The axiomatic derivation of absolute lower bounds

Yiannis N. Moschovakis
UCLA and University of Athens

Tarski Lecture 3, March 7, 2008

A lower bound result

Theorem (van den Dries, ynm)

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

$$c^s_\alpha(a,b) > \frac{1}{10} \log\log(\max(a,b)) \qquad (*)$$

In fact (*) holds for all solutions of Pell's equation, $a^2 = 1 + 2b^2$

- $\mathrm{iq}(x,y), \mathrm{rem}(x,y)$ are the integer quotient and remainder
- $c^s_\alpha(x,y)$ counts the number of applications of the primitives in the computation
- Claim: This applies to all algorithms from the specified primitives
- The Euclidean decides coprimeness from rem with complexity

$$c^s_\epsilon(a,b) \leq 2\log(\min(a,b)) \quad (\min(a,b) \geq 2)$$

Outline of Lecture 3

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

. . . and so they should be (to some extent) a matter of logic

(1) Tweak logic (a bit) so it applies smoothly to computation theory
(2) Three (simple) axioms for elementary algorithms,
a la *abstract model theory*
(3) Lower bounds from the axioms
(4) Lower bounds for elementary algorithms on logical extensions

Is the Euclidean algorithm optimal among its peers? (with vDD, 2004)
Arithmetic complexity (with vDD, to appear)

Partial algebras, embeddings and subalgebras

- A (Partial, pointed) algebra is a structure $\mathbf{M} = (M, 0, 1, \Phi^{\mathbf{M}})$ where $0, 1 \in M$, Φ is a set of function symbols (the vocabulary) and $\Phi^{\mathbf{M}} = \{\phi^{\mathbf{M}}\}_{\phi \in \Phi}$, with $\phi^{\mathbf{M}} : M^{n_\phi} \rightharpoonup M$ for each $\phi \in \Phi$

- An embedding $\iota : \mathbf{U} \rightarrowtail \mathbf{M}$ from one Φ-algebra into another is any injection $\iota : U \rightarrowtail M$ such that

$$\iota(0^{\mathbf{U}}) = 0^{\mathbf{M}}, \quad \iota(1^{\mathbf{U}}) = 1^{\mathbf{M}},$$

and for all $\phi \in \Phi, x_1, \ldots, x_n, w \in U$,

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

- $\mathbf{U} \subseteq_p \mathbf{M}$ if the identity $I : U \rightarrowtail M$ is an embedding

Algebra restrictions

$\mathbf{N}_\varepsilon = (\mathbb{N}, 0, 1, \mathrm{rem})$, the Euclidean algebra
$\mathbf{N}_u = (\mathbb{N}, 0, 1, S, \mathrm{Pd})$, the *unary numbers*
$\mathbf{N}_b = (\mathbb{N}, 0, 1, \mathrm{Parity}, \mathrm{iq}_2, (x \mapsto 2x), (x \mapsto 2x+1))$, the *binary numbers*

For $\mathbf{M} = (M, 0, 1, \Phi^{\mathbf{M}})$ and $\{0,1\} \subseteq U \subseteq M$, let

$$\mathbf{M} \restriction U = (U, 0, 1, \Phi^{\mathbf{U}}),$$

where for $\phi \in \Phi$,

$$\phi^{\mathbf{U}}(\vec{x}) = w \iff \vec{x}, w \in U \;\&\; \phi^{\mathbf{M}}(\vec{x}) = w$$

For finite $U \subset \mathbb{N}$, $\mathbf{N}_u \restriction U$ is a finite, properly partial subalgebra of $\mathbf{N}$

Subalgebras generated from the input, $\mathbf{M} = (M, 0, 1, \Phi^{\mathbf{M}})$

For $\vec{x} = x_1, \ldots, x_n \in M$, set

$$G_0(\vec{x}) = \{0, 1, x_1, \ldots, x_n\}$$
$$G_{m+1}(\vec{x}) = G_m(\vec{x}) \cup \{\phi^{\mathbf{M}}(\vec{u}) \mid \phi \in \Phi, \vec{u} \in G_m(\vec{x}) \text{ and } \phi^{\mathbf{M}}(\vec{u})\downarrow\}$$

so that

$$G_m(\vec{x}) = \{t^{\mathbf{M}}[x_1, \ldots, x_n] \in M \mid t(v_1, \ldots, v_n) \text{ is a term of depth} \leq m\}$$

$$\boxed{\mathbf{M} \restriction G_m(\vec{x}) \text{ is the subalgebra of depth } m \text{ generated by } \vec{x}}$$

($\mathbf{M} \restriction \bigcup_m G_m(\vec{x})$ is the subalgebra generated by $\vec{x}$)

I The Locality Axiom

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

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

- **M**-algorithms "compute" strict partial functions, and they can be localized (relativized) to arbitrary subalgebras of **M**

We write

$$\boxed{\mathbf{U} \models \overline{\alpha}(\vec{x}) = w \iff \vec{x} \in U^n, w \in U \text{ and } \overline{\alpha}^{\mathbf{U}}(\vec{x}) = w}$$

II The Embedding Axiom

If α *is an n-ary algorithm of* $\mathbf{M}$, $\mathbf{U}, \mathbf{V} \subseteq_p \mathbf{M}$, *and* $\iota : \mathbf{U} \rightarrowtail \mathbf{V}$ *is an embedding, then*

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

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

- An algorithm treats the primitives of $\mathbf{M}$ as *oracles*: it can request values $\phi^{\mathbf{M}}(\vec{y})$, and use them if they are provided

III The Finiteness Axiom

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

$$\mathbf{M} \models \overline{\alpha}(\vec{x}) = w \implies \textit{there is an } m \textit{ such that } \vec{x}, w \in G_m(\vec{x})$$
$$\text{and } \mathbf{M} \restriction G_m(\vec{x}) \models \overline{\alpha}(\vec{x}) = w$$

In particular,

$$\overline{\alpha}^{\mathbf{M}}(\vec{x})\downarrow \implies \overline{\alpha}(\vec{x}) \in \bigcup_m G_m(\vec{x})$$

- "The computation" of $\overline{\alpha}^{\mathbf{M}}(\vec{x})$ takes place within the subalgebra of **M** generated by the input, and it is finite: take m large enough so that every y used in "the computation" is in $G_m(\vec{x})$

All algorithms—really—satisfy these axioms

- Explicit computation: $\overline{\alpha}^{\mathbf{U}}(\vec{x}) = t^{\mathbf{U}}[\vec{x}]$, where $t(\vec{v})$ is a Φ-term
- $\overline{\alpha}^{\mathbf{U}}$ is the partial function computed a fixed recursive (McCarthy) program A in the signature Φ (as in Lecture 1)
- $\overline{\alpha}^{\mathbf{U}}$ is computed by a register machine (or RAM, or Turing machine or ...) from $\Phi^{\mathbf{U}}$
- $\overline{\alpha}^{\mathbf{U}}$ is computed in Plotkin's PCF above the algebra $\mathbf{U}$
- $\overline{\alpha}^{\mathbf{U}}$ by computed in non-deterministic versions of any of these

Axioms for elementary algorithms

- I, Locality Axiom: An algorithm α of arity n of an algebra $\mathbf{M} = (M, 0, 1, \Phi^{\mathbf{M}})$ assigns to each subalgebra $\mathbf{U} \subseteq_p \mathbf{M}$ an n-ary, strict partial function

$$\overline{\alpha}^{\mathbf{U}} : U^n \rightharpoonup U \qquad (\mathbf{U} \models \overline{\alpha}(\vec{x}) = w \iff \overline{\alpha}^{\mathbf{U}}(\vec{x}) = w)$$

- II, Embedding Axiom: If $\mathbf{U}, \mathbf{V} \subseteq_p \mathbf{M}$, and $\iota : \mathbf{U} \rightarrowtail \mathbf{V}$ is an embedding, then

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

- III, Finiteness Axiom:

$$\mathbf{M} \models \overline{\alpha}(\vec{x}) = w \implies \textit{there is an } m \textit{ such that } \vec{x}, w \in G_m(\vec{x})$$
$$\text{and } \mathbf{M} \restriction G_m(\vec{x}) \models \overline{\alpha}(\vec{x}) = w$$

The embedding complexity of an algorithm

If α is an algorithm of $\mathbf{M}$ and $\mathbf{M} \models \overline{\alpha}(\vec{x}) = w$, set

$$\boxed{c^{\iota}_{\alpha}(\vec{x}) = \text{the least } m \text{ such that } \mathbf{M} \restriction G_m(\vec{x}) \models \overline{\alpha}(\vec{x}) = w}$$

This is defined by the Finiteness Axiom

- Intuitively, if $m = c^{\iota}_{\alpha}(\vec{x})$, then any implementation of α will need to "consider" (use) some $u \in M$ of depth m; and so it will need at least m steps to construct this u from the input using the primitives
- If $\overline{\alpha}(\vec{x}) = t^{\mathbf{M}}[\vec{x}]$, then $c^{\iota}_{\alpha}(\vec{x}) \leq \text{depth}(t(\vec{v}))$
- c^{ι}_{α} is majorized by all usual time-complexity measures, including the number of calls to the primitives

The embedding complexity of a (computable) function

Fix $f : M^n \to M$. An embedding $\iota : \mathbf{M} \restriction G_m(\vec{x}) \rightarrowtail \mathbf{M}$ respects f at $\vec{x}$ if

$$f(\vec{x}) \in G_m(\vec{x}) \ \& \ \iota(f(\vec{x})) = f(\iota(\vec{x}))$$

Lemma

If some algorithm computes f in $\mathbf{M}$, then for each $\vec{x}$, there is some m such that every embedding $\iota : \mathbf{M} \restriction G_m(\vec{x}) \rightarrowtail \mathbf{M}$ respects f at $\vec{x}$

Proof Take $m = c^{\iota}_{\alpha}(\vec{x})$ for some α such that $f = \overline{\alpha}^{\mathbf{M}}$

$$\boxed{c^{\iota}_{f}(\vec{x}) = \text{the least } m \text{ such that every } \iota : \mathbf{M} \restriction G_m(\vec{x}) \rightarrowtail \mathbf{M} \text{ respects } f \text{ at } \vec{x}}$$

If α computes f in $\mathbf{M}$, then $c^{\iota}_{f}(\vec{x}) \leq c^{\iota}_{\alpha}(\vec{x})$

- To show that m is an absolute lower bound for the computation of $f(\vec{x})$ show that $\boxed{f(\vec{x}) \notin G_m(\vec{x})}$, or construct $\iota : \mathbf{M} \restriction G_m(\vec{x}) \rightarrowtail \mathbf{M}$ such that $\boxed{\iota f(\vec{x}) \neq f(\iota \vec{x})}$

Outline of a proof

Theorem (van den Dries, ynm)

For the algebra $\mathbf{M} = (\mathbb{N}, 0, 1, \leq, +, \dot{-}, \mathrm{iq}, \mathrm{rem})$ *and the relation of coprimeness* $x \perp\!\!\!\perp y$*,*

$$a^2 = 1 + 2b^2 \implies c^{\iota}_{\perp\!\!\!\perp}(a, b) > \frac{1}{10} \log\log(a) \qquad (*)$$

So if α *decides coprimeness in* $\mathbf{M}$*, then* $(*)$ *holds with* $c^{\iota}_{\alpha}(a, b)$

- If $2^{2^{4m+6}} \leq a$, then every $X \in G_m(a, b)$ can be written uniquely as

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

  and we can define $\iota : \mathbf{M} \restriction G_m(a, b) \rightarrowtail \mathbf{M}$ using $\lambda = 1 + a!$,

$$\iota(X) = \frac{x_0 + x_1 \lambda a + x_2 \lambda b}{x_3}, \text{ so } (\iota(a), \iota(b)) = (\lambda a, \lambda b)$$

$\mathbf{M} = (\mathbb{N}, 0, 1, \mathrm{Parity}, \mathrm{iq}_2, \leq, +, \dot{-}, \text{Presburger functions})$

- (van den Dries, ynm) *If $R(x)$ is one of the relations*

  $$x \textit{ is prime}, \quad x \textit{ is a perfect square}, \quad x \textit{ is square free},$$

  then for some $r > 0$ and infinitely many a, $c^{\iota}_R(a) > r\log(a)$

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

  $$c^{\iota}_{\perp\!\!\!\perp}(a, b) > r\log(\max(a, b))$$

- (Joe Busch) *If $R(x, p) \iff x$ is a square $\bmod p$,*
  then for some $r > 0$ and a sequence (a_n, p_n) with $p_n \to \infty$,

  $$c^{\iota}_R(a_n, p_n) > r\log(p_n)$$

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

Primality in $\mathbf{M} = (\mathbb{N}, 0, 1, \mathrm{Parity}, \mathrm{iq}_2, \leq, +, \dot{-}, \mathrm{Presburger})$

Theorem (van den Dries, ynm)

If $\mathrm{Prime}(p) \iff p$ is prime, *then in* $\mathbf{M}$, *for some* $r > 0$ *and all primes* p,

$$c^{\iota}_{\mathrm{Prime}}(p) > r \log p \qquad (*)$$

So if α *decides primality in* $\mathbf{M}$, *then* (*) *holds with* $c^{\iota}_{\alpha}(p)$

- If $2^{2m+2} \leq a$, then every $X \in G_m(a)$ can be written uniquely as

$$X = \frac{x_0 + x_1 a}{2^m} \quad \text{with } |x_i| \leq 2^{2m},$$

and we can define $\iota : \mathbf{M} \upharpoonright G_m(a) \rightarrowtail \mathbf{M}$ by

$$\iota(X) = \frac{x_0 + x_1 \lambda a}{2^m}, \text{ with } \lambda = 1 + 2^m, \text{ so } \iota(a) = \lambda a$$

Primality in binary

- If $\mathrm{Prime}(p) \iff p$ is prime, then in

$$\mathbf{N}_b = (\mathbb{N}, 0, 1, \mathrm{Parity}, \mathrm{iq}_2, (x \mapsto 2x), (x \mapsto 2x+1))$$

for some $r > 0$ and all primes p,

$$c^{\iota}_{\mathrm{Prime}}(p) \geq r \log p \qquad (*)$$

- This should follow trivially from number-theoretic results, because it takes at least i applications of the primitives of $\mathbf{N}_b$ to read i bits of the input; we should have $r = 1$
- Theorem (Tao). *For infinitely many primes p, if p' is constructed by changing any bit in the binary expansion of p except the highest, then p' is not prime*
- Tao found subsequently that this result is implicit in a paper of Cohen and Selfridge from 1975 and explicitly noted in a 2000 paper by Sun, and he obtained more general results

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?

Theorem (The lookup algorithm)

For each k-ary relation R on $\mathbb{N}$ and each n, there is an $\mathbf{N}_b$-term (with conditionals) $t_n(\vec{v})$ *of depth* $\leq n = \log_2(2^n)$ *which decides* $R(\vec{x})$ *for all* $\vec{x} < 2^n$.

- Non-uniform lower bounds 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 up to a multiplicative constant)
- They are mostly about "the size" of $t(\vec{v})$
- They do not follow from Axiom I – III

Recursive (McCarthy) programs of $\mathbf{M} = (M, 0, 1, \Phi^{\mathbf{M}})$

Explicit Φ-terms (with p_i^n partial function variables)

$$A :\equiv 0 \mid 1 \mid \mathsf{v}_i \mid \phi(A_1, \ldots, A_n) \mid \mathsf{p}_i^n(A_1, \ldots, A_n) \\ \mid \text{if } (A_0 = 0) \text{ then } A_1 \text{ else } A_2,$$

Recursive program (only $\vec{x}_i, p_1, \ldots, p_K$ occur in each part A_i):

$$A \quad : \quad \begin{cases} \mathsf{p}_A(\vec{\mathsf{x}}_0) = A_0 \\ \mathsf{p}_1(\vec{\mathsf{x}}_1) = A_1 \\ \quad\vdots \\ \mathsf{p}_K(\vec{\mathsf{x}}_K) = A_K \end{cases} \quad (A_0: \text{ the head}, (A_1, \ldots, A_K): \text{ the body})$$

The elementary algorithms of $\mathbf{M}$ *are expressed by recursive programs*

(and they satisfy Axioms I – III)

A non-uniform lower bound result for elementary algorithms

If α is the algorithm expressed by a recursive program in $\mathbf{M}$, let

$$c^s_\alpha(\vec{x}) = \text{the number of calls to the primitives}$$
$$\text{made in the computation of } \overline{\alpha}(\vec{x}) \;\geq c^\iota_\alpha(\vec{x})$$

Theorem (van den Dries, ynm)

Let $\mathbf{M} = (\mathbb{N}, 0, 1, \leq, +, \dot{-}, \mathrm{iq}, \mathrm{rem})$. *There is some* $r > 0$, *such that for all sufficiently large* n *and every* $\mathbf{M}$*-elementary algorithm* α *which decides coprimeness for all* $x, y < 2^n$, *there exist* $a, b < 2^n$ *such that*

$$c^s_\alpha(a, b) > r \log_2 n \geq r \log_2 \log_2(\max(a, b))$$

The proof is by the embedding method, but uses special properties of recursive programs (the computation space)

Logical extensions (a la Tarski)

A $(\Phi \cup \Psi)$-algebra $\overline{\mathbf{M}}$ is a logical extension of a Φ-algebra $\mathbf{M}$ if

(1) $M \subseteq \overline{M}$, $0^{\mathbf{M}} = 0^{\overline{\mathbf{M}}}$, $1^{\mathbf{M}} = 1^{\overline{\mathbf{M}}}$

(2) For each $\phi \in \Phi$, $\phi^{\mathbf{M}} = \phi^{\overline{\mathbf{M}}}$

(3) Every bijection $\iota : M \rightarrowtail\!\!\!\!\twoheadrightarrow M$ which fixes $0, 1$ can be extended to a bijection $\bar{\iota} : \overline{M} \rightarrowtail\!\!\!\!\twoheadrightarrow \overline{M}$ such that for every $\psi \in \Psi$,

$$\psi^{\overline{\mathbf{M}}}(\bar{\iota}\vec{x}) = \bar{\iota}\psi^{\overline{\mathbf{M}}}(\vec{x}) \qquad (\vec{x} \in \overline{M}^n)$$

i.e., $\bar{\iota}$ is an automorphism of the reduct $(\overline{M}, 0, 1, \Psi^{\overline{\mathbf{M}}})$

- Random Access (and all other kinds of) Machines from $\Phi^{\mathbf{M}}$, Plotkin's PCF over $\mathbf{M}$, etc., are all faithfully represented by recursive programs on logical extensions of $\mathbf{M}$

The persistence of embedding complexity

Theorem (van den Dries, Neeman, ynm)
If $f : M^n \to M$ and $\overline{\mathbf{M}}$ is a logical extension of $\mathbf{M}$, then

$$c_f^{\iota}(\vec{x}, \mathbf{M}) = c_f^{\iota}(\vec{x}, \overline{\mathbf{M}}) \quad (\vec{x} \in M^n)$$

- This is why the embedding method gives the same lower bounds (for a function f from specified primitives) for RAMs and for recursive programs, even though the direct *simulation* of RAMs by recursive programs has an overhead
- The basic non-uniform results obtained by the embedding method also extend to arbitrary logical extensions

Back to sorting

Theorem

If $\leq$ is an ordering of a set A, $\overline{\mathbf{A}}$ is a logical extension of $(A \cup \{0, 1\}, 0, 1, \leq)$ such that $A^* \subseteq A$, and α is an elementary algorithm of $\overline{\mathbf{A}}$ which sorts the sequences in A^*, then

$$|u| = n \implies c^s_\alpha(u) \geq \log_2(n!) \sim n \log_2(n),$$

where $c^s_\alpha(u)$ is the number of comparisons made by α in the computation of $\mathrm{sort}(u)$

This is proved by the classical, counting argument