Intrinsic complexity

Yiannis N. Moschovakis
UCLA and University of Athens

Panhellenic Logic Symposium 11, July 12 – 16, 2017, Delphi

The Euclidean algorithm

- For $a, b \in \mathbb{N} = \{0, 1, \ldots\},\ a \geq b \geq 1,$

$$(\varepsilon) \quad \boxed{\mathrm{gcd}(a,b) = \text{if } (\mathrm{rem}(a,b) = 0) \text{ then } b \text{ else } \mathrm{gcd}(b, \mathrm{rem}(a,b))}$$

where $a = \mathrm{iq}(a,b)b + \mathrm{rem}(a,b) \quad (0 \leq \mathrm{rem}(a,b) < b)$

$$\begin{aligned} \mathrm{calls}_{\{\mathrm{rem}\}}(\varepsilon, a, b) &= \text{the number of calls to rem } \varepsilon \text{ makes to compute } \mathrm{gcd}(a,b) \\ &\leq 2\log(b) \qquad (a \geq b \geq 2) \end{aligned}$$

- $\boxed{\text{Is } \varepsilon \text{ optimal for computing } \mathrm{gcd}(a,b) \text{ from } \{\mathrm{rem}, =_0\}?}$

Basic Conjecture I *For every algorithm* α *which computes* $\mathrm{gcd}(a,b)$ *from* $\{\mathrm{rem}, =_0\}$

$$(\exists r > 0)\Big[(\text{for infinitely many pairs } a \geq b)[\mathrm{calls}_{\{\mathrm{rem}\}}(\alpha, a, b) \geq r\log(a)]\Big]$$

- *Can we derive lower complexity bounds for natural mathematical problems which restrict all algorithms?*

The Value Complexity I

• A classical method for establishing intrinsic complexity lower bounds which assumes practically nothing about "what algorithms are"

Horner's rule: For any field F and $n \geq 1$, the value of a polynomial of degree n can be computed using no more than n multiplications and n additions in F:

$$a_0 + a_1x + a_2x^2 + \cdots + a_nx^n = a_0 + x\Big(a_1 + a_2x + \cdots + a_nx^{n-1}\Big)$$

Theorem (Pan 1966, (Winograd 1967, 1970))

Every algorithm from the complex field operations requires at least n multiplications/divisions and at least n additions/subtractions to compute $a_0 + a_1x + a_2x^2 + \cdots + a_nx^n$ when $\vec{a}, x$ are algebraically independent complex numbers (the generic case)

. . . because it takes that many applications of the field operations to construct the value $a_0 + a_1x + a_2x^2 + \cdots + a_nx^n$ from $a_0, \ldots, a_n, x$

The Value Complexity II

Theorem (van den Dries)

If an algorithm α *computes* $\mathrm{gcd}(x, y)$ *from* $0, 1, +, -, \mathrm{iq}, \mathrm{rem}, \cdot, <$ *and*

$$\mathrm{calls}(\alpha, x, y) = \textit{ the number of calls to the primitives } \alpha \textit{ makes to compute } \mathrm{gcd}(x, y),$$

then for all sufficiently large $a > b$ *such that* $a^2 = 1 + 2b^2$ (Pell pairs),

$$\mathrm{calls}(\alpha, a+1, b) \geq \frac{1}{4}\sqrt{\log \log b}$$

... because it takes at least that many applications of the primitives to construct the value $\mathrm{gcd}(a+1, b)$ from $a+1$ and b when (a, b) is a large Pell pair

- This is the best lower bound known for the gcd function from the primitives of $(\mathbb{N}, 0, 1, +, \cdot)$ expanded with arithmetic division

General aim

• *The method of value complexity cannot yield lower bounds for decision problems*, because their output (tt or ff) is available with no computation

• We will develop a general method for deriving widely applicable lower complexity bounds for algorithms which decide relations from specified primitives. e.g.,

• The nullity (0-testing) relation on a field F:

$$N_F(a_0, \ldots, a_n, x) \iff a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n = 0$$

from some or all of the field primitives and $=$

• The coprimeness relation

$$x \perp\!\!\!\perp y \iff x, y > 1 \ \& \ \gcd(x, y) = 1$$

from various primitives on $\mathbb{N} = \{0, 1, \ldots\}$

• I will list some of the applications at the end, but my main aim in this lecture is to make the homomorphism method precise and to justify it

Sample result: the intrinsic calls-complexity

Proposition *For every structure* $\mathbf{A} = (A, c_1, \ldots, R_1, \ldots, f_1, \ldots)$ *and every n-ary relation* $R(\vec{x})$ *on* A*, there is a a function*

$$\mathrm{calls}_R : A^n \to \mathbb{N} \cup \{\infty\}$$

such that, if α *is any (deterministic or non-deterministic) algorithm which decides* R *from the primitives of* $\mathbf{A}$*, then for every* $\vec{x}$*,*

$(\star)$ $\mathrm{calls}_R(\vec{x}) \leq$ the number of distinct calls to the primitives that α needs to make to decide $R(\vec{x})$

- We will define calls_R from R and $\mathbf{A}$ with no reference to "algorithms";
- $(\star)$ is a theorem for algorithms specified by *computation models* — recursive programs, RAMs, Turing machines with oracles ...
- it is plausible for all algorithms from specified primitives,from natural assumptions about such objects;
- *and it yields the best known lower bounds for the nullity and coprimeness problems from various primitives*

Outline

(1) Preliminaries
(2) Uniform processes and the Homomorphism Test
(3) Coprimeness in $\mathbb{N}$
(4) Polynomial 0-testing

Is the Euclidean algorithm optimal among its peers? (with vDD, 2004)
Arithmetic complexity (with van Den Dries, 2009)
Abstract recursion and intrinsic complexity,
forthcoming (2018) in the ASL Lecture Notes in Logic series

Y. Mansour, B. Schieber, and P. Tiwari (1991)
A lower bound for integer greatest common divisor computations
Lower bounds for computations with the floor operation
J. Meidânis (1991): *Lower bounds for arithmetic problems*
P. Bürgisser and T. Lickteig (1992)
Verification complexity of linear prime ideals
P. Bürgisser, T. Lickteig, and M. Shub (1992),
Test complexity of generic polynomials

(Partial) structures

• A vocabulary is a finite set Φ of function symbols, each with a specified arity n_ϕ and sort $s_\phi \in \{\mathtt{a}, \mathtt{boole}\}$; and a (partial) Φ-structure is a pair

$$\mathbf{A} = (A, \Phi^{\mathbf{A}}) = (A, \{\phi^{\mathbf{A}}\}_{\phi\in\Phi}),$$

where $\boxed{\phi^{\mathbf{A}} : A^{n_\phi} \rightharpoonup A_{s_\phi}}$ with $A_{\mathtt{a}} = A$ and $A_{\mathtt{boole}} = \{\mathrm{tt}, \mathrm{ff}\}$

• $\mathbf{N} = (\mathbb{N}, 0, 1, +, \cdot, =)$, the standard structure of arithmetic

• $\mathbf{A} \upharpoonright U = (U, \{\phi^{\mathbf{A}} \upharpoonright U\}_{\phi\in\Phi})$, for any $U \subseteq A$ and $f : A^n \rightharpoonup A_s$, with

$$(f \upharpoonright U)(\vec{x}) = w \iff \vec{x} \in U^n, w \in U_s \;\&\; f(\vec{x}) = w$$

• The (equational) diagram of a Φ-structure is the set of its basic equations,

$$\mathrm{eqdiag}(\mathbf{A}) = \{(\phi, \vec{x}, w) : \vec{x} \in A^{n_\phi}, w \in A_{s_\phi},\ \text{and } \phi^{\mathbf{A}}(\vec{x}) = w\}$$

• We allow $A = \emptyset$ and $\phi^{\mathbf{A}}$ the totally undefined n_ϕ-ary partial function with values in A_{s_ϕ}, in which case $\mathrm{eqdiag}(\mathbf{A}) = \emptyset$

Substructures and homomorphisms

- Substructures (pieces): For any two Φ-structures $\mathbf{U}, \mathbf{A}$:

$$\mathbf{U} \subseteq_p \mathbf{A} \iff U \subseteq A \;\&\; (\forall \phi \in \Phi)[\phi^{\mathbf{U}} \sqsubseteq \phi^{\mathbf{A}}]$$

Substructures may be finite and not closed under $\mathbf{\Phi}$

- Generated substructures: With $\vec{x} = (x_1, \ldots, x_n)$, $G_0(\mathbf{A}, \vec{x}) = \{\vec{x}\}$ and

$$G_{k+1}(\mathbf{A}, \vec{x}) = G_k(\mathbf{A}, \vec{x}) \cup \{\phi^{\mathbf{A}}(t_1, \ldots, t_m) \mid t_1, \ldots, t_m \in G_k(\mathbf{A}, \vec{x})\}$$

$\mathbf{A}$ *is generated by* $\vec{x}$ *if* $A = \bigcup_k G_k(\mathbf{A}, \vec{x})$

- A homomorphism $\pi : \mathbf{U} \to \mathbf{V}$ is any $\pi : U \to V$ such that for all $\phi \in \Phi, x_1, \ldots, x_n \in U, w \in U_s$, (with $\pi(\mathrm{tt}) = \mathrm{tt}, \pi(\mathrm{ff}) = \mathrm{ff}$)

$$\phi^{\mathbf{U}}(x_1, \ldots, x_n) = w \implies \phi^{\mathbf{V}}(\pi x_1, \ldots, \pi x_n) = \pi w$$

- May have $x \neq y, \pi(x) = \pi(y)$, unless $(=, x, y, \mathrm{ff}) \in \mathrm{eqdiag}(\mathbf{U})$
- π is an embedding if it is injective (in which case it preserves $\neq$)

- We use finite substructures $\mathbf{U} \subseteq_p \mathbf{A}$ to represent calls to the primitives by an algorithm during a computation in $\mathbf{A}$

Algorithms from primitives – the basic intuition

- An n-ary algorithm α of $\mathbf{A} = (A, \mathbf{\Phi})$ (or from $\mathbf{\Phi}$) "computes" some n-ary partial function or relation

$$\overline{\alpha} = \overline{\alpha}^{\mathbf{A}} : A^n \rightharpoonup A_s$$

using the primitives in $\mathbf{\Phi}$ as oracles and nothing else about $\mathbf{A}$

- We understand this to mean that in the course of a "computation" of $\overline{\alpha}(\vec{x})$, the algorithm may request from the oracle for any $\phi^{\mathbf{A}}$ any particular value $\phi^{\mathbf{A}}(\vec{u})$, for arguments $\vec{u}$ which it has already computed from $\vec{x}$, and that if the oracles cooperate, then "the computation" of $\overline{\alpha}(\vec{x})$ is completed in a finite number of "steps"

- The notion of a uniform process, coming up next, attempts to capture minimally (in the style of abstract model theory) these aspects of algorithms from specified primitives

- It does not capture their effectiveness, but their uniformity —that an algorithm applies "the same" (possibly not effective or non-deterministic) "procedure" to all arguments in its input set

Uniform processes: I The Locality Axiom

- *A* *uniform process* α *of arity n and sort s of a structure* $\mathbf{A} = (A, \Phi^{\mathbf{A}})$ *assigns to each substructure* $\mathbf{U} \subseteq_p \mathbf{A}$ *an n-ary partial function*

$$\overline{\alpha}^{\mathbf{U}} : U^n \rightharpoonup U_s$$

It defines the partial function or relation $\overline{\alpha}^{\mathbf{A}} : A^n \rightharpoonup A_s$

- For an algorithm α, intuitively, $\overline{\alpha}^{\mathbf{U}}$ is the restriction to U of the partial function computed by α when the oracles respond only to questions with answers in eqdiag($\mathbf{U}$)
- We sometimes use the notation for satisfaction,

$$\mathbf{U} \models \alpha(\vec{x}) = w \iff \overline{\alpha}^{\mathbf{U}}(\vec{x}) = w,$$

$$\mathbf{U} \models \alpha(\vec{x})\downarrow \iff (\exists w)[\overline{\alpha}^{\mathbf{U}}(\vec{x}) = w]$$

Uniform processes: II The Homomorphism Axiom

- *If α is an n-ary uniform process of* $\mathbf{A}$, $\mathbf{U}, \mathbf{V} \subseteq_p \mathbf{A}$, *and* $\pi : \mathbf{U} \to \mathbf{V}$ *is a homomorphism, then*

$$\mathbf{U} \models \alpha(\vec{x}) = w \implies \mathbf{V} \models \alpha(\pi\vec{x}) = \pi w \quad (x_1, \ldots, x_n \in U, w \in U_s)$$

In particular, if $\mathbf{U} \subseteq_p \mathbf{A}$, then $\overline{\alpha}^{\mathbf{U}} \sqsubseteq \overline{\alpha}^{\mathbf{A}}$

- For algorithms: when asked for $\phi^{\mathbf{U}}(\vec{x})$, the oracle for ϕ may consistently provide $\phi^{\mathbf{V}}(\pi\vec{x})$, if π is a homomorphism
- The Homomorphism Axiom is obvious for the identity embedding $I : \mathbf{U} \rightarrowtail \mathbf{A}$, but it is a strong restriction for algorithms from rich primitives (stacks, higher type constructs, etc.)
- It can be verified for all (deterministic and non-deterministic) algorithms specified by the standard computation models, **provided all their primitives are included in $\mathbf{\Phi}$**

Uniform processes: III The Finiteness Axiom

- *If α is an n-ary uniform process of* $\mathbf{A}$, *then*

$$\mathbf{A} \models \alpha(\vec{x}) = w$$
$$\Longrightarrow \textit{ there is a finite } \mathbf{U} \subseteq_p \mathbf{A} \textit{ generated by } \vec{x} \textit{ such that } \mathbf{U} \models \alpha(\vec{x}) = w$$

- For every call $\phi(\vec{u})$ to the primitives, the algorithm must construct the arguments $\vec{u}$, and so the entire computation takes place within a finite substructure generated by the input $\vec{x}$
- We write

$$\mathbf{U} \vdash_c \alpha(\vec{x}) = w \iff \mathbf{U} \text{ is finite, generated by } \vec{x} \text{ and } \mathbf{U} \models \alpha(\vec{x}) = w,$$
$$\mathbf{U} \vdash_c \alpha(\vec{x})\downarrow \iff (\exists w)[\mathbf{U} \vdash_c \alpha(\vec{x}) = w]$$

We read $\mathbf{U} \vdash_c \alpha(\vec{x})\downarrow$ as "$\mathbf{U}$ *computes* $\alpha(\vec{x})$"
and we think of $(\mathbf{U}, \vec{x}, w)$ as a computation of α on the input $\vec{x}$

⋆ Uniform processes, summary

- I The Locality Axiom: *A uniform process* α *of a structure* $\mathbf{A} = (A, \Phi^{\mathbf{A}})$ *with arity n and sort* $s \in \{\mathtt{a}, \mathtt{boole}\}$ *assigns to each substructure* $\mathbf{U} \subseteq_p \mathbf{A}$ *an n-ary partial function* $\overline{\alpha}^{\mathbf{U}} : U^n \rightharpoonup U_s$

It defines the partial function or relation $\overline{\alpha}^{\mathbf{A}} : A^n \rightharpoonup A_s$

$$\boxed{\mathbf{U} \models \alpha(\vec{x}) = w \iff \overline{\alpha}^{\mathbf{U}} = w, \quad \mathbf{U} \models \alpha(\vec{x})\downarrow \iff \overline{\alpha}^{\mathbf{U}}(\vec{x})\downarrow}$$

- II The Homomorphism Axiom: *If* $\mathbf{U}, \mathbf{V} \subseteq_p \mathbf{A}$ *and* $\pi : \mathbf{U} \to \mathbf{V}$ *is a homomorphism, then* $\overline{\alpha}^{\mathbf{U}}(\vec{x}) = w \implies \overline{\alpha}^{\mathbf{V}}(\pi\vec{x}) = \pi w$

$$\boxed{\mathbf{U} \vdash_c \alpha(\vec{x})\downarrow \iff \mathbf{U} \text{ is finite, generated by } \vec{x} \text{ and } \overline{\alpha}^{\mathbf{U}}(\vec{x})\downarrow}$$

- III The Finiteness Axiom: $\mathbf{A} \models \alpha(\vec{x})\downarrow \implies (\exists \mathbf{U} \subseteq_p \mathbf{A})[\mathbf{U} \vdash_c \alpha(\vec{x})\downarrow]$
- Uniform processes do not capture computability:
If $\mathbf{A}$ *is generated by its primitives, then every* $f : A^n \rightharpoonup A_s$ *is computed by a uniform process of* $\mathbf{A}$

Complexity measures generated by substructure norms

• For any vocabulary Φ, a Φ-substructure norm is an operation μ which assigns to every pair $(\mathbf{U}, \vec{x})$ of a finite Φ-structure $\mathbf{U}$ and a tuple $\vec{x} \in U^n$ that generates it a number $\mu(\mathbf{U}, \vec{x})$ and respects isomorphisms, i.e.,

$$(1) \quad \mathbf{U} \text{ is finite, generated by } x_1, \ldots, x_n \in U \ \& \ \pi : \mathbf{U} \rightarrowtail\!\!\!\!\twoheadrightarrow \mathbf{V} \\ \Longrightarrow \ \mu(\mathbf{U}, x_1, \ldots, x_n) = \mu(\mathbf{V}, \pi(x_1), \ldots, \pi(x_n))$$

• Example: $\mu(\mathbf{U}, \vec{x}) = |\text{eqdiag}(\mathbf{U})| =$ the size of the diagram of $\mathbf{U}$

• For any Φ-substructure norm μ, any n-ary uniform process α on a Φ-structure $\mathbf{A}$ and any $\vec{x} \in A^n$,

$$\boxed{C_\mu(\alpha, \vec{x}) = \min\{\mu(\mathbf{U}, \vec{x}) : \mathbf{U} \subseteq_p \mathbf{A} \ \& \ \mathbf{U} \vdash_c \alpha(\vec{x})\downarrow\}}$$

• These are the complexity measures to which we can apply the homomorphism method

⋆ Three basic complexity measures for uniform processes

- $\boxed{\text{calls}_{\Phi_0}(\alpha, \vec{x}) = \min\{|\text{eqdiag}(\mathbf{U} \upharpoonright \Phi_0)| : \mathbf{U} \vdash_c \alpha(\vec{x})\downarrow\} \quad (\Phi_0 \subseteq \Phi)}$

  $=$ the least number of calls to $\phi \in \Phi_0$ α must make to compute $\overline{\alpha}^{\mathbf{A}}(\vec{x})$

- $\mathbf{U}_{\text{vis}} = \{u \in U \mid u \text{ occurs in eqdiag}(\mathbf{U})\}$ (the visible part of $\mathbf{U}$)
- $\boxed{\text{size}(\alpha, \vec{x}) = \min\{|U_{\text{vis}}| : \mathbf{U} \vdash_c \alpha(\vec{x})\downarrow\}}$

  $=$ the least number of members of $\mathbf{A}$ that α must see)

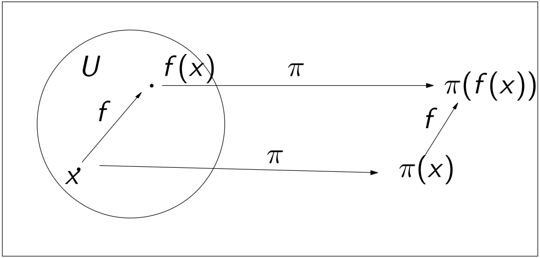
- $\boxed{\text{depth}(\alpha, \vec{x}) = \min\{\text{depth}(\mathbf{U}, \vec{x}) : \mathbf{U} \vdash_c \alpha(\vec{x})\downarrow\}}$

  $=$ the least number of calls α must execute in sequence

Theorem $\boxed{\text{depth}(\alpha, \vec{x}) \leq \text{size}(\alpha, \vec{x}) \leq \text{calls}(\alpha, \vec{x}) = \text{calls}_{\Phi}(\alpha, \vec{x})}$

- *These are not larger than similarly named standard complexity functions for algorithms defined by standard computation models* (at least for depth and calls)

★★ The forcing $\Vdash^{\mathbf{A}}$ and certification $\Vdash_c^{\mathbf{A}}$ relations



Suppose $f : A^n \rightharpoonup A_s$, $f(\vec{x})\downarrow$, $\mathbf{U} \subseteq_p \mathbf{A}$

- A homomorphism $\pi : \mathbf{U} \to \mathbf{A}$ respects f at $\vec{x}$ if

$$\boxed{\vec{x} \in U^n \;\&\; f(\vec{x}) \in U_s \;\&\; \pi(f(\vec{x})) = f(\pi(\vec{x}))}$$

so for relations $\boxed{\vec{x} \in U^n \;\&\; \Big(f(\vec{x}) \iff f(\pi(\vec{x}))\Big)}$

$$\boxed{\mathbf{U} \Vdash^{\mathbf{A}} f(\vec{x})\downarrow \iff \text{every homomorphism } \pi : \mathbf{U} \to \mathbf{A} \text{ respects } f \text{ at } \vec{x}}$$

$$\boxed{\mathbf{U} \Vdash_c^{\mathbf{A}} f(\vec{x})\downarrow \iff \mathbf{U} \text{ is finite, generated by } \vec{x} \text{ and } \mathbf{U} \Vdash^{\mathbf{A}} f(\vec{x})\downarrow}$$

⋆ The intrinsic μ-complexity in $\mathbf{A}$ of $f : A^n \rightharpoonup A_s$

$$\boxed{C_\mu(\mathbf{A}, f, \vec{x}) = \min\{\mu(\mathbf{U}, \vec{x}) : \mathbf{U} \Vdash_c^{\mathbf{A}} f(\vec{x})\downarrow\} \in \mathbb{N} \cup \{\infty\}}$$

Lemma

If μ is any Φ- substructure norm and α is a uniform process which computes $f : A^n \rightharpoonup A_s$ in a Φ-structure $\mathbf{A}$, then

$$C_\mu(\mathbf{A}, f, \vec{x}) \leq C_\mu(\alpha, \vec{x}) \qquad (f(\vec{x})\downarrow)$$

Lemma (The Homomorphism Test)

Suppose μ is a substructure norm (e.g., $\mathrm{calls}_{\Phi_0}, \mathrm{size}, \mathrm{depth}$), $\mathbf{A}$ is a Φ-structure, $\boxed{f : A^n \rightharpoonup A_s, f(\vec{x})\downarrow, m \in \mathbb{N}}$, and

for every finite $\mathbf{U} \subseteq_p \mathbf{A}$ which is generated by $\vec{x}$,

$$\Big(f(\vec{x}) \in U_s \ \&\ \mu(\mathbf{U}, \vec{x}) < m\Big) \implies (\exists \pi : \mathbf{U} \to \mathbf{A})[f(\pi(\vec{x})) \neq \pi(f(\vec{x}))];$$

then $\boxed{C_\mu(\mathbf{A}, f, \vec{x}) \geq m}$.

A lower bound for coprimeness on $\mathbb{N}$

$\mathbf{A} = (\mathbb{N}, 0, 1, +, \dot{-}, \mathrm{iq}, \mathrm{rem}, =, <, \mathbf{\Psi})$, $\mathbf{\Psi}$ a finite set of *Presburger functions*

Theorem (van den Dries, ynm, 2004, 2009)

If $\xi > 1$ is quadratic irrational, then for some $r > 0$ and all sufficiently large coprime (a, b),

$$(2) \qquad \left|\xi - \frac{a}{b}\right| < \frac{1}{b^2} \implies \mathrm{depth}(\mathbf{A}, \perp\!\!\!\perp, a, b) \geq r \log\log a$$

In particular, the conclusion of (2) *holds with some r*

- *for positive Pell pairs (a, b) satisfying $a^2 = 2b^2 + 1 \quad (\xi = \sqrt{2})$*
- *for Fibonacci pairs (F_{k+1}, F_k) with $k \geq 3 \quad (\xi = \frac{1}{2}(1 + \sqrt{5}))$*

Theorem (Pratt 2008, unpublished)

There is a non-deterministic algorithm ε_{nd} of $\mathbf{N}_\varepsilon$ which decides coprimeness, is at least as effective as the Euclidean everywhere and

$$\mathrm{calls}(\varepsilon_{nd}, F_{k+1}, F_k) \leq K \log\log F_{k+1}$$

- The theorem is best possible from its hypotheses

Horner's rule for polynomial 0-testing

The nullity relation on a field F:

$$N_F(a_0, \ldots, a_n, x) \iff a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n = 0$$

Theorem

Let F be the field of real or complex numbers
If $n \geq 1$ and $a_0, \ldots, a_n, x$ are algebraically independent, then:

(1) $\mathrm{calls}_{\{\cdot,\div\}}(F, N_F, \vec{a}, x) = n$

(2) $\mathrm{calls}_{\{\cdot,\div,=\}}(F, N_F, \vec{a}, x) = n + 1$

(3) $\mathrm{calls}_{\{+,-\}}(F, N_F, \vec{a}, x) = n - 1$

(4) $\mathrm{calls}_{\{+,-,=\}}(F, N_F, \vec{a}, x) = n$ (Horner needs $n + 1$)

• The method for constructing the required homomorphsm in (1) is an elaboration of Winograd's proof of the $\{\cdot, \div\}$-optimality of Horner's rule for poly evaluation

• For algebraic decision trees, (1) is due to Bürgisser and Lickteig (1992) and results similar to (3), (4) are due to Bürgisser, Lickteig and Shub (1992). These papers use very different methods

Two open problems about coprimeness

Let $\mathbf{A} = (\mathbb{N}, 0, 1, +, \dot{-}, \mathrm{iq}, \mathrm{rem}, =)$

(1) **Basic Conjecture II** *For some $r > 0$ and infinitely many pairs (a, b),*

$$\mathrm{calls}(\mathbf{A}, \perp\!\!\!\perp, a, b) \geq r \log \max(a, b)$$

• We proved this with a double log and Pratt's example shows that our proof does not establish it with a single log, but there may be an entirely different proof

(2) **Basic Conjecture III** *For every algorithm α of $\mathbf{A}$ expressed by a deterministic recursive program which decides the coprimeness relation, there is some $r > 0$ and infinitely many (a, b) such that*

$$\mathrm{calls}(\alpha, a, b) \geq r \log \max(a, b)$$

• The deterministic recursive programs of a structure $\mathbf{A}$ (arguably) express faithfully all the deterministic algorithms from $\mathbf{A}$

...and a general, vague open problem

- For a (total) structure $\mathbf{A}$ and a function $f : A^n \to A_s$, do any of the complexity functions

$$\vec{a} \mapsto \text{depth}(\mathbf{A}, f, \vec{a}), \quad \text{size}(\mathbf{A}, f, \vec{a}), \quad \text{calls}(\mathbf{A}; f, \vec{a})$$

encode interesting model theoretic properties of $(\mathbf{A}, f)$?

... perhaps for specific, algebraic structures and "natural" f?

THE END