Superexponential computations in exponential domain

Extended abstract of a lecture at JAF, Seville 11.6.2007

Pavel Pudlák
Mathematical Institute, Academy of Sciences, Prague *

November 13, 2007

In this lecture we shall discuss the possibility of using superexponential computations for showing separations of certain complexity classes that are computable in polynomial time using an oracle for an NP set. Of course, if we only use polynomial space we shall loop after an exponential number of step, so superexponential computation cannot be done in an exponential domain. So our idea is to take a nonstandard model and connect computations in a superexponential domain with computation in an exponential domain.

Our motivation is to show that the hierarchy of $\forall\Sigma_1^b$ sentences provable in $T_2^n(\alpha)$ is strictly increasing for $n = 0, 1, \ldots$. Here we denote by $T_2^n(\alpha)$ the relativized versions of the theories T_2^n introduced by Buss [1].

Provable Σ_1^b sentences and total search problems

Suppose $T_2^n \vdash \forall x \exists y \leq t(x)\ \phi(x, y)$,

1. where we assume $\phi(x, y)$ is Σ_0^b, hence it defines a *polynomial time computable relation.* Then
2. $y \leq t(x)$ implies that *the length of y is bounded by a polynomial in the length of x;*
3. the sentence $\forall x \exists y \leq t(x)\ \phi(x, y)$ *is true.*

*Supported by grant A1019401

Conditions 1 - 3 define a *Total Search Problem* (TSP).

Example 1 *Let $f(x,y)$ be a polynomial time computable function and $Q(x,y)$ a polynomial time predicate such that for all x, y,*

$$f(x,y) \leq x,\ \ Q(x,0) \text{ and } \forall y(Q(x,y) \rightarrow Q(x, f(x,y))).$$

Then

$$\forall x \exists y (Q(x,y) \wedge f(x,y) \leq y)$$

determines a TSP.

The natural algorithm to find y is (with parameter x suppressed):

```
y:=0;
do y:=f(y) until f(y)≤ y.
```

The algorithm runs in exponential time and we do not expect that in general the problem can be solved in polynomial time. This problem belongs to the class of TSPs called *Polynomial Local Search*.

A TSP $\forall x \exists y\ \phi(x,y)$ is *polynomially reducible* to a TSP $\forall u \exists v\ \psi(u,v)$, if there exist polynomial time computable which for a given x computes a solution y using queries to the search problem $\forall u \exists v\ \psi(u,v)$, ie., the algorithm computes some u_i and obtains v_i such that $\psi(u_i, v_i)$ for polynomially many i.

Lemma 1 *Suppose*

$$T^0_2 \vdash \forall u \exists v \leq t(u)\ \psi(u,v) \rightarrow \forall x \exists y \leq s(x)\ \phi(x,y).$$

Then TSP ϕ is reducible to TSP ψ (using a constant number of queries).[1]

Corollary 2 *In order to show that T^{n+1}_2 proves more Σ^b_1 sentences than T^n_2, it suffices*

1. *to find characterizations σ_n and σ_{n+1} for T^n_2 and T^{n+1}_2;*

[1] Here I am using T^0_2 extended by the function MSP as proposed by Jeřábek [3], which is equivalent to PV.

2. *to show that the TSP σ_{n+1} is not reducible to the TSP σ_n (purely computational problem!).*

Characterizing Σ_1^b sentences of T means to find a Σ_1^b sentence σ such that (1) $T \vdash \sigma$, (2) for all Σ_1^b sentences ξ, if $T \vdash \xi$, then $T_2^0 \vdash \sigma \to \xi$. The state of art in characterizing Σ_1^b sentences of T_2^k is:

- A characterization of provable Σ_1^b sentences is known for all T_2^k, but the sentences $\mathbf{A_k}$ are rather complicated [**?**, 8].
- "Nice" characterizations has been known for $k = 0, 1, 2$ and 3, [2, 5].
- Quite recently Thapen found simpler characterizations for all n [**?**].
- However, separations of the relativized theories $T_2^k(\alpha)$ using $\Sigma_1^b(\alpha)$ sentences are known only for $k = 0$ and 1, [**?**, **?**]

We want to pursue an idea based on the following considerations.

1. Search problems often come with "natural" algorithms.
2. We can classify them by the running time of these algorithms.
3. We might get oracle separations based on different running times of the "natural algorithms".

While we can give examples of 1. and 2., it is a challenge to realize 3. We have only partial results in that direction.

An example of a problem whose "natural algorithm" needs larger time than the size of the search space.

The idea is to define a sort of k-times iterated PLS. We want to solve the PLS problem 1, but in order to compute the neighborhood function for the PLS problem i, $i < k$, we have to solve an instance of the problem $i + 1$.

For an input a we shall assume that all numbers are $\leq a$. Parameter k is a constant (potentially it can also be a variable).

1. Functions and predicates:

- $f_1(x,y,z),\ldots,f_{k-1}(x,y,z),f_k(x,y)$ — functions
- $Q_1(x,y),\ldots,Q_k(x,y)$ — predicates

2. Clauses

   - $Q_1(x,0),\ldots,Q_k(x,0)$
   - for $1 \le i \le k-2$

   $$(Q_i(u,x) \wedge Q_{i+1}(x,y) \wedge f_{i+1}(x,y,z) \le y) \to \ Q_i(u, f_i(u,x,y))$$

   - for $k-1$

   $$(Q_{k-1}(u,x) \wedge Q_k(x,y) \wedge f_k(x,y) \le y) \to \ Q_i(u, f_{k-1}(u,x,y))$$

   - for k

   $$Q_k(x,y) \to \ Q_k(x, f_k(x,y))$$

3. The task is, for a given input a, either to find y,z such that $Q_1(a,y) \wedge f_1(a,y,z) \le y$, or find a clause that is violated.

The natural algorithm to solve this problem runs in time a^k, but we can actually solve it in time a^4, since each clause has at most 4 variables.

An interpretation of naturalness

Let

$$\forall x \exists y \le t(x)\ \phi(x,y)$$

be a TSP, let $n = |x|$ be the input size. Then $|y| \le n^c$. Now suppose the sentence $\forall x \exists y\ \phi(x,y)$ remains true when we omit the bound on y. Then the running time for finding y may be arbitrarily large. The growth rate of this function can give us some hint about how difficult the original search problem is.

In particular this happens in case of *relativized* versions of TSPs when the formula $\phi(x,y)$ can be viewed as a *first order* formula.

Let us consider Example 1 in which we introduce a *cost function* c and *do not restrict the search space.*

Example 2 *Suppose for all y,*

$$c(f(y)) \leq x, \ Q(0) \ \textit{and} \ \forall y(Q(y) \rightarrow Q(f(y))).$$

Then

$$\exists y(Q(y) \wedge c(f(y)) \leq c(y)),$$

Observe that the cost function ensures the existence of a solution even if the domain of the function f is infinite.

Suppose we modify the problem from the foregoing section by

1. letting the parameter range an arbitrary domain D (instead of the interval $[0, a]$),
2. introducing a valuation $\nu : D \rightarrow [0, a]$,
3. replacing the inequalities by $\nu(f(x, y, z)) \leq \nu(y)$.

Then, assuming that the functions and predicates are oracles, the search problem requires a^k steps to be solved, whereas the original problem only needs time a^4.

Principles $\mathbf{A}_k$

Sentences $\mathbf{A}_k$ were introduced in [**?**]. We have shown that they characterizes Σ_1^b theorems of T_2^k [**?**, 8]. Moreover we showed that *a natural algorithm for solving the TSP $\mathbf{A}_k$ runs in time $\approx 2_{ck}^n$,* where 2_r^n denotes r-times iterated exponential function, and $c = \Theta(1)$, This gives a classification of $\mathbf{A}_k$, and supports the conjecture that they form a properly increasing hierarchy.

Principles $\mathbf{A}_k$ are based on finite games. Their definition is rather complicated, so we refer the reader to aforementioned papers.

Nonstandard models of arithmetic instead of the adversary method

Here we want to suggest that nonstandard models may be useful for constructing oracles needed for relativized separation.

Let $\mathbf{M}$ be a countable nonstandard model of True Arithmetic. Let $n \in M$ be nonstandard. Suppose $\mathbf{B}$ is an algebraic structure definable in $\mathbf{M}$ (internal) such that its support $|B| \geq 2^n$ ("B" stands for "big").

Lemma 3 *There exists an external bijection $\iota : B \to \{0,1\}^n$ which preserves polynomial time computations. Specifically, if P is a straight-line program of length n^c, for $c \in \omega$, and $b_1, \ldots, b_t$ is a computation of P on some input elements of $\mathbf{B}$, then ι restricted to $\{b_1, \ldots, b_t\}$ is an internal set in $\mathbf{M}$.*

This lemma is trivial. It would be more interesting to find such mappings with more specific properties.

In order to apply the lemma we need to relax the assumptions. Say that $\mathbf{B}$ is *almost internal*, if it is a subalgebra of an internal algebra closed under operations defined by polynomial size straight-line programs. Thus we can prove:

Lemma 4 *If $k << l$, then there exists an algebraic structure $\mathbf{B}$ larger than 2^n such that*

- *every polynomially $\mathbf{B}$-definable instance of $\mathbf{A}_k$ holds true in $\mathbf{B}$;*
- *$\mathbf{A}_l$ fails in $\mathbf{B}$.*

In a *polynomially* $\mathbf{B}$*-definable instance* of $\mathbf{A}_k$ functions are replaced by polynomial size straight-line programs and predicates by polynomial size formulas computed in the algebra $\mathbf{B}$.

This lemma enables us to prove that $\mathbf{A}_l$ cannot be reduced to $\mathbf{A}_k$ at least in case when only a special kind of reductions is used.

Corollary 5 *There exist an oracle with respect to which $\mathbf{A}_l$ is not polynomially reducible to $\mathbf{A}_k$* using polynomially definable reductions.

Proof. Consider the image $\iota(\mathbf{B})$ of $\mathbf{B}$ on $\{0,1\}^n$. Then instances of $\mathbf{A}_k$ are true in this structure, but no instance of $\mathbf{A}_l$ is. In this external structure we can still interpret every polynomial computations that uses only the images of the functions of $\mathbf{B}$.

Let r be a straight-line program in the language of $\mathbf{B}$. Suppose it is a reduction of $\mathbf{A}_l$ to a polynomially definable version of $\mathbf{A}_k$. Given $x \in \{0,1\}$ we can run $r(x)$. Since $\mathbf{A}_k$ is true, we should find a solution for $\mathbf{A}_l$. But $\mathbf{A}_l$ and is false in $\iota(\mathbf{B})$.

This gives an adversary strategy against the particular reduction. Thus we have an adversary strategy for every reduction. This implies the existence of an oracle. ∎

To get the above result without any restriction, we need to include boolean operation into computations. But the pull-back of the boolean structure on B is not internal, so we cannot ensure $\mathbf{A}_k$ to hold true. A possible solution could be to replace *external bijections* $\iota : \mathbf{B} \to \{0,1\}^n$, by *internal hash functions.* (ι is a hash function, if no polynomial time computation can find a conflict, ie., a pair $x \neq y$ such that $\iota(x) = \iota(y)$.)

Conclusions and further research

Presently the set of sentences $\mathbf{A}_k$ is the only set to which we have been able to associate a hierarchy of functions. So it is not clear whether is just an incident, or an instance of a general principle. If it were the latter, it would open the possibility of using new means in Bounded Arithmetic, means similar to those used in the classical proof theory (ordinal analysis).

References

[1] S.R. Buss, *Bounded Arithmetic,* Bibliopolis, 1986

[2] S.R. Buss and J. Krajíček, An application of boolean complexity to separation problems in bounded arithmetic, *Proc. of the London Math. Soc.* 69(3), 1994, pp.1-21.

[3] E. Jeřábek, *The strength of sharply bounded induction,* Mathematical Logic Quarterly 52 (2006), no. 6, pp. 613-624.

[4] J. Krajíček, No counter-example interpretation and interactive computations, in *Logic from Computer Science,* ed. Y.N. Moschovakis, Springer-Verlag 1992, pp.287-293.

[5] J. Krajíček, A. Skelley, and N. Thapen, *NP search problems in low fragments of bounded arithmetic,* to appear in the Journal of Symbolic Logic.

[6] P. Pudlák, Some relations between subsystems of arithmetic and the complexity of computations, in *Logic from Computer Science,* ed. Y.N. Moschovakis, Springer-Verlag 1992, pp.308-317.

[7] P. Pudlák, *Consistency and games - in search of new combinatorial principles.* Proc. Logic Colloquium'03, Helsinki, eds. V. Stoltenberg-Hansen and J. Vaananen, Assoc. for Symbolic Logic 2006, pp.244-281

[8] P. Pudlák: *Fragments of Bounded Arithmetic and the lengths of proofs,* (in preparation).