Real Number Complexity Theory

Klaus Meer

Brandenburg University of Technology, Cottbus, Germany

Santaló's Summer School, Part 1, July 2012

(work supported by DFG, GZ:ME 1424/7-1)

Overview of talks

Talk 1: Introduction to real number complexity theory; structural results

Talk 2: Probabilistically Checkable Proofs; the 'classical' PCP theorem

Talk 3: PCPs in real number complexity

Outline today

1. Introduction

Since several years increasing interest in alternative (w.r.t. Turing machine) models of computation

Typical reasons:

- treatment of different problems
- more appropriate description of algorithmic phenomena which are hard or impossible to model by Turing machines

- focus on different aspects of a problem
- hope for new methods/results also for discrete problems
- ...

Important: Not a single model is the only correct one, but each is an idealization focussing on particular aspects

Classical model: Turing - machine: discrete problems, bit complexity

Alternative models:

real,algebraic models	computational geometry , analysis of algorithms in numerics, computer algebra etc.
recursive analysis	continuous real functions
neural nets	machine learning, optimization
analogue models	dynamical systems as algorithms
IBC	incomplete information: numerical integration
quantum computers	factorization
biology	membrane computing etc.

Example: Motion synthesis in robotics

Many practical problems within computational geometry result in question, whether a polynomial system is solvable

Here: Design of certain mechanisms in mechanical engineering

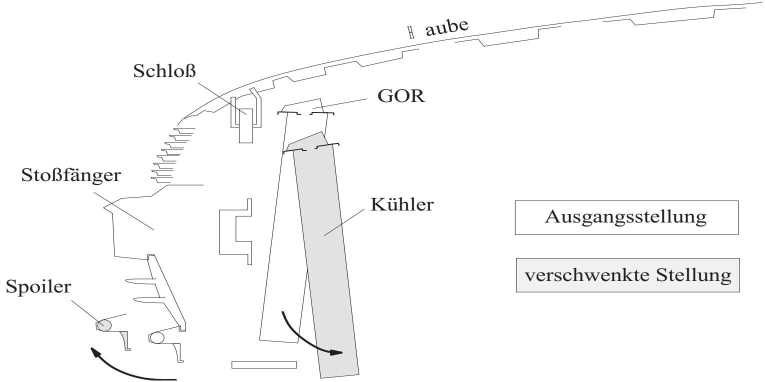
Example: Motion synthesis in robotics

Many practical problems within computational geometry result in question, whether a polynomial system is solvable

Here: Design of certain mechanisms in mechanical engineering

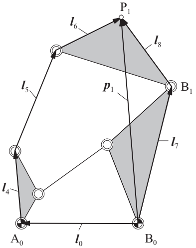
Task leads to interesting problems in different computational models: Turing model, real/complex BSS model, ...

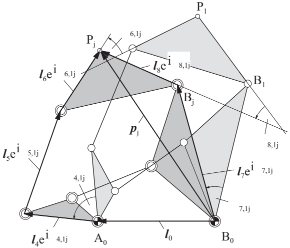
Example: Protection of pedestrians in traffic



Required motion of cooler and spoiler

Problem in kinematics: Design gearing mechanism satisfying certain demands





Stephenson gear

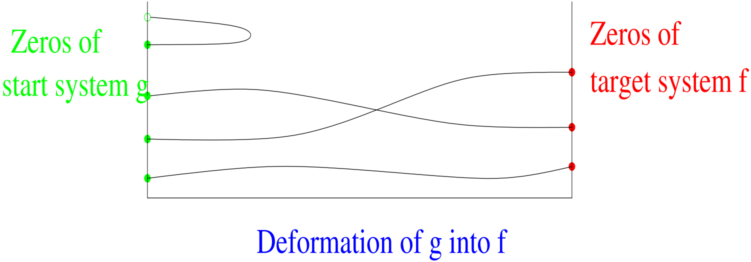
Example of a required motion:

Move point P through certain given positions

Typically leads to problem of solving a polynomial system with real or complex coefficients

Difficulty: Already few variables and low degrees can result in a system out of range of current methods!

Homotopy methods: Deform an easy to handle start system into the target system; then follow numerically the zeros of the start system into those of the target system



Here: First complete motion synthesis for so called

Stephenson mechanisms

M.& Schmitt & Schreiber, Mechanism and Machine Theory 2002

using PHC-package by Verschelde

Solvability question for polynomial system interesting from different viewpoints:

Applications	many problems lead to such systems, f.e., in robotics, non-linear optimization etc.
Mathematics	computational (semi-) algebraic geometry
Computer Science	fundamental importance in complexity theory and design of algorithms

Efficiency of homotopy methods relies on existence and number of zeros (paths)

$\rightsquigarrow$ Analysis needs more theory

Intermezzo: Carrying coals to Newcastle ...

Fundamental contributions on homotpy methods by:

Shub & Smale

Beltrán &Pardo

Bürgisser & Cucker

Dedieu, Li, Malajovich, Verschelde, . . .

Here: only one particular aspect related to questions picked up in later talks

2. A combinatorial optimization problem

Several (deep) mathematical methods for bounding number of zeros for polynomial systems $f : \mathbb{C}^n \mapsto \mathbb{C}^n$:

Bézout number	generalizes fundamental theorem of algebra, easy to compute, too a large bound
Mixed Volumes	Minkowski sum of Newton polytopes, hard to compute, (generically) correct bound
multi-homogeneous Bézout numbers	partitioning of variables, then Bézout for each group; mainly used in practice; complexity of computing it??

Example (Eigenpairs)

Find eigenpairs $(\lambda, v) \in \mathbb{C}^{n+1}$ of $M \in \mathbb{C}^{n\times n}$:

$$M \cdot v - \lambda \cdot v = 0\ ,\ \ v_n - 1 = 0$$

Has (generically) n solutions, but Bézout number 2^n.

Multi-homogeneous Bézout numbers: Group variables as

$$M \cdot v - \lambda \cdot v = 0\ ,\ \ v_n - 1 = 0$$

and homogenize w.r.t. both groups

$$\lambda_0 \cdot M \cdot v - v \cdot \lambda = 0\ ,\ \ v_n - v_0 = 0$$

Then the number of isolated roots in $(\mathbb{C})^n$ is bounded by the

2-homogeneous Bézout number, which here is n.

Consider $n \in \mathbb{N}$, a finite $A \subset \mathbb{N}^n$ and a polynomial system

$$\begin{array}{rcl} f_1(z) & = & \sum_{\alpha \in A} f_{1\alpha} z_1^{\alpha_1} z_2^{\alpha_2} \cdots z_n^{\alpha_n} \\ & \vdots & \\ f_n(z) & = & \sum_{\alpha \in A} f_{n\alpha} z_1^{\alpha_1} z_2^{\alpha_2} \cdots z_n^{\alpha_n} \ , \end{array}$$

where the $f_{i\alpha}$ are non-zero complex coefficients.

Thus, all f_i have the same support A

A multi-homogeneous structure: partition of $\{1, \ldots, n\}$ into k subsets

$$(I_1, \ldots, I_k) \ , \ I_j \subseteq \{1, \ldots, n\}$$

Define for each partition $(I_1, \ldots, I_k)$:

- block of variables related to I_j : $Z_j = \{z_i | i \in I_j\}$
- corresponding degree of f_i with respect to $\mathbb{Z}_j$:

  $d_j := \max\limits_{\alpha \in A} \sum_{l \in I_j} \alpha_l$

  (the same for all polynomials f_i because of same support)

Definition

a) The multi-hom. Bézout number w.r.t. partition $(I_1, \ldots, I_k)$ is the coefficient of $\prod_{j=1}^k \zeta_j^{|I_k|}$ in the formal polynomial $(d_1\zeta_1 + \cdots + d_k\zeta_k)^n$ (if each group not yet homogeneous)

$$\mathrm{Béz}(A, I_1, \ldots, I_k) = \binom{n}{|I_1| \; |I_2| \; \cdots \; |I_k|} \prod_{j=1}^{k} d_j^{|I_j|}$$

b) Minimal multi-hom. Bézout number:

$$\min_{\mathbf{I} \text{ partition}} \mathrm{Béz}(A, \mathbf{I})$$

Important: minimum is defined purely combinatorially

Theorem (Malajovich & M., 2005)

a) Given a polynomial system $f : \mathbb{C}^n \to \mathbb{C}^n$ there is no efficient Turing-algorithm that computes the minimal multi-homogeneous Bézout number (unless $P = NP$).

b) The same holds with respect to efficiently approximating the minimal such number within an arbitrary constant factor.

Theorem (Malajovich & M., 2005)

a) Given a polynomial system $f : \mathbb{C}^n \rightarrow \mathbb{C}^n$ there is no efficient Turing-algorithm that computes the minimal multi-homogeneous Bézout number (unless $P = NP$).

b) The same holds with respect to efficiently approximating the minimal such number within an arbitrary constant factor.

Proof.

Relate problem to 3-coloring problem for graphs: edges become binomials, triangles trinomials; mhBn shows a constant gap between 3-colorable and not 3-colorable graphs; part b) consequence of multiplicative structure of mhBn.

In practice: Balance whether additional effort for constructing start system pays out

alternatively: choose start system by random (Smale & Shub, Beltrán & Pardo, Bürgisser & Cucker)

Remark.

MHBN important in analysis of central path in interior point methods (Dedieu & Malajovich & Shub)

3. Complexity theory over $\mathbb{R}$: Blum & Shub & Smale model

Decision problem:	$L \subseteq \mathbb{R}^* := \bigcup_{n \geq 1} \mathbb{R}^n$
Operations:	$+, -, *, :, x \geq 0?$
Size of problem instance:	number of reals specifying input
Cost of an algorithm:	number of operations

Important: Algorithms are allowed to introduce finite set of parameters into its calculations:

Machine constants

Definition (Complexity class $\mathrm{P}_{\mathbb{R}}$)

$L \in \mathrm{P}_{\mathbb{R}}$ if efficiently decidable, i.e., number of steps in an algorithm deciding whether input $x \in \mathbb{R}^*$ belongs to L polynomially bounded in (algebraic) size of input x

Example

Solvability of linear system $A \cdot x = b$ by Gaussian elimination;

Existence of real solution of univariate polynomial $f \in \mathbb{R}[x]$

(Sturm)

Definition (Complexity class $\mathrm{NP}_{\mathbb{R}}$)

$L \in \mathrm{NP}_{\mathbb{R}}$ if efficiently verifiable, i.e., given $x \in \mathbb{R}^*$ and potential membership proof $y \in \mathbb{R}^*$, there is an algorithm verifying whether y proves $x \in L$.

Definition (Complexity class $\mathrm{NP}_{\mathbb{R}}$)

$L \in \mathrm{NP}_{\mathbb{R}}$ if efficiently verifiable, i.e., given $x \in \mathbb{R}^*$ and potential membership proof $y \in \mathbb{R}^*$, there is an algorithm verifying whether y proves $x \in L$.

If $x \in L$ there must exist such a proof; if $x \notin L$ no proof y is accepted.

Definition (Complexity class $\mathrm{NP}_{\mathbb{R}}$)

$L \in \mathrm{NP}_{\mathbb{R}}$ if efficiently verifiable, i.e., given $x \in \mathbb{R}^*$ and potential membership proof $y \in \mathbb{R}^*$, there is an algorithm verifying whether y proves $x \in L$.

If $x \in L$ there must exist such a proof; if $x \notin L$ no proof y is accepted.

The running time is polynomially bounded in (algebraic) size of input x (and thus, only polynomially bounded y's are relevant)

Example

1.) Quadratic Polynomial Systems QPS (Hilbert Nullstellensatz):

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;

Do the p_i's have a common real root?

Example

1.) Quadratic Polynomial Systems QPS (Hilbert Nullstellensatz):

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;

Do the p_i's have a common real root?

$\mathrm{NP}_{\mathbb{R}}$-verification for solvability of system

$$p_1(x) = 0 \ , \ \ldots, p_m(x) = 0$$

guesses solution $y^* \in \mathbb{R}^n$ and plugs it into all p_i's ; obviously all components of y^* have to be seen

Example (cntd.)

2. Mathematical Programming

Input: polynomial $f \in \mathbb{R}[x_1, \ldots, x_n]$ as objective function,

linear constraints $Ax \leq b$ where $A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m$

$$\text{Is } \; Min\{f(x) | x \in \mathbb{R}^n, Ax \leq b\} \leq 0 \quad ?$$

f linear/quadratic leads to Linear/Quadratic Programming

Example (cntd.)

2. Mathematical Programming

Input: polynomial $f \in \mathbb{R}[x_1, \ldots, x_n]$ as objective function,

linear constraints $Ax \leq b$ where $A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m$

$$\text{Is} \quad Min\{f(x) | x \in \mathbb{R}^n, Ax \leq b\} \leq 0 \quad ?$$

f linear/quadratic leads to Linear/Quadratic Programming

$\mathrm{NP}_{\mathbb{R}}$-verification: guess feasible point y in $\mathbb{R}^n$, check feasibility and evaluate objective function

Example (cntd.)

2. Mathematical Programming

Input: polynomial $f \in \mathbb{R}[x_1, \ldots, x_n]$ as objective function,

linear constraints $Ax \le b$ where $A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m$

$$\text{Is} \quad Min\{f(x) | x \in \mathbb{R}^n, Ax \le b\} \le 0 \quad ?$$

f linear/quadratic leads to Linear/Quadratic Programming

$\mathrm{NP}_{\mathbb{R}}$-verification: guess feasible point y in $\mathbb{R}^n$, check feasibility and evaluate objective function

Complexity in Turing model: LP $\in$ P ; QP is NP-complete

Complexity in BSS model: unknown

Conjectures: LP $\notin \mathrm{P}_{\mathbb{R}}$;

Example (cntd.)

2. Mathematical Programming

Input: polynomial $f \in \mathbb{R}[x_1, \ldots, x_n]$ as objective function,

linear constraints $Ax \leq b$ where $A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m$

$$\text{Is} \quad Min\{f(x) | x \in \mathbb{R}^n, Ax \leq b\} \leq 0 \quad ?$$

f linear/quadratic leads to Linear/Quadratic Programming

$\mathrm{NP}_{\mathbb{R}}$-verification: guess feasible point y in $\mathbb{R}^n$, check feasibility and evaluate objective function

Complexity in Turing model: LP $\in$ P ; QP is NP-complete

Complexity in BSS model: unknown

Conjectures: LP $\notin \mathrm{P}_{\mathbb{R}}$; QP not $\mathrm{NP}_{\mathbb{R}}$-complete (M. '94)

Definition ($\mathrm{NP}_{\mathbb{R}}$-completeness)

L is $\mathrm{NP}_{\mathbb{R}}$-complete if each problem A in $\mathrm{NP}_{\mathbb{R}}$ can be reduced in polynomial time to L, i.e., instead of deciding whether $x \in A$ one can decide whether $f(x) \in L$, where f can be computed in polynomial time in $size_{\mathbb{R}}(x)$.

Definition ($\mathrm{NP}_{\mathbb{R}}$-completeness)

L is $\mathrm{NP}_{\mathbb{R}}$-complete if each problem A in $\mathrm{NP}_{\mathbb{R}}$ can be reduced in polynomial time to L, i.e., instead of deciding whether $x \in A$ one can decide whether $f(x) \in L$, where f can be computed in polynomial time in $size_{\mathbb{R}}(x)$.

Complete problems have universal complexity within $\mathrm{NP}_{\mathbb{R}}$

Main open problem: Is $\mathrm{P}_{\mathbb{R}} = \mathrm{NP}_{\mathbb{R}}$?

Equivalent: Are there $\mathrm{NP}_{\mathbb{R}}$-complete problems in $\mathrm{P}_{\mathbb{R}}$?

Definition ($\mathrm{NP}_{\mathbb{R}}$-completeness)

L is $\mathrm{NP}_{\mathbb{R}}$-complete if each problem A in $\mathrm{NP}_{\mathbb{R}}$ can be reduced in polynomial time to L, i.e., instead of deciding whether $x \in A$ one can decide whether $f(x) \in L$, where f can be computed in polynomial time in $size_{\mathbb{R}}(x)$.

Complete problems have universal complexity within $\mathrm{NP}_{\mathbb{R}}$

Main open problem: Is $\mathrm{P}_{\mathbb{R}} = \mathrm{NP}_{\mathbb{R}}$?

Equivalent: Are there $\mathrm{NP}_{\mathbb{R}}$-complete problems in $\mathrm{P}_{\mathbb{R}}$?

Remark.

Similar definitions for structures like $\mathbb{C}$ (with $=?$ test), groups, vector spaces, ...

Theorem (Blum-Shub-Smale '89)

a) *The Hilbert-Nullstellensatz problem $QPS_{\mathbb{R}}$ is $NP_{\mathbb{R}}$-complete. Considered as problem $QPS_{\mathbb{C}}$ over $\mathbb{C}$ it is $NP_{\mathbb{C}}$-complete.*

b) *The real Halting problem $\mathbb{H}_{\mathbb{R}}$ is undecidable in the BSS model: Given a machine M (as codeword in $\mathbb{R}^*$) together with input $x \in \mathbb{R}^*$, does M halt on x?*

c) *Other undecidable problems: $\mathbb{Q}$ inside $\mathbb{R}$, the Mandelbrot set as subset of $\mathbb{R}^2$*

Theorem (Blum-Shub-Smale '89)

a) *The Hilbert-Nullstellensatz problem $QPS_{\mathbb{R}}$ is $NP_{\mathbb{R}}$-complete. Considered as problem $QPS_{\mathbb{C}}$ over $\mathbb{C}$ it is $NP_{\mathbb{C}}$-complete.*

b) *The real Halting problem $\mathbb{H}_{\mathbb{R}}$ is undecidable in the BSS model: Given a machine M (as codeword in $\mathbb{R}^*$) together with input $x \in \mathbb{R}^*$, does M halt on x?*

c) *Other undecidable problems: $\mathbb{Q}$ inside $\mathbb{R}$, the Mandelbrot set as subset of $\mathbb{R}^2$*

Both $\mathbb{H}_{\mathbb{R}}$ and $\mathbb{Q}$ are semi-decidable, i.e., there is a BSS algorithm that halts precisely on inputs from these sets.

Theorem

All problems in $\mathrm{NP}_{\mathbb{R}}$ are decidable in simple exponential time; similarly for $NP_{\mathbb{C}}$.

Theorem

All problems in $\mathrm{NP}_{\mathbb{R}}$ are decidable in simple exponential time; similarly for $NP_{\mathbb{C}}$.

Proof.

Difficulty: uncountable search space; requires quantifier elimination algorithms for real/algebraically closed fields

Long history starting with Tarski; fundamental contributions by Collins, Heintz et al., Grigoriev & Vorobjov, Renegar, Basu & Pollack & Roy, ...

Effective Hilbert Nullstellensatz: Giusti & Heintz, Pardo, ...

Some related questions treated below:

1. Structural complexity theory in different settings, transfer results for P=NP? question
2. Structure inside NP: Are there non-complete problems between P and NP?
3. Recursion theory: Undecidable problems, degrees of undecidability

(with focus on own research!)

4. P versus NP in different settings

Since P versus NP is major question in above (and further) models as well it is natural to ask, how these (and further) questions relate in different models, in particular:

how is classical Turing complexity theory related to results over $\mathbb{R}, \mathbb{C}, \ldots$?

Transfer Results

Theorem (Blum & Cucker & Shub& Smale 1996)

For all algebraically closed fields of characteristic 0 *the P versus NP question has the same answer.*

Theorem (Blum & Cucker & Shub& Smale 1996)

For all algebraically closed fields of characteristic 0 *the P versus NP question has the same answer.*

Proof.

Main idea is to eliminate complex machine constants in algorithms for problems that can be defined without such constants;

the $\mathrm{NP}_{\mathbb{C}}$-complete problem QPS has this property;

price to pay for elimination only polynomial slowdown

Technique: Some algebraic number theory

Elimination of machine constants important technique for several transfer results;

alternative proof by Koiran does it applying again Quantifier Elimination:

- algebraic constants are coded via minimal polynomials
- transcendental constants satisfy no algebraic equality test in algorithm, so each test is answered the same in a neighborhood of such a constant; using results from complex QE shows that there is a small rational point in such a neighborhood which can replace the transcendental constant

Relation between complex BSS model and randomized Turing algorithms through class BPP of discrete problems that can be decided with small two-sided error in polynomial time

Relation between complex BSS model and randomized Turing algorithms through class BPP of discrete problems that can be decided with small two-sided error in polynomial time

Theorem (Smale, Koiran)

Suppose $P_{\mathbb{C}} = NP_{\mathbb{C}}$*, then* $\mathrm{NP} \subseteq \mathrm{BPP}$.

Relation between complex BSS model and randomized Turing algorithms through class BPP of discrete problems that can be decided with small two-sided error in polynomial time

Theorem (Smale, Koiran)

Suppose $P_{\mathbb{C}} = NP_{\mathbb{C}}$, *then* $\mathrm{NP} \subseteq \mathrm{BPP}$.

Proof.

Extract from $\mathrm{P}_{\mathbb{C}}$ algorithm for $\mathrm{QPS}_{\mathbb{C}}$ a randomized algorithm for NP-complete variant of QPS;

Relation between complex BSS model and randomized Turing algorithms through class BPP of discrete problems that can be decided with small two-sided error in polynomial time

Theorem (Smale, Koiran)

Suppose $P_{\mathbb{C}} = NP_{\mathbb{C}}$, then $\mathrm{NP} \subseteq \mathrm{BPP}$.

Proof.

Extract from $\mathrm{P}_{\mathbb{C}}$ algorithm for $\mathrm{QPS}_{\mathbb{C}}$ a randomized algorithm for NP-complete variant of QPS; replacement of complex constants by randomly choosing small rational constants from a suitable set which with high probability contains rationals that behave the same as original constants.

Both above results not known for real algorithms; first deeper relation between real and Turing algorithms via additive real BSS machines, i.e., algorithms that only perform $+, -$ and tests $x \geq 0$;

Both above results not known for real algorithms; first deeper relation between real and Turing algorithms via additive real BSS machines, i.e., algorithms that only perform $+,-$ and tests $x \geq 0$; here: no non-rational machine constants

Both above results not known for real algorithms; first deeper relation between real and Turing algorithms via additive real BSS machines, i.e., algorithms that only perform $+, -$ and tests $x \geq 0$; here: no non-rational machine constants

Theorem (Fournier & Koiran 1998)

$$\mathrm{P} = \mathrm{NP} \textit{ (Turing)} \Leftrightarrow P_{\mathbb{R}}^{add} = NP_{\mathbb{R}}^{add} \textit{ (additive model)}$$

Both above results not known for real algorithms; first deeper relation between real and Turing algorithms via additive real BSS machines, i.e., algorithms that only perform $+, -$ and tests $x \geq 0$; here: no non-rational machine constants

Theorem (Fournier & Koiran 1998)

$$\mathrm{P} = \mathrm{NP} \text{ (Turing)} \Leftrightarrow P_{\mathbb{R}}^{add} = NP_{\mathbb{R}}^{add} \text{ (additive model)}$$

Proof.

Replacement of machine constants using deep result on point location in hyperplane arrangements by Meyer auf der Heide

Remark.

1. Similar results when allowing real machine constants, but introduces non-uniformity into Turing results.
2. In additive model with equality tests only, P and NP are provably different (M.'95)

5. Inside $\mathrm{NP}_{\mathbb{R}}$

Classical result in Turing complexity/recursion theory:

5. Inside $\mathrm{NP}_{\mathbb{R}}$

Classical result in Turing complexity/recursion theory:

Theorem (Ladner 1975)

If $\mathrm{P} \neq \mathrm{NP}$ *there are non-complete problems in* $\mathrm{NP} \setminus \mathrm{P}$.

5. Inside $\mathrm{NP}_{\mathbb{R}}$

Classical result in Turing complexity/recursion theory:

Theorem (Ladner 1975)

If $\mathrm{P} \neq \mathrm{NP}$ there are non-complete problems in $\mathrm{NP} \setminus \mathrm{P}$.

Proof.

Key point is diagonalization against family $\{P_1, P_2, \ldots\}$ of

P-machines and family $\{R_1, R_2, \ldots\}$ of poly-time reductions;

both algorithm-classes are countable in Turing model;

Proof (cntd.)

given NP-complete L construct $\tilde{L} \in \mathrm{NP}$ s.t. one after the other P_i fails to decide $\tilde{L}$ and R_i fails to reduce L to $\tilde{L}$;

Proof (cntd.)

given NP-complete L construct $\tilde{L} \in \mathrm{NP}$ s.t. one after the other

P_i fails to decide $\tilde{L}$ and R_i fails to reduce L to $\tilde{L}$;

$\tilde{L}$ constructed dimensionwise: find effectively error dimensions for each P_i, R_i;

rest a folklore padding argument to force $\tilde{L}$ into NP

Proof (cntd.)

given NP-complete L construct $\tilde{L} \in \mathrm{NP}$ s.t. one after the other

P_i fails to decide $\tilde{L}$ and R_i fails to reduce L to $\tilde{L}$;

$\tilde{L}$ constructed dimensionwise: find effectively error dimensions for each P_i, R_i;

rest a folklore padding argument to force $\tilde{L}$ into NP

Computational models over $\mathbb{R}, \mathbb{C}$: set of algorithms uncountable

thus, direct transformation of above construction fails

Theorem (Malajovich & M. 1995)

If $P_{\mathbb{C}} \neq NP_{\mathbb{C}}$ there are non-complete problems in $NP_{\mathbb{C}} \setminus P_{\mathbb{C}}$.

Theorem (Malajovich & M. 1995)

If $P_{\mathbb{C}} \neq NP_{\mathbb{C}}$ there are non-complete problems in $NP_{\mathbb{C}} \setminus P_{\mathbb{C}}$.

Proof.

Efficient elimination of complex machine constants allows to reduce problem to the algebraic closure $\bar{\mathbb{Q}}$ of $\mathbb{Q}$ in $\mathbb{C}$, i.e., to a countable setting; then adapt Ladner's proof

'Elimination-of-constants' result is not known for $\mathbb{R}$, so what can be done?

'Elimination-of-constants' result is not known for $\mathbb{R}$, so what can be done?

Central complexity class for investigations: $\mathrm{P}_{\mathbb{R}}/\mathrm{const}$ (Michaux)

$\mathrm{P}_{\mathbb{R}}/\mathrm{const}$ allows diagonalization technique in uncountable settings

idea: consider discrete skeleton of real/complex algorithms,

split real/complex constants from skeleton

'Elimination-of-constants' result is not known for $\mathbb{R}$, so what can be done?

Central complexity class for investigations: $\mathrm{P}_{\mathbb{R}}/\mathrm{const}$ (Michaux)

$\mathrm{P}_{\mathbb{R}}/\mathrm{const}$ allows diagonalization technique in uncountable settings

idea: consider discrete skeleton of real/complex algorithms, split real/complex constants from skeleton

$\rightsquigarrow$ basic machine:

'Elimination-of-constants' result is not known for $\mathbb{R}$, so what can be done?

Central complexity class for investigations: $\mathrm{P}_{\mathbb{R}}/\mathrm{const}$ (Michaux)

$\mathrm{P}_{\mathbb{R}}/\mathrm{const}$ allows diagonalization technique in uncountable settings

idea: consider discrete skeleton of real/complex algorithms,

split real/complex constants from skeleton

$\rightsquigarrow$ basic machine: $\underbrace{M}_{\text{skeleton}}$

'Elimination-of-constants' result is not known for $\mathbb{R}$, so what can be done?

Central complexity class for investigations: $\mathrm{P}_{\mathbb{R}}/\mathrm{const}$ (Michaux)

$\mathrm{P}_{\mathbb{R}}/\mathrm{const}$ allows diagonalization technique in uncountable settings

idea: consider discrete skeleton of real/complex algorithms,

split real/complex constants from skeleton

$\rightsquigarrow$ basic machine: $\underbrace{M}_{\text{skeleton}} (\underbrace{x}_{\textit{input}},$

'Elimination-of-constants' result is not known for $\mathbb{R}$, so what can be done?

Central complexity class for investigations: $\mathrm{P}_{\mathbb{R}}/\mathrm{const}$ (Michaux)

$\mathrm{P}_{\mathbb{R}}/\mathrm{const}$ allows diagonalization technique in uncountable settings

idea: consider discrete skeleton of real/complex algorithms, split real/complex constants from skeleton

$$\rightsquigarrow \text{basic machine: } \underbrace{M}_{\text{skeleton}}(\underbrace{x}_{\textit{input}},\underbrace{c}_{\substack{\textit{machine}\\ \textit{constants}}})$$

$L \in \mathrm{P}_{\mathbb{R}}/\mathrm{const}$ $\Leftrightarrow$ there is a skeleton M using k constants such that

$L \in \mathrm{P}_{\mathbb{R}}/\mathrm{const}$ $\Leftrightarrow$ there is a skeleton M using k constants such that for each input dimension n there is a choice $c^{(n)} \in \mathbb{R}^k$ such that $M(\bullet, c^{(n)})$ decides L upto dimension n in polynomial time.

$L \in \mathrm{P}_{\mathbb{R}}/\mathrm{const}$ $\Leftrightarrow$ there is a skeleton M using k constants such that for each input dimension n there is a choice $c^{(n)} \in \mathbb{R}^k$ such that $M(\bullet, c^{(n)})$ decides L upto dimension n in polynomial time.

Important:

skeleton is used uniformly, machine constants non-uniformly,

$L \in \mathrm{P}_{\mathbb{R}}/\mathrm{const} \Leftrightarrow$ there is a skeleton M using k constants such that for each input dimension n there is a choice $c^{(n)} \in \mathbb{R}^k$ such that $M(\bullet, c^{(n)})$ decides L upto dimension n in polynomial time.

Important:

skeleton is used uniformly, machine constants non-uniformly,

$\mathrm{P}_{\mathbb{R}}/\mathrm{const}$ is a restricted version of non-uniform class $\mathrm{P}_{\mathbb{R}}/poly$;

set of basic machines countable!

Similar for other models: $\mathrm{P}_{\mathbb{C}}/\mathrm{const}, \mathrm{P}_{\mathbb{R}}^{\mathrm{add}}/\mathrm{const}, \mathrm{P}_{\mathbb{R}}^{\mathrm{rc}}/\mathrm{const}$

Theorem (Ben-David & M. & Michaux 2000)

If $\mathrm{NP}_{\mathbb{R}} \not\subseteq \mathrm{P}_{\mathbb{R}}/\mathrm{const}$ *there exist problems in* $\mathrm{NP}_{\mathbb{R}} \setminus \mathrm{P}_{\mathbb{R}}/\mathrm{const}$ *which are not* $\mathrm{NP}_{\mathbb{R}}$*-complete under* $\mathrm{P}_{\mathbb{R}}/\mathrm{const}$ *reductions.*

Theorem (Ben-David & M. & Michaux 2000)

If $\mathrm{NP}_{\mathbb{R}} \not\subseteq \mathrm{P}_{\mathbb{R}}/\mathrm{const}$ *there exist problems in* $\mathrm{NP}_{\mathbb{R}} \setminus \mathrm{P}_{\mathbb{R}}/\mathrm{const}$ *which are not* $\mathrm{NP}_{\mathbb{R}}$*-complete under* $\mathrm{P}_{\mathbb{R}}/\mathrm{const}$ *reductions.*

Proof.

Construct again diagonal problem $\tilde{L}$ along Ladner's line;

fool step by step all basic decision / reduction machines;

fooling dimensions computed via quantifier elimination: for each n and basic machine M it is first order expressible whether M with some choice of constants decides problem upto dimension n.

Thus central: analysis of $\mathrm{P}/\mathrm{const}$ in different models;

here notions from model theory enter

Theorem (Michaux; Ben-David & Michaux & M.)

For every ω-saturated structure it is $\mathrm{P} = \mathrm{P}/\mathrm{const}$.

Thus central: analysis of $\mathrm{P}/\mathrm{const}$ in different models;

here notions from model theory enter

Theorem (Michaux; Ben-David & Michaux & M.)

For every ω-saturated structure it is $\mathrm{P} = \mathrm{P}/\mathrm{const}.$

ω-saturation roughly means: given countable family $\phi_n(c)$ of first-order formulas such that each finite subset is commonly satisfiable, then the entire family is satisfiable.

$\mathbb{R}$ is not ω-saturated: $\phi_n(c) \equiv c \geq n$

What is known about $\mathrm{P}/\mathrm{const}$ in different models:

What is known about $\mathrm{P/const}$ in different models:

Turing: $\mathrm{P} = \mathrm{P/const}$ thus Ladner holds

What is known about $\mathrm{P}/\mathrm{const}$ in different models:

Turing: $\mathrm{P} = \mathrm{P}/\mathrm{const}$ thus Ladner holds

BSS over $\mathbb{C}$: $\mathrm{P}_{\mathbb{C}} = \mathrm{P}_{\mathbb{C}}/\mathrm{const}$ thus Ladner holds

What is known about $\mathrm{P}/\mathrm{const}$ in different models:

Turing: $\mathrm{P} = \mathrm{P}/\mathrm{const}$ thus Ladner holds

BSS over $\mathbb{C}$: $\mathrm{P}_{\mathbb{C}} = \mathrm{P}_{\mathbb{C}}/\mathrm{const}$ thus Ladner holds

BSS over $\mathbb{R}$: highly unlikely that $\mathrm{P}_{\mathbb{R}} = \mathrm{P}_{\mathbb{R}}/\mathrm{const}$

Chapuis & Koiran

What is known about $\mathrm{P}/\mathrm{const}$ in different models:

Turing: $\mathrm{P} = \mathrm{P}/\mathrm{const}$ thus Ladner holds

BSS over $\mathbb{C}$: $P_{\mathbb{C}} = P_{\mathbb{C}}/\mathrm{const}$ thus Ladner holds

BSS over $\mathbb{R}$: highly unlikely that $\mathrm{P}_{\mathbb{R}} = \mathrm{P}_{\mathbb{R}}/\mathrm{const}$

Chapuis & Koiran

additive BSS over $\mathbb{R}$: $P_{\mathbb{R}}^{add} = P_{\mathbb{R}}^{add}/\mathrm{const}$ thus Ladner holds

Chapuis & Koiran

What is known about $\mathrm{P}/\mathrm{const}$ in different models:

Turing: $\mathrm{P} = \mathrm{P}/\mathrm{const}$ thus Ladner holds

BSS over $\mathbb{C}$: $\mathrm{P}_{\mathbb{C}} = \mathrm{P}_{\mathbb{C}}/\mathrm{const}$ thus Ladner holds

BSS over $\mathbb{R}$: highly unlikely that $\mathrm{P}_{\mathbb{R}} = \mathrm{P}_{\mathbb{R}}/\mathrm{const}$

Chapuis & Koiran

additive BSS over $\mathbb{R}$: $\mathrm{P}_{\mathbb{R}}^{add} = \mathrm{P}_{\mathbb{R}}^{add}/\mathrm{const}$ thus Ladner holds

Chapuis & Koiran

real BSS with restricted use of constants Ladner holds

M. 2012

Restricted BSS model:

restricted use of machine constants:

Restricted BSS model:

restricted use of machine constants:

input variables can be used arbitrarily; all intermediate results depend linearly on machine constants (thus no multiplication between machine constants)

$$\rightsquigarrow \quad \text{classes} \;\; \mathrm{P}_{\mathbb{R}}^{\mathrm{rc}}, \quad \mathrm{NP}_{\mathbb{R}}^{\mathrm{rc}}, \quad \mathrm{P}_{\mathbb{R}}^{\mathrm{rc}}/\mathrm{const}$$

Restricted BSS model:

restricted use of machine constants:

input variables can be used arbitrarily; all intermediate results depend linearly on machine constants (thus no multiplication between machine constants)

$$\rightsquigarrow \quad \text{classes} \quad \mathrm{P}^{\mathrm{rc}}_{\mathbb{R}}, \quad \mathrm{NP}^{\mathrm{rc}}_{\mathbb{R}}, \quad \mathrm{P}^{\mathrm{rc}}_{\mathbb{R}}/\mathrm{const}$$

Theorem

QPS is $\mathrm{NP}^{\mathrm{rc}}_{\mathbb{R}}$-complete (under $\mathrm{P}^{\mathrm{rc}}_{\mathbb{R}}$-reductions)

thus: restricted model closer to full BSS model than linear/additive models $\rightsquigarrow$ motivation for studying it!

Theorem (M. 2012)

Ladner's theorem holds in the real BSS model with restricted use of constants.

Proof.

Main step is to prove equality $\mathrm{P}_{\mathbb{R}}^{\mathrm{rc}} = \mathrm{P}_{\mathbb{R}}^{\mathrm{rc}}/\mathrm{const}$; proof relies on a limit argument in affine geometry that allows elimination of non-uniform machine constants by uniform ones

Theorem (M. 2012)

Ladner's theorem holds in the real BSS model with restricted use of constants.

Proof.

Main step is to prove equality $\mathrm{P}_{\mathbb{R}}^{\mathrm{rc}} = \mathrm{P}_{\mathbb{R}}^{\mathrm{rc}}/\mathrm{const}$; proof relies on a limit argument in affine geometry that allows elimination of non-uniform machine constants by uniform ones

Problem: Can ideas be somehow used to prove Ladner in full real BSS model?

6. Recursion theory over $\mathbb{R}$

Blum-Shub-Smale: Real Halting problem is BSS undecidable

$$\mathbb{H}_{\mathbb{R}} := \{\text{code of BSS machine } M \text{ that halts on empty input}\}$$

6. Recursion theory over $\mathbb{R}$

Blum-Shub-Smale: Real Halting problem is BSS undecidable

$$\mathbb{H}_{\mathbb{R}} := \{\text{code of BSS machine } M \text{ that halts on empty input}\}$$

further undecidable problems:

- $\mathbb{Q}$, i.e., given $x \in \mathbb{R}$, is x rational? Problem is semi-decidable: there is an algorithms which stops exactly for inputs from $\mathbb{Q}$;
- graphs of sin and exp functions
- Mandelbrot and certain Julia sets
- ...

Typical related questions:

- degrees of undecidability
- Post's problem: are there problems easier than $\mathbb{H}_{\mathbb{R}}$ yet undecidable?
- find other natural undecidable problems equivalent to $\mathbb{H}_{\mathbb{R}}$

Typical related questions:

- degrees of undecidability
- Post's problem: are there problems easier than $\mathbb{H}_{\mathbb{R}}$ yet undecidable?
- find other natural undecidable problems equivalent to $\mathbb{H}_{\mathbb{R}}$

Formalization of comparing problems via oracle machines:

A is Turing reducible to B iff A can be decided by a BSS machine that additionally has access to an oracle for membership in B.

A equivalent to B iff both are Turing reducible to each other

Real Post's problem: Are there problems Turing reducible to $\mathbb{H}_{\mathbb{R}}$ that are not Turing reducible from $\mathbb{H}_{\mathbb{R}}$ but yet undecidable?

Real Post's problem: Are there problems Turing reducible to $\mathbb{H}_{\mathbb{R}}$ that are not Turing reducible from $\mathbb{H}_{\mathbb{R}}$ but yet undecidable?

Turing setting: question posted in 1944 and solved 57/58 by Friedberg & Muchnik;

however: no explicit problem with this property known so far

Theorem (M. & Ziegler 2007)

The rational numbers $\mathbb{Q}$ are strictly easier than $\mathbb{H}_{\mathbb{R}}$ yet undecidable.

Proof.

Show that set T of transcendent reals is not semi-decidable even with oracle for $\mathbb{Q}$; topological and number theoretic arguments how rational functions map dense subsets of algebraic numbers.

Note: Algebraic numbers $= \mathbb{R} \setminus T$ are semi-decidable.

Word Problem for Groups I

Consider product bab^2ab^2aba in free semi-group $\langle\{a, b\}\rangle$;

subject to

- rule $ab = 1$ it can be simplified to b^2 but not to 1
- using additional rules $a^4 = a^2$ it can be simplified to 1

Word Problem for Groups I

Consider product bab^2ab^2aba in free semi-group $\langle\{a, b\}\rangle$;

subject to

- rule $ab = 1$ it can be simplified to b^2 but not to 1
- using additional rules $a^4 = a^2$ it can be simplified to 1

Fix set X and set R of equations over $\langle X\rangle = (X \cup X^{-1})^*$.

Word problem for $\langle X\rangle$: Given a formal product

$w := x_1^{\pm 1} x_2^{\pm 1 \dots} x_n^{\pm 1}, x_i \in X$, does it hold subject to R that $w = 1$?

Boone '58, Novikov '59: There exist finite X, R such that the related word problem is equivalent to discrete Halting problem.

Word Problem for Groups II

Now set $X \subset \mathbb{R}^*$ of real generators, R rules on $\langle X \rangle$;

word problem as before, but suitable for BSS setting

Example

$X := \{x_r | r \in \mathbb{R}\}; R := \{x_{nr} = x_r, x_{r+k} = x_r | r \in \mathbb{R}, n \in \mathbb{N}, k \in \mathbb{Z}\}$

X, R are BSS decidable and $x_r = 1 \Leftrightarrow$

Word Problem for Groups II

Now set $X \subset \mathbb{R}^*$ of real generators, R rules on $\langle X \rangle$;

word problem as before, but suitable for BSS setting

Example

$X := \{x_r | r \in \mathbb{R}\}; R := \{x_{nr} = x_r, x_{r+k} = x_r | r \in \mathbb{R}, n \in \mathbb{N}, k \in \mathbb{Z}\}$

X, R are BSS decidable and $x_r = 1 \Leftrightarrow r \in \mathbb{Q}$

Word Problem for Groups II

Now set $X \subset \mathbb{R}^*$ of real generators, R rules on $\langle X \rangle$;

word problem as before, but suitable for BSS setting

Example

$X := \{x_r | r \in \mathbb{R}\}; R := \{x_{nr} = x_r, x_{r+k} = x_r | r \in \mathbb{R}, n \in \mathbb{N}, k \in \mathbb{Z}\}$

X, R are BSS decidable and $x_r = 1 \Leftrightarrow r \in \mathbb{Q}$

Thus this world problem is undecidable, but easier than $\mathbb{H}_{\mathbb{R}}$.

Theorem (M. & Ziegler 2009)

There are BSS decidable sets $X \subset \mathbb{R}^N, R \subset \mathbb{R}^*$ such that the resulting word problem is equivalent to $\mathbb{H}_{\mathbb{R}}$.

Proof.

Lot of combinatorial group theory: Nielsen reduction, HNN extensions, Britton's Lemma, amalgamation, ...

Theorem (M. & Ziegler 2009)

There are BSS decidable sets $X \subset \mathbb{R}^N, R \subset \mathbb{R}^*$ *such that the resulting word problem is equivalent to* $\mathbb{H}_{\mathbb{R}}$.

Proof.

Lot of combinatorial group theory: Nielsen reduction, HNN extensions, Britton's Lemma, amalgamation, ...

Reals enter as index set for set of generators; no particular influence of semi-algebraic geometry; word problem is located in computational group theory and thus presents new kind of complete problem in BSS recursion theory.

Further research questions:

- power of other undecidable problems like Mandelbrot set?
- use of machine constants: what power does one gain by using more machine constants?
- find word problems representing real number complexity classes like $\mathrm{NP}_{\mathbb{R}}$ or $\mathrm{P}_{\mathbb{R}}$
- Bounded query computations: how many queries to an oracle B are needed to compute characteristic function χ_n^A for A^n on $(\mathbb{R}^*)^n$?

  Example: For $A = B = \mathbb{H}_{\mathbb{R}}$ $\log n$ queries are sufficient.