The axiomatic derivation of absolute lower bounds

Yiannis N. Moschovakis
UCLA and University of Athens

Iowa State University, April 15, 2010
The Euclidean algorithm

For \( x, y \in \mathbb{N} = \{0, 1, \ldots\}, \ x \geq y \geq 1, \)

\[
gcd(x, y) = \begin{cases} 
  y & \text{if } (\text{rem}(x, y) = 0) \text{ then } \text{y} \text{ else } \gcd(y, \text{rem}(x, y)) 
\end{cases}
\]

where \( \text{rem}(x, y) \) is the remainder of the division of \( x \) by \( y \),

\[
x = \text{iq}(x, y) \cdot y + \text{rem}(x, y) \quad (0 \leq \text{rem}(x, y) < y)
\]

calls\(_{\varepsilon}(x, y) = \) the number of divisions required to compute \( \gcd(x, y) \) by the Euclidean algorithm

\[
\leq 2 \log(y) \quad (x \geq y \geq 2)
\]

- Is the Euclidean optimal for computing \( \gcd(x, y) \) from \( \text{rem} \)?
- Is the Euclidean optimal for deciding coprimeness from \( \text{rem} \)?

\[
x \perp y \iff \gcd(x, y) = 1
\]
Theorem (van den Dries, ynm, 2004)

If an algorithm $\alpha$ decides the coprimeness relation $x \perp y$ on $\mathbb{N}$ from the primitives $\leq, +, -, \text{iq}, \text{rem}$, then for infinitely many $a, b$ with $a > b$,

$$\text{calls}_\alpha(a, b) \geq \text{depth}_\alpha(a, b) > \frac{1}{10} \log \log a \quad (*)$$

where $\text{depth}_\alpha(a, b)$ is the least number of applications of the primitives which must be executed in sequence in the computation.

- depth$_\alpha(x, y)$ is a natural parallel time complexity
- The result is one log short of establishing the optimality of the Euclidean (and one log = $\infty$ in this context)
- (*) holds for all sufficiently large $a, b$ such that
  - $a^2 = 1 + 2b^2$ (solutions of Pell’s equation),
  - or $a = F_{n+1}, b = F_n$ (successive Fibonacci numbers)
- Claim: This applies to all algorithms from the specified primitives
Outline

Slogan: *Absolute lower bound results are the undecidability facts about decidable problems*

...and so their precise formulation should be a matter of logic

(1) Tweak logic (a bit) so it applies smoothly to computation theory
(2) Three (simple) axioms for elementary algorithms, in the style of *abstract model theory*
(3) Verify that the axioms are satisfied by all computation models
(4) Derive lower bounds from the axioms

*Is the Euclidean algorithm optimal among its peers? (with vDD, 2004)*

*Arithmetic complexity* (with vDD, 2009)

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*
A (partial, pointed) algebra is a structure $\mathbf{M} = (M, 0, 1, \Phi^M)$ where $0, 1 \in M$, $\Phi$ is a set of function symbols (the vocabulary) and $\Phi^M = \{\phi^M\}_{\phi \in \Phi}$, where each primitive $\phi^M : M^{n_\phi} \to M$ is a partial function of arity $n_\phi$ determined by the symbol $\phi$

$\mathbb{N}_\varepsilon = (\mathbb{N}, 0, 1, \text{rem})$, the Euclidean algebra

$\mathbb{N}_u = (\mathbb{N}, 0, 1, S, \text{Pd})$, the unary numbers

$\mathbb{N}_b = (\mathbb{N}, 0, 1, \text{Parity}, \text{iq}_2, (x \mapsto 2x), (x \mapsto 2x + 1))$, the binary numbers

$\mathbb{N} = (\mathbb{N}, 0, 1, +, \div, \text{iq}, \text{rem}, \cdot)$, the full algebra of arithmetic

$\mathbb{N}_\varepsilon \upharpoonright U = (U, 0, 1, \text{rem} \upharpoonright U)$ where $\{0, 1\} \subseteq U \subseteq \mathbb{N}$ and

$$(\text{rem} \upharpoonright U)(x, y) = w \iff x, y, w \in U \land \text{rem}(x, y) = w$$

The diagram of a $\Phi$-algebra is the set of its basic identities,

$$\text{diag}(\mathbf{M}) = \{(\phi, \vec{x}, w) : \phi^M(\vec{x}) = w\}$$

$\mathbf{M}$ is completely determined by $M, 0, 1$ and $\text{diag}(\mathbf{M})$
Homomorphisms, embeddings and subalgebras

- A homomorphism $\pi : U \rightarrow M$ of one $\Phi$-algebra into another is any function $\pi : U \rightarrow M$ such that
  \[
  \pi(0^U) = y \iff y = 0^M, \quad \pi(1^U) = y \iff y = 1^M,
  \]
  and for all $\phi \in \Phi, x_1, \ldots, x_n, w \in U$,
  \[
  \phi^U(x_1, \ldots, x_n) = w \implies \phi^M(\pi x_1, \ldots, \pi x_n) = \pi w
  \]
- $\pi$ is an embedding if, in addition, it is one-to-one
- Subalgebras:
  \[
  U \subseteq_p M \iff \{0, 1\} \subseteq U \subseteq M
  \]
  \[
  \text{& the identity } I : U \leftrightarrow M \text{ is an embedding}
  \]
  \[
  \iff \{0, 1\} \subseteq U \subseteq M \& \text{diag}(U) \subseteq \text{diag}(M)
  \]
- We will use finite subalgebras $U \subseteq_p M$ to represent calls to the primitives executed during a computation in $M$
Terms and depth

The terms of a vocabulary $\Phi$: $t \equiv 0 \mid 1 \mid \nu \mid \phi(t_1, \ldots, t_n)$

\[
\phi(u, \psi(\chi(0)), \psi(\nu)) : 3
\]

$u : 0$ $\psi(\chi(0)) : 2$ $\psi(\nu) : 1$

$\chi(0) : 1$ $\nu : 0$

$0 : 0$

$\text{depth}(0) = \text{depth}(1) = \text{depth}(\nu) = 0,$

$\text{depth}(\phi(t_1, \ldots, t_n)) = \max(\text{depth}(t_1), \ldots, \text{depth}(t_n)) + 1$
Term evaluation and algebra generation

- If $\vec{v} = v_1, \ldots, v_n$ includes all the variables which occur in $t$,

$$t(\vec{v}) = (t, \vec{v})$$

- For any subalgebra $U \subseteq pM$ and any $\vec{x} = x_1, \ldots, x_n \in U$,

$$t^U[\vec{x}] = \text{the value of } t \text{ in } U \text{ when } \vec{v} := \vec{x} \text{ (if it converges)}$$

- $G_m[U, \vec{x}] = \{ t^U[\vec{x}] : \text{depth}(t(\vec{v})) \leq m \& t^U[\vec{x}] \downarrow \}$

- $G_\infty[U, \vec{x}] = \bigcup_m G_m[U, \vec{x}] = \text{the subalgebra of } U \text{ generated by } \vec{x}$

- $\text{depth}(w, U, \vec{x}) = \min\{m : w \in G_m[U, \vec{x}]\}$

$\text{depth}(w, U, \vec{x})$ is the least number of applications of the primitives which must be executed in sequence to construct $w$ from $\vec{x}$ in $U$

- $\text{depth}_{\vec{x}}(U) = \max\{\text{depth}(w, U, \vec{x}) : w \in U\}$ (U generated by $\vec{x}$)
The depth complexity of values of a function

**Basic principle:** If an algorithm $\alpha$ computes $f : M^n \rightarrow M$ from the primitives of $M$, then $f(\vec{x}) \in G_\infty[M, \vec{x}]$ and

$$\text{calls}_\alpha(\vec{x}) \geq \text{depth}(f(\vec{x}), M, \vec{x})$$

- The value must be constructed by the primitives from the input
- Can be used to derive lower bounds for functions which grow fast, e.g., multiplication or the Ackermann function (in unary or binary arithmetic)

We can also exploit gaps in $G_m[M, a]$ when $a$ is large compared to $m$:

$$G_m[N_b, a] : 0 \ 1 \ 2 \ \cdots \ 2^{m+1} - 1 \ \text{gap} \ \text{iq}(a, 2^m) \ \cdots \ a \ \cdots$$
**Theorem** (van den Dries)

*If an algorithm $\alpha$ computes $\gcd(x, y)$ from*

\[+ \div < \equiv \text{ iq rem} \cdot\]

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

\[\text{calls}_\alpha(a + 1, b) \geq \text{depth}(a + 1, b) \geq \frac{1}{4} \sqrt{\log \log b} \]

This is the best known lower bound for the gcd from primitives which include multiplication

▶ **This method cannot yield lower bounds for decision problems**

(because their value (0 or 1) is available with no computation)
The Locality Axiom

An algorithm \( \alpha \) of arity \( n \) of an algebra \( M = (M, 0, 1, \Phi^M) \) assigns to each subalgebra \( U \subseteq_p M \) an \( n \)-ary (strict) partial function

\[
\overline{\alpha}^U : U^n \rightarrow U
\]

An \( M \)-algorithm \( \alpha \) “computes” a partial function \( \overline{\alpha}^M : M^n \rightarrow M \) using the primitives of \( M \) as oracles, and it can be naturally localized (restricted) to arbitrary subalgebras of \( M \).

We write

\[
U \vdash \overline{\alpha}(\vec{x}) = w \iff \vec{x} \in U^n, w \in U \text{ and } \overline{\alpha}^U(\vec{x}) = w
\]
II The Homomorphism Axiom

If $\alpha$ is an $n$-ary algorithm of $M$, $U, V \subseteq_p M$, and $\pi : U \to V$ is a homomorphism, then

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

In particular, if $U \subseteq_p M$, then $\overline{\alpha}^U \subseteq \overline{\alpha}^M$

- When asked for $\phi^U(\vec{x})$, the oracle for $\phi$ may consistently provide $\phi^V(\pi \vec{x})$, if $\pi$ is a homomorphism.
III The Finiteness Axiom

If $\alpha$ is an $n$-ary algorithm of $\mathbf{M}$, then

$$\mathbf{M} \vdash \overline{\alpha}(\overline{x}) = w$$

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

- The algorithm must construct the arguments $\overline{u}$ for every call $\phi(\overline{u})$ to the primitives, and so the entire computation takes place within the subalgebra generated by the input $\overline{x}$
- A computation of $\overline{\alpha}^\mathbf{M}(\overline{x})$ from the primitives of $\mathbf{M}$ is finite, and so it makes finitely many calls to the primitives
- Intuitively, the axiom is satisfied by any $U$ whose diagram includes $(\phi, \overline{u}, \phi^\mathbf{M}(\overline{u}))$ for every call to $\phi^\mathbf{M}$ made by $\alpha$ during the computation
All elementary algorithms satisfy I – III (with suitable \( M \))

- Explicit computation: \( \overline{\alpha}^M (\vec{x}) = t^M[\vec{x}] \), where \( t(\vec{v}) \) is a \( \Phi \)-term

- \( \overline{\alpha}^M \) is the partial function computed a fixed recursive (McCarthy) program \( P \) in the vocabulary \( \Phi \)

- \( \overline{\alpha}^M \) is computed from \( \Phi^M \) by any of the familiar machine models of computation with oracles—register machines, Random Access Machines (of all kinds), Turing machines, etc.

- \( \overline{\alpha}^M \) is computed in PCF (typed \( \lambda \)-calculus) above the algebra \( M \)

- \( \overline{\alpha}^M \) by computed by non-deterministic versions of any of these

**Note:** For computation models (e.g., Turing machines), the functions built into the model must be included among the primitives of \( M \).
Abstract algorithms

- **I, Locality Axiom:** An abstract algorithm \( \alpha \) of arity \( n \) of an algebra \( M = (M, 0, 1, \Phi^M) \) assigns to each subalgebra \( U \subseteq_p M \) an \( n \)-ary partial function \( \overline{\alpha}^U : U^n \rightarrow U \)

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

- **II, Homomorphism Axiom:** If \( U, V \subseteq_p M \), and \( \pi : U \rightarrow V \) is a homomorphism, then

\[
U \models \overline{\alpha}(\vec{x}) = w \implies V \models \overline{\alpha}(\pi \vec{x}) = \pi w
\]

Set

\[
U \models g \overline{\alpha}(\vec{x}) = w \iff U \text{ is generated by } \vec{x} \text{ & } U \models \overline{\alpha}(\vec{x}) = w
\]

- **III, Finiteness Axiom:**

\[
M \models \overline{\alpha}(\vec{x}) = w \implies \text{there is a finite } U \subseteq_p M \text{ such that } U \models g \overline{\alpha}(\vec{x}) = w
\]
Complexity functions for abstract algorithms

Suppose $\alpha$ is an $n$-ary abstract algorithm of $\mathbf{M}$ and $\mathbf{M} \vdash \overline{\alpha}(\vec{x}) = w$

- $\text{calls}_\alpha(\vec{x}) = \min\{|\text{diag}(U)| : U \vdash \beta(\vec{x}) = w\}$
  (the least number of calls $\alpha$ must execute to compute $\overline{\alpha}^\mathbf{M}(\vec{x})$)

- $\text{size}_\alpha(\vec{x}) = \min\{|U \setminus \{0, 1, \vec{x}\}| : U \vdash \beta(\vec{x}) = w\}$
  (the least number of elements of $\mathbf{M}$ (other than 0, 1, $\vec{x}$) that $\alpha$ must see to compute $\overline{\alpha}^\mathbf{M}(\vec{x})$)

- $\text{depth}_\alpha(\vec{x}) = \min\{\text{depth}_\beta(U) : U \vdash \beta(\vec{x}) = w\}$
  (the least number of calls $\alpha$ must execute in sequence to compute $\overline{\alpha}^\mathbf{M}(\vec{x})$)

Thm $\text{depth}_\alpha(\vec{x}) \leq \text{size}_\alpha(\vec{x}) \leq \text{calls}_\alpha(\vec{x})$

These notions agree with standard definitions for concrete algorithms.
The certification relation $U \vdash f(\bar{x}) = w$

Suppose $f : M^n \to M$, $U \subseteq_p M$.

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

$$\bar{x} \in U \land f(\bar{x}) \in U \land \pi(f(\bar{x})) = f(\pi(\bar{x}))$$

$U \vdash f(\bar{x}) = w \iff$ every homomorphism $\pi : U \to M$ respects $f$ at $\bar{x}$

**Lemma**

(1) *If an abstract $M$-algorithm $\alpha$ computes $f : M^n \to M$, then*

$$U \vdash_{g} \bar{\alpha}(\bar{x}) = w \implies U \vdash f(\bar{x}) = w$$

(2) *If some abstract $M$-algorithm $\alpha$ computes $f : M^n \to M$ and $f(\bar{x}) = w$, then there is a finite $U$ such that $U \vdash f(\bar{x}) = w$*

**Proof.** By the Homomorphism and Finiteness axioms.
Certificates

For coprimeness in the Euclidean algebra $\mathbb{N}_\varepsilon = (\mathbb{N}, 0, 1, \text{rem})$,

$$U = \{\text{rem}(x, y) = r_1, \text{rem}(y, r_1) = r_2, \ldots, \text{rem}(r_n, r_{n+1}) = 1\}$$

certifies that $x \perp y$.

To prove (by this method) that the Euclidean is number-of-calls optimal for coprimeness, we would need to show that every certificate for $x \perp y$ in $\mathbb{N}_\varepsilon$ is at least as large as $n + 2 = \text{calls}_\varepsilon(x, y)$ for infinitely many $(x, y)$. 
Complexity functions for computable functions

Suppose some abstract $M$-algorithm $\alpha$ computes $f : M^n \to M$.

- $\text{calls}_f(M, \vec{x}) = \min \{|\text{diag}(U)| : U \models f(\vec{x}) = w\} \leq \text{calls}_\alpha(\vec{x})$
- $\text{size}_f(M, \vec{x}) = \min \{|U \setminus \{0, 1, \vec{x}\}| : U \models f(\vec{x}) = w\} \leq \text{size}_\alpha(\vec{x})$
- $\text{depth}_f(M, \vec{x}) = \min \{\text{depth}_{\vec{x}}(U) : U \models f(\vec{x}) = w\} \leq \text{depth}_\alpha(\vec{x})$

**Thm** $\text{depth}_f(M, \vec{x}) \leq \text{size}_f(M, \vec{x}) \leq \text{calls}_f(M, \vec{x})$

**Interpretation:** $\text{depth}_f(M, \vec{x})$ is the least number of calls to the primitives which must be executed in sequence by any algorithm which computes $f : M^n \to M$ from the primitives of $M$.

- These complexities are most useful for relations $R \subseteq M^n$ whose characteristic functions take values in $\{0, 1\}$.
Outline of a proof

**Theorem** (van den Dries, ynm)

\[ M = (\mathbb{N}, 0, 1, \leq, +, \div, \text{iq, rem}) : \text{if } a \text{ is sufficiently large and } a^2 = 1 + 2b^2 \text{ or } a = F_{n+1}, b = F_n, \text{ then} \]

\[ \text{depth}_\perp (M, a, b) > \frac{1}{10} \log \log(a) \] (*)

So if \( \alpha \) decides coprimeness in \( M \), then (*) holds with \( \text{depth}_\alpha (a, b) \)

**Pf.** If \( 2^{2^4m+6} \leq a \), then every \( X \in G_m[M, a, b] \) can be written uniquely as

\[ X = \frac{x_0 + x_1 a + x_2 b}{x_3} \quad \text{with } x_i \in \mathbb{Z}, \ |x_i| \leq 2^{2^4m} \]

and we can define \( \pi : G_m[M, a, b] \rightarrow M \) letting \( \lambda = 1 + a! \),

\[ \pi(X) = \frac{x_0 + x_1 \lambda a + x_2 \lambda b}{x_3}, \text{ so } (\pi(a), \pi(b)) = (\lambda a, \lambda b) \]
The “universal constant” $\frac{1}{10}$

**Detailed version of result**

In $M = (\mathbb{N}, 0, 1, \leq, +, \cdot, \text{iq}, \text{rem})$: if $1 < \xi < 2$, $\xi$ is a quadratic algebraic irrational, $C > 0$, $a$ is sufficiently large, and $a \perp b$, then

$$\frac{1}{Cb^2} < \left| \frac{\xi - a}{b} \right| < \frac{1}{b^2} \implies \text{depth}_{\perp} (a, b) > \frac{1}{K} \log \log a,$$

with $K \geq 2 \log(\log C + 19)$

- Liouville: for sufficiently large $C$, infinitely many $a, b$ satisfy the hypothesis
- With $\xi = \sqrt{2}$ and $a^2 = 1 + 2b^2$, we can take $C = 5, K \geq 10$
- With $\xi = \frac{1}{2}(1 + \sqrt{5})$ and $a = F_{n+1}, b = F_n$, we can take again, $C = 5, K \geq 10$
\[ M = (\mathbb{N}, 0, 1, \text{Parity}, \text{iq}_2, \leq, +, \div, \text{Presburger functions}) \]

- (van den Dries, ynm) If \( R(x) \) is one of the relations
  
  \( x \text{ is prime}, \ x \text{ is a perfect square}, \ x \text{ is square free}, \)

  then for some \( r > 0 \) and infinitely many \( a \),

  \[ \text{depth}_R(M, a) > r \log(a) \]

- (van den Dries, ynm) For some \( r > 0 \) and infinitely many \( a, b \),

  \[ \text{depth}_\bot(M, a, b) > r \log(\max(a, b)) \]

- (Joe Busch) If \( R(x, p) \iff x \text{ is a square mod } p \),

  then for some \( r > 0 \) and a sequence \((a_n, p_n)\) with \( p_n \to \infty \),

  \[ \text{depth}_R(M, a_n, p_n) > r \log(p_n) \]

In the last two examples, the results match up to a multiplicative constant well-known known binary algorithms, so these are optimal
Non-uniform complexity

What if you are only interested in deciding $R(\vec{x})$ for $n$-bit numbers ($< 2^n$) and you are willing to use a different algorithm for each $n$?

► **The lookup algorithm:** For any $k$-ary relation $R$ on $\mathbb{N}$ and each $n$, there is an $\mathbb{N}_b$-term (with conditionals) $t_n(\vec{v})$ of depth $\leq n = \log(2^n)$ which decides $R(\vec{x})$ for all $\vec{x} < 2^n$.

► Non-uniform lower bounds on depth are never greater than $\log$.

► The best ones establish the optimality of the lookup algorithm and are most interesting when some uniform algorithm matches the lookup algorithm up to a multiplicative constant.

► They are quite easy for Presburger primitives.
Coprimesness from division, non-uniformly

**Theorem** (van den Dries, ynm)

Let $M = (\mathbb{N}, 0, 1, \leq, +, -, \mod, \rem)$ and for each $n$, let

$$x \mod_n y \iff x, y < 2^n \& \gcd(x, y) = 1.$$ 

There is some $r > 0$, such that for all sufficiently large $n$, there are $a, b < 2^n$ such that

$$\text{calls}_n(M, a, b) \geq \text{size}_n(M, a, b) > r \log n \quad (**)$$

So if $n$ is large enough and $\alpha$ decides coprimeness in $M$ for all $x, y < 2^n$, then (**) holds with $\text{calls}_\alpha(a, b), \text{size}_\alpha(a, b)$ on the left.

- I do not know how to get the corresponding result for $\text{depth}_n$.
Concluding remarks

(1) A technique for deriving lower bounds for decision problems which are absolute, i.e., they hold of all computational models

(2) Main limitation: in its current version, it only yields lower bounds which are no better than $O(n)$ (linear in the length of the input)

(3) Problem: prove that the Euclidean algorithm is optimal for computing the gcd in the algebra $\mathbb{N}_\varepsilon = (\mathbb{N}, 0, 1, \text{rem})$

(4) Problem: prove an $O(n^2)$ lower bound for primality in $\mathbb{N}_b = (\mathbb{N}, 0, 1, \text{Parity}, \text{iq}_2, (x \mapsto 2x), (x \mapsto 2x + 1))$

Comment: (4) may need some number theory, but it will also need some logical analysis of computation (since the entire input is known in $O(n)$ steps)