, y, \rho) = \text{'reject'}\} \geq \frac{1}{2}$$

Important: probability aspects still refer to discrete probabilities

Real verifiers as well produce random bits.

Definition

$\mathcal{R}, \mathcal{Q}$ function classes.
$L \in \mathrm{PCP}_{\mathbb{R}}(\mathcal{R}, \mathcal{Q})$ iff L is accepted by a real verifier $V(r, q)$ with $r \in \mathcal{R}, q \in \mathcal{Q}$

Definition

$\mathcal{R}, \mathcal{Q}$ function classes.
$L \in \mathrm{PCP}_{\mathbb{R}}(\mathcal{R}, \mathcal{Q})$ iff L is accepted by a real verifier $V(r, q)$ with $r \in \mathcal{R}, q \in \mathcal{Q}$

Example

$\mathrm{NP}_{\mathbb{R}} = \mathrm{PCP}_{\mathbb{R}}(0, poly)$

Definition

$\mathcal{R}, \mathcal{Q}$ function classes.
$L \in \mathrm{PCP}_{\mathbb{R}}(\mathcal{R}, \mathcal{Q})$ iff L is accepted by a real verifier $V(r, q)$ with $r \in \mathcal{R}, q \in \mathcal{Q}$

Example

$\mathrm{NP}_{\mathbb{R}} = \mathrm{PCP}_{\mathbb{R}}(0, poly)$
$\mathrm{NP}_{\mathbb{R}} \supseteq \mathrm{PCP}_{\mathbb{R}}(O(\log n), O(1))$

Definition

$\mathcal{R}, \mathcal{Q}$ function classes.
$L \in \mathrm{PCP}_{\mathbb{R}}(\mathcal{R}, \mathcal{Q})$ iff L is accepted by a real verifier $V(r, q)$ with $r \in \mathcal{R}, q \in \mathcal{Q}$

Example

$\mathrm{NP}_{\mathbb{R}} = \mathrm{PCP}_{\mathbb{R}}(0, poly)$
$\mathrm{NP}_{\mathbb{R}} \supseteq \mathrm{PCP}_{\mathbb{R}}(O(\log n), O(1))$

$\mathrm{PCP}_{\mathbb{R}}(O(\log n), 1)$: leads to questions about existence of zeros for (system of) univariate polynomials given by straight line program

Definition

$\mathcal{R}, \mathcal{Q}$ function classes.
$L \in \mathrm{PCP}_{\mathbb{R}}(\mathcal{R}, \mathcal{Q})$ iff L is accepted by a real verifier $V(r, q)$ with $r \in \mathcal{R}, q \in \mathcal{Q}$

Example

$\mathrm{NP}_{\mathbb{R}} = \mathrm{PCP}_{\mathbb{R}}(0, poly)$
$\mathrm{NP}_{\mathbb{R}} \supseteq \mathrm{PCP}_{\mathbb{R}}(O(\log n), O(1))$

$\mathrm{PCP}_{\mathbb{R}}(O(\log n), 1)$: leads to questions about existence of zeros for (system of) univariate polynomials given by straight line program

Classical PCP theorem: $\mathrm{NP} = \mathrm{PCP}(O(\log n), O(1))$.

Theorem (M., 2005)

$\mathrm{NP}_{\mathbb{R}} \subseteq PCP_{\mathbb{R}}(f(n), O(1))$, *where* f *is superlogarithmic.*

Transparent long proof: 'transparent' because only $O(1)$ bits need to be read; 'long' because proof has 2^f = superpolynomial number of real components.

Theorem (M., 2005)

$\mathrm{NP}_{\mathbb{R}} \subseteq PCP_{\mathbb{R}}(f(n), O(1))$, *where* f *is superlogarithmic.*

Transparent long proof: 'transparent' because only $O(1)$ bits need to be read; 'long' because proof has 2^f = superpolynomial number of real components.

Note: Discrete analogue of above result important for both existing proofs of PCP theorem in Turing model because of its structure; size of f is irrelevant there.

3. Results

Main result in this talk: First characterization of $\mathrm{NP}_{\mathbb{R}}$ via non-trivial real $\mathrm{PCP}_{\mathbb{R}}$-classes

Theorem (Main Theorem)

$\mathrm{NP}_{\mathbb{R}} = PCP_{\mathbb{R}}(O(\log(n)), polylog(n)))$

i.e., $\mathrm{NP}_{\mathbb{R}}$ has small almost transparent proofs.

3. Results

Main result in this talk: First characterization of $\mathrm{NP}_{\mathbb{R}}$ via non-trivial real $\mathrm{PCP}_{\mathbb{R}}$-classes

Theorem (Main Theorem)

$$\mathrm{NP}_{\mathbb{R}} = PCP_{\mathbb{R}}(O(\log(n)), polylog(n)))$$

i.e., $\mathrm{NP}_{\mathbb{R}}$ has small almost transparent proofs.

Immediate consequence:

Corollary

$$NEXP_{\mathbb{R}} = PCP_{\mathbb{R}}(poly(n), poly(n)))$$

4. Proof of Main Theorem

Goal: Construct required real verifier for QPS

4. Proof of Main Theorem

Goal: Construct required real verifier for QPS

in principle try to follow lines of classical proof by Arora et al. for 3-SAT;

some points below are standard, some try to pinpoint important differences in real number model

disregard tuning of parameters, only line of arguments given

Three main ingredients in design of verifier for checking solvability of polynomial system $\mathcal{P}$ over $\mathbb{R}^n$:

(1) coding of potential zero $a \in \mathbb{R}^n$ as table of function values of low-degree polynomial in k variables, degree d on suitable point set H^k : $f_a : H^k \rightarrow \mathbb{R}, k, d$ suitably chosen;
'low-degree' refers to choice of $d = O(\log n)$ later on

Three main ingredients in design of verifier for checking solvability of polynomial system $\mathcal{P}$ over $\mathbb{R}^n$:

(1) coding of potential zero $a \in \mathbb{R}^n$ as table of function values of low-degree polynomial in k variables, degree d on suitable point set H^k : $f_a : H^k \to \mathbb{R}, k, d$ suitably chosen;
'low-degree' refers to choice of $d = O(\log n)$ later on

(2) new problem introduced by (1): test whether table of function values for f_a indeed represents ld-polynomial

Three main ingredients in design of verifier for checking solvability of polynomial system $\mathcal{P}$ over $\mathbb{R}^n$:

(1) coding of potential zero $a \in \mathbb{R}^n$ as table of function values of low-degree polynomial in k variables, degree d on suitable point set H^k : $f_a : H^k \to \mathbb{R}, k, d$ suitably chosen;
'low-degree' refers to choice of $d = O(\log n)$ later on

(2) new problem introduced by (1): test whether table of function values for f_a indeed represents ld-polynomial

(3) verify whether a is a zero of $\mathcal{P}$ by evaluating finitely many canonically arising huge monomial sums over H^{3k}; requires to extend f_a to a larger domain F^k, where $H \subset F$.

Differences between Turing and BSS model:

minor: classical sum-check algorithm for dealing with (3) has to be adapted to real setting

Differences between Turing and BSS model:

minor: classical sum-check algorithm for dealing with (3) has to be adapted to real setting

major: classically, coding of satisfying assignment for 3-SAT formula results in ld-polynomial $f : F^k \rightarrow F$, where $F \supset H$ is finite field, i.e., algorithms are performed on highly structured domain;

Differences between Turing and BSS model:

minor: classical sum-check algorithm for dealing with (3) has to be adapted to real setting

major: classically, coding of satisfying assignment for 3-SAT formula results in ld-polynomial $f : F^k \rightarrow F$, where $F \supset H$ is finite field, i.e., algorithms are performed on highly structured domain;

now, range must be subset of $\mathbb{R}$ creating new problems:

Differences between Turing and BSS model:

minor: classical sum-check algorithm for dealing with (3) has to be adapted to real setting

major: classically, coding of satisfying assignment for 3-SAT formula results in ld-polynomial $f : F^k \to F$, where $F \supset H$ is finite field, i.e., algorithms are performed on highly structured domain;

now, range must be subset of $\mathbb{R}$ creating new problems:

- addition/multiplication not closed on used domain;
- probability arguments don't work since they rely on invariance of uniform distribution under some basic operations on finite fields;

Differences between Turing and BSS model:

minor: classical sum-check algorithm for dealing with (3) has to be adapted to real setting

major: classically, coding of satisfying assignment for 3-SAT formula results in ld-polynomial $f : F^k \to F$, where $F \supset H$ is finite field, i.e., algorithms are performed on highly structured domain;

now, range must be subset of $\mathbb{R}$ creating new problems:

- addition/multiplication not closed on used domain;
- probability arguments don't work since they rely on invariance of uniform distribution under some basic operations on finite fields;

low-degree test for (2) has to be extended to arbitrary finite (unstructured) domains

Let an QPS instance $\mathcal{P}$ over n real variables be given;

Let an QPS instance $\mathcal{P}$ over n real variables be given;

Idea: code (potential) zero $a \in \mathbb{R}^n$ of $\mathcal{P}$ as function $f_a : H^k \longrightarrow \mathbb{R}$, k such that $|H^k| \geq n$;

important: for $i \in \{1, \ldots, n\} \subseteq H^k$ let $f_a(i) = a_i$

in order to make following steps working f_a has to be extended to low-degree polynomial $f_a : F^k \rightarrow \mathbb{R}$ on larger domain F

Proof for verifier: function value table for f_a (plus additional informations ...)

Now show that with high probability

a) table with values for f_a represents a ld-polynomial

b) the corresponding $a \in \mathbb{R}^n$ is a zero of all polynomials in $\mathcal{P}$

Ad a): Low-degree test

Task: Given function-value table of $f : F^k \rightarrow \mathbb{R}$, does f represent with high probability a low-degree polynomial on F^k?

Ad a): Low-degree test

Task: Given function-value table of $f : F^k \longrightarrow \mathbb{R}$, does f represent with high probability a low-degree polynomial on F^k?

- many related tests in literature
- tests work mostly over finite fields, i.e., highly structured domains

Suitable for our situation: Test by Friedl & Hatsagi & Shen

arbitrary finite $F \subseteq \mathbb{R}, k, d \in \mathbb{N}$

Input: Function-value table for $f : F^k \to \mathbb{R}$

1. Fix arbitrary $c_1, \ldots, c_{d+1} \in F$;
2. pick uniformly at random $i \in \{1, \ldots, k\}$ and random elements $r_1, \ldots, r_k \in F$;
3. check whether f on $d+2$ many points $(r_1, \ldots, r_{i-1}, x, r_{i+1}, \ldots, r_k)$ for $x \in \{r_i, c_1, \ldots, c_{d+1}\}$ corresponds to univariate polynomial with respect to x of degree d

Accept if 'yes', else reject

Suitable for our situation: Test by Friedl & Hatsagi & Shen

arbitrary finite $F \subseteq \mathbb{R}, k, d \in \mathbb{N}$

Input: Function-value table for $f : F^k \to \mathbb{R}$

1. Fix arbitrary $c_1, \ldots, c_{d+1} \in F$;
2. pick uniformly at random $i \in \{1, \ldots, k\}$ and random elements $r_1, \ldots, r_k \in F$;
3. check whether f on $d + 2$ many points $(r_1, \ldots, r_{i-1}, x, r_{i+1}, \ldots, r_k)$ for $x \in \{r_i, c_1, \ldots, c_{d+1}\}$ corresponds to univariate polynomial with respect to x of degree d

Accept if 'yes', else reject

Remark: For coding reasons we choose $F \subset \mathbb{Z}$; sufficient because components of zero a should occur in the range of f

Theorem (Friedl et al, adapted)

Choose $\delta > 0$ sufficiently small.
If ld test in all of $O(\frac{k}{\delta})$ rounds accepts, then with high probability f is δ-close to a degree-d polynomial in k variables, i.e., disagrees only on a fraction of δ arguments of F^k with a unique such polynomial.

Theorem (Friedl et al, adapted)

Choose $\delta > 0$ sufficiently small.
If ld test in all of $O(\frac{k}{\delta})$ rounds accepts, then with high probability f is δ-close to a degree-d polynomial in k variables, i.e., disagrees only on a fraction of δ arguments of F^k with a unique such polynomial.

Ressources:

- $O(\frac{1}{\delta} \cdot k \cdot \log |F|)$ random bits
- $O(\frac{1}{\delta} \cdot k \cdot d)$ values of f inspected

Ad b): Suppose f_a is correct polynomial for $a \in \mathbb{R}^n$;
polynomials in instance $\mathcal{P}$ are of finitely many different types

one such type T :

$$x_i - x_j \cdot x_k = 0 \text{ for } i,j,k \in \{1,\ldots,n\}$$

χ_T characteristic function for type T, i.e.,

$$\chi_T : \{1,\ldots,n\} \to \{0,1\}$$
$$\chi_T(i,j,k) = 1 \;\Leftrightarrow\; x_i - x_j \cdot x_k \text{ occurs in } \mathcal{P}$$

Since $|H^k| > n$, χ_T is function on H^{3k}

If $a \in \mathbb{R}^n$ is zero and $\chi_T(i,j,k) = 1$, then

$$p_a^T(i,j,k) := f_a(i) - f_a(j) \cdot f_a(k) = a_i - a_j \cdot a_k = 0$$

If $a \in \mathbb{R}^n$ is zero and $\chi_T(i,j,k) = 1$, then

$$p_a^T(i,j,k) := f_a(i) - f_a(j) \cdot f_a(k) = a_i - a_j \cdot a_k = 0$$

Thus a is zero of all polynomials of type T iff

$$\sum_{(i,j,k) \in H^{3k}} \left(\chi_T(i,j,k) \cdot p_a^T(i,j,k) \right)^2 \; = \; 0$$

huge monomial sum

If $a \in \mathbb{R}^n$ is zero and $\chi_T(i,j,k) = 1$, then

$$p_a^T(i,j,k) := f_a(i) - f_a(j) \cdot f_a(k) = a_i - a_j \cdot a_k = 0$$

Thus a is zero of all polynomials of type T iff

$$\sum_{(i,j,k) \in H^{3k}} \left(\chi_T(i,j,k) \cdot p_a^T(i,j,k) \right)^2 = 0$$

huge monomial sum

Note: evaluation of p_a^T in single point requires inspection of three values of f_a; deterministic evaluation of entire sum requires $O(n^3)$ values from f_a , thus too many

Solution: Randomized Sum-Check (Lund & Fortnow & Karloff & Nisan)

- works with minor modifications as well in our setting
- probability estimates require to consider all involved functions on larger sets F^k and F^{3k}, respectively
- needs $O(\log n)$ random bits and $polylog(n)$ values of f_a to be inspected

Putting it all together:

Theorem

$$QPS \in PCP_{\mathbb{R}}(O(\log n), polylog(n))$$

and thus

$$\mathrm{NP}_{\mathbb{R}} = PCP_{\mathbb{R}}(O(\log n), polylog(n))$$

Putting it all together:

Theorem

$$QPS \in PCP_{\mathbb{R}}(O(\log n), polylog(n))$$

and thus

$$\mathrm{NP}_{\mathbb{R}} = PCP_{\mathbb{R}}(O(\log n), polylog(n))$$

Proof (outline). Given QPS instance $\mathcal{P}$

- verifier expects as proof table for ld-polynomial $f_a, a \in \mathbb{R}^n$ plus additional information necessary for sum-check.
- verifier performs low-degree test on f_a
- for all (finitely many) types of polynomials in $\mathcal{P}$ verifier performs sum-check algorithm

Verifier accepts if no test (repeated sufficiently many times) fails

- \# of random bits used: $O(k \cdot \log |F|)$
- \# of proof components read:

  $O(d \cdot k)$ in low-degree test
  $O(k \cdot |F|)$ in each sum-check procedure

Choosing $k = O(\frac{\log n}{\log \log n}), d = |H| = O(\log n), |F| = O(\log^4 n)$ results in:

- $O(\log n)$ random bits, thus proof size is polynomial
- $polylog(n)$ inspected components
- verification time is $polylog(n)$

Further questions

Main question: can we obtain full PCP theorem also for $\mathrm{NP}_{\mathbb{R}}$, i.e., can the number of inspected proof components be reduced from $polylog(n)$ to $O(1)$?

Further questions

Main question: can we obtain full PCP theorem also for $\mathrm{NP}_{\mathbb{R}}$, i.e., can the number of inspected proof components be reduced from $polylog(n)$ to $O(1)$?

Classical proof of PCP theorem requires at this point composition of verifiers and additional structure of low-degree test
Is both possible over the reals?

Further questions

Main question: can we obtain full PCP theorem also for $\mathrm{NP}_{\mathbb{R}}$, i.e., can the number of inspected proof components be reduced from $polylog(n)$ to $O(1)$?

Classical proof of PCP theorem requires at this point composition of verifiers and additional structure of low-degree test
Is both possible over the reals?

Start from alternative proof (Dinur): Ongoing work with M. Baartse

Further questions

Main question: can we obtain full PCP theorem also for $\mathrm{NP}_{\mathbb{R}}$, i.e., can the number of inspected proof components be reduced from $polylog(n)$ to $O(1)$?

Classical proof of PCP theorem requires at this point composition of verifiers and additional structure of low-degree test
Is both possible over the reals?

Start from alternative proof (Dinur): Ongoing work with M. Baartse

Approximation issues over the reals? Given a polynomial system over $\mathbb{R}$, can we efficiently approximate within a constant factor the maximal number of commonly solvable (over $\mathbb{R}$) equations?