CSG399: Gems of Theoretical Computer Science. Lecture 28. Apr. 21, 2009.
Instructor: Emanuele Viola Scribe: Emanuele Viola

# Natural Proofs

The central challenge of computational complexity is to prove lower bounds, i.e. exhibiting explicit functions that cannot be computed with limited resources. In this lecture we discuss the *Natural Proofs* result by Razborov and Rudich which shows that some of the known techniques for lower bounds fall in a class of techniques which, under well-known assumptions, cannot prove the strong, desired lower bounds such as that NP cannot be computed by polynomial-size circuits. In some cases, for example to establish an exponential lower bound for the discrete-log function, one needs no assumption but can prove unconditionally that the class of techniques cannot prove such bounds.

Informally, to show that some function $f : \{0,1\}^n \to \{0,1\}$ cannot be computed with limited resources (e.g., by small circuits), most lower bounds proceed by exhibiting some property $P(f)$ of boolean functions such that:

1. $P$ holds for functions computable with limited resources, and
2. $P$ does not hold for $f$.

For example, when we showed that parity cannot be computed by small constant-depth circuits, the property $P(f)$ was "$f$ is approximable by a low-degree polynomial." For the communication lower bound, $P$ was "$R(f)$ is close to 1."

As it turns out, many lower bound proofs actually show more and give a property $P$ that satisfies:

I. (1, unchanged) $P$ holds for functions computable with limited resources, and

II. $P$ does not hold for $2^{-cn}$ fraction of $n$-bit functions (i.e., a noticeable fraction of functions), and

III. $P$ is efficiently computable: Given a truth-table of length $2^n$ of a function $f : \{0,1\}^n \to \{0,1\}$, $P(f) \in \{0,1\}$ can be computed by a circuit of size $\leq 2^{cn}$ (i.e., polynomial in the input length).

A proof that yields a property $P$ satisfying the three conditions above is called *natural.* In the communication lower bound the quantity $R$ is indeed efficiently computable (if $k$ is not too large), and the same is true for many other properties in the literature. (Warning: as far as I know, it has not been pointed out whether the property we used for the lower bound for parity is efficiently computable, though related properties are.)

The idea is that such a proof will not work for models like polynomial-size circuits because the associated property $P$ could be used to distinguish random functions (i.e., a random truth table of length $2^n$) from functions $f : \{0,1\}^n \to \{0,1\}$ computable in the model. But this is known to be impossible under well-known assumptions.

**Theorem 1.** *Assume for every $k$ there is a function $f : \{0,1\}^k \to \{0,1\}^k$ that is one-way with the following parameters:*

- *$f$ is computable by circuits of size* $\text{poly}(k)$*,*
- *$f$ is $2^{k^{\Omega(1)}}$-hard to invert.*

*Then, interpreting "limited resources" in (I) with "*$\text{poly}(n)$*-size circuits," we have that (I) + (II) + (III) is impossible.*

As we discussed, a candidate function for the hypothesis of the theorem is basically integer multiplication (under the assumption that factoring integers is sufficiently hard).

*Proof sketch.* Set $k := n^d$ for $d$ to be chosen later. From $f$, we construct a distribution $C_a : \{0,1\}^n \to \{0,1\}$ such that

- For every $a$, $C_a$ is computable by circuits of size $\text{poly}(n)$, and
- there is $\epsilon > 0$ (independent from $d$) such that any circuit $D$ of size $2^{k^\epsilon}$ is fooled by $C_a$:

$$\left| \Pr_a[D(C_a(0)C_a(1)\ldots C_a(2^n-1)) = 1] - \Pr_U[D(U) = 1] \right| \leq 2^{-k^\epsilon},$$

  where $U$ is the uniform distribution over truth-tables of length $2^n$.

But this yields a contradiction as follows: $P$ is computable by a circuit of size $2^{cn}$ (by III) and we have

$$\left| \Pr_a[P(C_a(0)C_a(1)\ldots C_a(2^n-1)) = 1] - \Pr_U[P(U) = 1] \right| \geq 1 - (1 - 2^{cn}) \geq 2^{cn},$$

where we use (I) and (II). This is a contradiction for $d = 2/\epsilon$. □

How do we construct $C_a$? The generic construction has two steps. First, we construct a length-doubling pseudorandom generator $G : \{0,1\}^\ell \to \{0,1\}^{2\ell}$ where $\ell = \text{poly}(k)$, then we use a tree construction to obtain $C_a$. The tree is binary; each node is an application of $G$ whose input is half the output of the parent. The root is fed with $a$. The input to $C_a$ specifies a path in the tree and the output is, say, the first bit of the leaf we reach.

This construction has large depth and is not usable for constant-depth circuits. But Naor and Reingold showed how, under more specific assumptions (the hardness of factoring is among them), a suitable $C_a$ can be computed by unbounded fan-in depth-5 circuits with majority gates. So the natural proofs result already applies to this seemingly restricted computational model.