

# Intrinsic complexities

Yiannis N. Moschovakis  
UCLA and University of Athens

Institute of Mathematics, Prague, February 8, 2016

A preliminary and incomplete version of this talk  
was given at a workshop in Bristol in 2013

# The Euclidean algorithm

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

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

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

$\text{calls}_{\{\text{rem}\}}(\varepsilon, a, b) =$  the number of divisions  $\varepsilon$  needs to compute  $\text{gcd}(a, b)$   
 $\leq 2 \log(b)$  ( $a \geq b \geq 2$ )

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

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

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

- *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  $F$ :

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

**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  $a_0, \dots, a_n, x$

# The Value Complexity II

## Theorem (van den Dries)

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

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

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

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

... because it takes at least that many applications of the primitives to **construct the value**  $\gcd(a + 1, b)$  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  $F$ :

$$N_F(a_0, \dots, a_n, x) \iff a_0 + a_1x + a_2x^2 + \dots + a_nx^n = 0$$

from some or all of the field primitives and =

- The **coprimeness relation**

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

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

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

$$\text{calls}_R : A^n \rightarrow \mathbb{N} \cup \{\infty\}$$

with the following properties:

- (1) If  $\text{calls}_R(\vec{x}) = 0$  for some tuple  $\vec{x}$  of distinct elements, then  $R$  is constant on  $A^n$ , and
- (2) If  $\alpha$  is any (deterministic or non-deterministic) algorithm which decides  $R$  from the primitives of  $\mathbf{A}$ , then for every  $\vec{x}$ ,

$$\text{calls}_R(\vec{x}) \leq \text{the number of calls to the primitives} \\ \text{that } \alpha \text{ needs to make to decide } R(\vec{x})$$

- We will define  $\text{calls}_R$  from  $R$  and  $\mathbf{A}$  with no reference to “algorithms”; (2) is a theorem for algorithms specified by *computation models*; it is plausible for all “algorithms from specific 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***, monograph (in preparation)

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 set  $\Phi$  of function symbols, each with a specified **arity**  $n_\phi$  and **sort**  $s_\phi \in \{\mathbf{a}, \mathbf{bool}\}$ ; and a (partial)  $\Phi$ -**structure** is a pair

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

where  $\phi^{\mathbf{A}} : A^{n_\phi} \rightarrow A_{s_\phi}$  with  $A_{\mathbf{a}} = A$  and  $A_{\mathbf{bool}} = \{\mathbf{tt}, \mathbf{ff}\}$

- $\mathbf{N} = (\mathbb{N}, 0, 1, +, \cdot, =)$ , the standard structure of arithmetic
- $\mathbf{A} \upharpoonright U = (U, \{\phi^{\mathbf{A}} \upharpoonright U\}_{\phi \in \Phi})$ , for any  $U \subseteq A$  and  $f : A^n \rightarrow A_s$ , with

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

- The (equational) **diagram** of a  $\Phi$ -structure is the set of its basic equations,

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

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



## Substructures and homomorphisms

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

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

*Substructures may be finite and not closed under  $\Phi$*

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

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

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

- A **homomorphism**  $\pi : \mathbf{U} \rightarrow \mathbf{V}$  is any  $\pi : U \rightarrow V$  such that for all  $\phi \in \Phi, x_1, \dots, x_n \in U, w \in U_s$ , (with  $\pi(\mathbb{t}) = \mathbb{t}, \pi(\text{ff}) = \text{ff}$ )

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

- May have  $x \neq y, \pi(x) = \pi(y)$ , unless  $(=, x, y, \text{ff}) \in \text{eqdiag}(\mathbf{U})$
- $\pi$  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**  $\alpha$  of  $\mathbf{A} = (A, \Phi)$  (or **from**  $\Phi$ ) “computes” some  $n$ -ary partial function or relation

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

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

- We understand this to mean that in the course of a “computation” of  $\bar{\alpha}(\vec{x})$ , 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  $\bar{\alpha}(\vec{x})$  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*  $\alpha$  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

$$\bar{\alpha}^{\mathbf{U}} : U^n \rightarrow U_s$$

It *computes* the partial function or relation  $\bar{\alpha}^{\mathbf{A}} : A^n \rightarrow A_s$

- For an algorithm  $\alpha$ , intuitively,  $\bar{\alpha}^{\mathbf{U}}$  is the restriction to  $U$  of the partial function computed by  $\alpha$  *when the oracles respond only to questions with answers in  $\text{eqdiag}(\mathbf{U})$*
- We write

$$\begin{aligned} \mathbf{U} \vdash \alpha(\vec{x}) = w &\iff \bar{\alpha}^{\mathbf{U}}(\vec{x}) = w, \\ \mathbf{U} \vdash \alpha(\vec{x}) \downarrow &\iff (\exists w)[\bar{\alpha}^{\mathbf{U}}(\vec{x}) = w] \end{aligned}$$

## Uniform processes: II The Homomorphism Axiom

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

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

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

- For algorithms: when asked for  $\phi^{\mathbf{U}}(\vec{x})$ , the oracle for  $\phi$  may consistently provide  $\phi^{\mathbf{V}}(\pi\vec{x})$ , if  $\pi$  is a homomorphism
- This is obvious for the identity embedding  $I : \mathbf{U} \hookrightarrow \mathbf{A}$ , 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  $\Phi$**

## Uniform processes: III The Finiteness Axiom

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

$$\mathbf{A} \vdash \alpha(\vec{x}) = w$$

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

- 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

$\mathbf{U} \vdash_c \alpha(\vec{x}) = w \iff \mathbf{U}$  is finite, generated by  $\vec{x}$  and  $\mathbf{U} \vdash \alpha(\vec{x}) = w$ ,

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

and we think of  $(\mathbf{U}, \vec{x}, w)$  as a **computation** of  $\alpha$  on the input  $\vec{x}$

## ★ Uniform processes, summary

- I The **Locality Axiom**: A uniform process  $\alpha$  of a structure  $\mathbf{A} = (A, \Phi^{\mathbf{A}})$  with arity  $n$  and sort  $s \in \{\mathbf{a}, \mathbf{bool}\}$  assigns to each substructure  $\mathbf{U} \subseteq_p \mathbf{A}$  an  $n$ -ary partial function  $\bar{\alpha}^{\mathbf{U}} : U^n \rightarrow U_s$   
It **computes** the partial function or relation  $\bar{\alpha}^{\mathbf{A}} : A^n \rightarrow A_s$

$$\mathbf{U} \vdash \alpha(\vec{x}) = w \iff \bar{\alpha}^{\mathbf{U}} = w, \quad \mathbf{U} \vdash \alpha(\vec{x}) \downarrow \iff \bar{\alpha}^{\mathbf{U}}(\vec{x}) \downarrow$$

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

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

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

## Complexity measures generated by substructure norms

- For any vocabulary  $\Phi$ , a  $\Phi$ -**substructure norm** is an operation  $\mu$  which assigns to every pair  $(\mathbf{U}, \vec{x})$  of a finite  $\Phi$ -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.,

$$(1) \quad \pi : \mathbf{U} \twoheadrightarrow \mathbf{V} \ \& \ \mathbf{U} \text{ is generated by } x_1, \dots, x_n \in U \\ \implies \mu(\mathbf{U}, x_1, \dots, x_n) = \mu(\mathbf{V}, \pi(x_1), \dots, \pi(x_n))$$

- Example:  $\mu(\mathbf{U}, \vec{x}) = |\text{eqdiag}(\mathbf{U})|$
- For any  $\Phi$ -substructure norm  $\mu$ , any  $n$ -ary uniform process  $\alpha$  on a  $\Phi$ -structure  $\mathbf{A}$  and any  $\vec{x} \in A^n$ ,

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

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

## ★ The 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  $\phi \in \Phi_0$   $\alpha$  **must make** to compute  $\bar{\alpha}^{\mathbf{A}}(\vec{x})$
- $\mathbf{U}_{\text{vis}} = \{u \in U \mid u \text{ occurs in } \text{eqdiag}(\mathbf{U})\}$  (the **visible** part of  $\mathbf{U}$ )
- $\boxed{\text{size}(\alpha, \vec{x}) = \min\{|\mathbf{U}_{\text{vis}}| : \mathbf{U} \vdash_c \alpha(\vec{x}) \downarrow\}}$   
= the least number of members of  $\mathbf{A}$  that  $\alpha$  **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  $\alpha$  **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 and certification relations

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

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

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

so for relations  $\vec{x} \in U^n \ \& \ (R(\vec{x}) \iff R(\pi(\vec{x})))$

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

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

The **intrinsic complexities** of  $f$  in  $\mathbf{A}$

- $C_\mu(\mathbf{A}, f, \vec{x}) = \min\{\mu(\mathbf{U}, \vec{x}) : \mathbf{U} \Vdash_c^{\mathbf{A}} f(\vec{x}) \downarrow\} \in \mathbb{N} \cup \{\infty\}$
- $\text{calls}_{\Phi_0}(\mathbf{A}, f, \vec{x}) = \min\{|\text{eqdiag}(\mathbf{U} \upharpoonright \Phi_0)| : \mathbf{U} \Vdash_c^{\mathbf{A}} f(\vec{x}) \downarrow\}$
- $\text{size}(\mathbf{A}, f, \vec{x}) = \min\{|\mathbf{U}_{\text{vis}}| : \mathbf{U} \Vdash_c^{\mathbf{A}} f(\vec{x}) \downarrow\}$
- $\text{depth}(\mathbf{A}, f, \vec{x}) = \min\{\text{depth}(\mathbf{U}, \vec{x}) : \mathbf{U} \Vdash_c^{\mathbf{A}} f(\vec{x}) \downarrow\}$

## ★ Deriving lower bounds by constructing homomorphisms

- The following two facts are immediate from the definitions:

### Lemma

If  $\mu$  is any  $\Phi$ -substructure norm and  $\alpha$  is a uniform process which computes  $f : A^n \rightarrow A_s$  in a  $\Phi$ -structure  $\mathbf{A}$ , then

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

### Lemma (The homomorphism test)

Suppose  $\mu$  is a substructure norm (e.g.,  $\text{calls}_{\Phi_0}$ , size, depth),  $\mathbf{A}$  is a  $\Phi$ -structure,  $f : A^n \rightarrow A_s$ ,  $f(\vec{x}) \downarrow$ , and

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

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

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

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

$\mathbf{A} = (\mathbb{N}, 0, 1, +, \cdot, \text{iq}, \text{rem}, =, <, \Psi)$ ,  $\Psi$  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  $(a, b)$ ,

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

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

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

Theorem (Pratt, unpublished)

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

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

- The theorem is best possible from its hypotheses

## Non-uniform complexity

Given  $N$ , how good can a coprimeness algorithm be if we only insist that it works for and uses only  $N$ -bit numbers?

$\mathbf{A} = (\mathbb{N}, 0, 1, +, \cdot, \text{iq}, \text{rem}, =, <, \Psi)$  as before.

For any  $N$ , and any one of the intrinsic complexities as above, let

$$C_{\mu}(\mathbf{A}, f, 2^N) = \max\{C_{\mu}(\mathbf{A} \upharpoonright [0, 2^N), f, \vec{x}) : x_1, \dots, x_n < 2^N\}$$

Theorem (van den Dries, ynm 2009)

For some rational number  $r > 0$  and all sufficiently large  $N$ ,

$$\text{calls}(\mathbf{A}, \perp, 2^N) \geq \text{size}(\mathbf{A}, \perp, 2^N) \geq r \log N.$$

- Non-uniform lower bound for  $\text{depth}(\mathbf{A}, \perp, 2^N)$ ?

## Horner's rule for polynomial 0-testing

The **nullity relation** on a field  $F$ :

$$N_F(a_0, \dots, a_n, x) \iff a_0 + a_1x + a_2x^2 + \dots + a_nx^n = 0$$

### Theorem

Let  $F$  be the field of real or complex numbers

If  $n \geq 1$  and  $a_0, \dots, a_n, x$  are algebraically independent, then:

- (1)  $\text{calls}_{\{., \div\}}(F, N_F, \vec{a}, x) = n$
  - (2)  $\text{calls}_{\{., \div, =\}}(F, N_F, \vec{a}, x) = n + 1$
  - (3)  $\text{calls}_{\{+, -\}}(F, N_F, \vec{a}, x) = n - 1$
  - (4)  $\text{calls}_{\{+, -, =\}}(F, N_F, \vec{a}, x) = n$  (Horner needs  $n + 1$ )
- The method for constructing the required homomorphism in (1) is an elaboration of Winograd's proof of the  $\{., \div\}$ -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 by very different methods

## Two open problems about coprimeness

Let  $\mathbf{A} = (\mathbb{N}, 0, 1, +, \cdot, \text{iq}, \text{rem}, =)$

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

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

- 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  $\alpha$  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*

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

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

## ... and a general, vague open problem

- For a (total) structure  $\mathbf{A}$  and a function  $f : A^n \rightarrow A_s$ , 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”  $f$ ?