03684155: On the P vs. BPP problem. 18/12/16 – Lecture 10

Natural proofs

Amnon Ta-Shma and Dean Doron

1 Natural proofs

The ultimate goal we have is separating classes (or proving they are equal if they are). But our success so far seems to be limited: We know a (quite) tight time hierarchy, a tight space hierarchy, non-deterministic hierarchies, and some other (weaker) hierarchies. However, we are unable to even prove that $\mathsf{LOG} \neq \mathsf{P}^{\#\mathsf{P}}$ or that $\mathsf{BPP} \neq \mathsf{NEXP}$. In this lecture we seek some justification to this state of affairs.

First, there is this issue that often hierarchies are built using simulation (and negation) and then almost by definition these hierarchies are not only true, but also true with respect to any oracle. For example, diagonalization not only shows $\mathsf{DTIME}(n) \neq \mathsf{DTIME}(n^2)$ but also that for any oracle O, $\mathsf{DTIME}(n)^O \neq \mathsf{DTIME}(n^2)^O$. Such results are called *relativized*, i.e., they remain true relative to any possible oracle. Since there are oracles O_1 and O_2 such that $\mathsf{P}^{O_1} = \mathsf{NP}^{O_1}$ and $\mathsf{P}^{O_2} \neq \mathsf{NP}^{O_2}$ the P vs. NP question does not relativize and cannot be solved that way. In fact it seems the borderline between what we know and what we don't roughly corresponds to what relativizes and what does not. However, there are exceptions.

This observation (that diagonalization usually relativize and does not suffice for many interesting problems) is due to Baker, Gill and Solovay [1].

We now shift attention to the problem of proving lower bounds against non-uniform computation. For example we may ponder why is it so hard to find an explicit function that is hard for polynomial-size circuits (i.e., is not in $\mathsf{P/poly}$). We discussed a lot in class the implications of $\mathsf{EXP} \subseteq \mathsf{P/poly}$ and $\mathsf{NEXP} \subseteq \mathsf{P/poly}$ (a collapse to MA) and $\mathsf{EXP} \not\subseteq \mathsf{P/poly}$ (a partial derandomization of BPP). As we know, it is still not known whether $\mathsf{NEXP} \subseteq \mathsf{P/poly}$ or not. Why is it so difficult to prove such lower-bounds?

In this lecture we define what *natural* proofs are, and show that if we believe in cryptography natural proofs do not exist for many interesting separations we believe in. The results are from Razborov and Rudich [3].

Say we have a non-uniform complexity class C, for example, you might think of $\mathsf{SIZE}(s(n))$, $\mathsf{P/poly}$ or AC^0. For every n there are those function $f \in F_n = \{f : \{0,1\}^n \to \{0,1\}\}$ that belong to the class (e.g., have a small circuit, or a small constant-depth circuit) and those that do not. Let $C_n = C \cap F_n$. For example, if $C = \mathsf{SIZE}(s(n))$ then $f \in C_n$ if f is a function on n bits that can be computed by $s(n)$ size circuits.

Our goal is to find a function (in fact, a sequence of functions) that is not in C. Razborov and Rudich call a proof *natural* if we do that by finding a property P such that:

1. Many functions (equivalently, a random function) have this property,
2. All functions in C do not have this property, and,

3. Given a truth table one can determine in polynomial time whether it satisfies the property P or not.

It may be argued if indeed natural proofs really need to obey the above conditions, but it is true that most lower bound (or at least many) are natural. Be it natural or not, we continue with the following definitions:

Definition 1. *$F_n = \{f : \{0,1\}^n \to \{0,1\}\}$. We sometimes view $f \in F_n$ as a boolean function on n variables, and sometimes as $f \in \{0,1\}^{2^n}$.*

Definition 2 (Non-uniform class of functions)**.** *A non-uniform class $\mathcal{C}$ is a collection $\{\mathcal{C}_n \subseteq F_n\}_n$, where we think of $\mathcal{C}_n$ as the set of all functions on n variables that are within the class (e.g., have a small circuit).*

$\mathsf{SIZE}(s(n))$ and depth-4 circuits of size n^{10} are examples for such non-uniform classes of functions. $\mathsf{P/poly}$ or AC^0 are slightly different, because the bound is asymptotic (and for every fixed n it is possible we may take 2^n size). However, they also fall into this frame of reasoning and for simplicity we continue with this definition. In fact, instead of proving a bound for $\mathsf{P/poly}$ you can assume we try to prove a lower bound on, say, for $n^{\log\log\log n}$.

Definition 3 (Property)**.** *A property $\mathcal{P}$ is a collection $\{\mathcal{P}_n \subseteq F_n\}_n$, where we think of $\mathcal{P}_n$ as the set of all functions on n variables that have the property (e.g., they cannot be approximated by a low-degree polynomial).*

Definition 4 (Constructive property)**.** *Let Γ be a complexity class (e.g., P or NC^2). We say a property $\mathcal{P}$ is* Γ-constructive *if there exists an algorithm in Γ that given the truth table of a function in $f \in F_n$, given by its truth table of length 2^n, determines whether $f \in \mathcal{P}_n$.*

For example if $\Gamma = \mathsf{P}$ then it is possible to determine in polynomial time (in 2^n) membership in $\mathcal{P}_n$.

Definition 5 (Large property)**.** *We say a property $\mathcal{P}$ is* large *if for every n,*

$$\Pr_{f \in F_n}[f \in \mathcal{P}_n] \;=\; \Pr_{f \in \{0,1\}^{n'}}[f \in \mathcal{P}_n] \;\geq\; \frac{1}{\mathrm{poly}(n')}$$

where $n' = 2^n$. I.e., a non-negligible fraction of all functions belong to the property (the property is not negligible).

We now define what a natural property is:

Definition 6 (Natural property)**.** *A property $\mathcal{P}$ is* Γ-natural *if it is Γ-constructive and large.*

In essence, the constructivity property tries to capture the fact that we look for a uniform bound on the non-uniform class, and the largeness property follows the intuition that we not only prove our function is not in $\mathcal{C}$ but rather prove that many functions (often, almost all functions) are not in $\mathcal{C}$.

We also want our property to give examples that are not in $\mathcal{C}$ so we define:

Definition 7. *A property $\mathcal{P}$ is* useful *against a non-uniform class $\mathcal{C}$ if for every n large enough, $\mathcal{P}_n \cap \mathcal{C}_n = \emptyset$. (Alternatively, another reasonable definition is that any sequence from $\mathcal{P}$ infinitely often does not belong to $\mathcal{C}$).*

For example suppose we want to prove a uniform bound on $\mathsf{SIZE}(n^5)$. Suppose L is a language in $\mathsf{SIZE}(n^5)$. Then, there are functions $L_n \in F_n$ such that $\mathsf{SIZE}(L_n) \leq n^5$. We let $\mathcal{C}_n$ be all functions on n variables that have circuits of size n^5, and in this notation $L_n \in \mathcal{C}_n$ for all n. A natural lower-bound on $\mathsf{SIZE}(n^5)$ would present a property $\mathcal{P}$ that is constructive and large and *disjoint* from $\mathcal{C}$.

We remark that even if we have a natural proof $\mathcal{P}$ against $\mathcal{C}$ we dot not necessarily have a uniform function not in $\mathcal{C}$ (sampling a function in F_n, checking it is in $\mathcal{P}_n$ and evaluating it on the input does not work (why?), finding the lexicographically first function not in $\mathcal{C}$ requires many non-deterministic oracle class) and it is up to the lower-bound proof to show an explicit function with property $\mathcal{P}$.

Yet, Razborov and Rudich argue that even natural proofs are probably not strong enough to prove lower bounds on non-uniform classes, i.e., if we believe in cryptography. So next we give some very brief background about pseudo-random generators in cryptography (vs. the ones we have seen in complexity).

2 Quoting from the [3] paper

Their article states:

> [...] consider a commonly envisioned proof strategy for proving $\mathsf{P} \neq \mathsf{NP}$:
>
> - Formulate some mathematical notion of "discrepancy" or "scatter" or "variation" of the values of a Boolean function, or of an associated polytope or other structure. [...]
> - Show by an inductive argument that polynomial-sized circuits can only compute functions of "low" discrepancy. [...]
> - Then show that SAT, or some other function in NP, has "high" discrepancy.
>
> Such a proof is likely to be natural. If cryptography is correct, such proofs cannot work for $\mathsf{P} \neq \mathsf{NP}$.

3 Cryptographic PRGs

We recall the definition of a PRG we worked with:

Definition 8. *A function $G : \{0,1\}^\ell \to \{0,1\}^m$ is a PRG against a non-uniform class $\mathcal{C}$ if for every $T \in \mathcal{C}$,*

$$\left| \Pr_{x \in U_\ell}[T(G(x)) = 1] - \Pr_{y \in U_m}[T(y) = 1] \right| \leq \varepsilon.$$

We say G is efficient *if $G(x)$ runs in polynomial time in the* output length m.

We stress that we measured efficiency as a function of the *output length*. This is because for the application we have in mind we are going to go over all 2^ℓ seeds, so exponential time in the input length is fine with us. A byproduct of this is that the generator G runs in time greater than the

adversary C it tries to fool (that in our applications has complexity $\mathcal{C}$). Thus, in a sense, we do not try to generate randomness out of thin air, but rather in hard work. As we saw, even this we cannot do without assumptions, because it requires a uniform function hard against a non-uniform class, and we currently do not have such a function.

However, cryptography assumes much more. The OWF assumption assumes (among other things) that one can generate a function in polynomial time that is hard to invert even for much stronger adversaries. Indeed the OWF assumption requires $\mathsf{NP} \neq \mathsf{P}$, and in fact if we allow non-uniform adversaries as is normally the case, $\mathsf{NP} \not\subseteq \mathsf{P/poly}$, as opposed to the $\mathsf{EXP} \not\subseteq \mathsf{P/poly}$ that we needed for some partial derandomization. In fact the assumption $\mathsf{NP} \not\subseteq \mathsf{P/poly}$ does not suffice to prove the existence of a OWF, but we will not get into it here.

Thus, cryptography assumes much more and gets much more. The thing is most people believe that the assumptions underlying cryptography are correct. Thus, let us assume there are indeed OWFs, and we will see that if this is the case then there are no P-natural proofs against $\mathsf{P/poly}$.

Suppose there are OWFs. Then, a celebrated result shows there exist the following PRGs:

Definition 9. *If OWFs exist then there exists a PRG $\{G_k\}$ with $G_k : \{0,1\}^k \to \{0,1\}^{2k}$, computable in time* poly(k)*, such that for all c, for all circuits C of polynomial size $S \leq k^c$, for every large enough k,*

$$|\Pr[C(G_k(x)) = 1] - \Pr[C(U_{2k}) = 1]| \ \leq \ 1/S.$$

For simplicity let us assume that the PRG runs in poly(k) time and is hard for circuits of size $k^{\log\log\log k}$ (which can be achieved by choosing the assumptions accordingly). The important thing to notice is that the generator (that runs in poly(k) time) fools much stronger adversaries (that run in super-poly(k) time).

Once we have such wonderful PRGs we can get much more:

Definition 10. *A collection of functions $H \subseteq F_n = \{f : \{0,1\}^n \to \{0,1\}\}$ is a* pseudorandom function (PRF) *if for every circuit C (limited, say, by size, as above) and every $m =$ poly(n) indices $i_1, \ldots, i_m \in \{0,1\}^n$,*

$$\left| \Pr_{h \in H}[C(h(i_1), \ldots, h(i_m)) = 1] - \Pr[C(U_m) = 1] \right| \ \leq \ \varepsilon.$$

I.e., we use the seed to sample a function (before we had 2^k possible values for the seed and we will keep that). The seed determines a function with 2^n values (before we had only $2k$ values). The construction has the property similar to poly(n)-wise independence (but instead of true randomness we get computation indistinguishability from uniform), i.e., for any poly(n) values we choose to look at the values look to a limited adversary as uniform.

The transition from a PRG to a PRF (known as GGM, due to Goldreich, Goldwasser and Micali [2]) is simple and elegant.

Let G_0, G_1 be the first and last k bits of the output of G. For a string $y \in \{0,1\}^n$, define $G_y : \{0,1\}^k \to \{0,1\}^k$ by

$$G_y(x) \ = \ (G_{y_n} \circ G_{y_{n-1}} \circ \ldots \circ G_{y_1})(x).$$

For $x \in \{0,1\}^k$, let $h_x : \{0,1\}^n \to \{0,1\}$ be the function that on input $y \in \{0,1\}^n$, outputs the first bit of $G_y(x)$.

Theorem 11. *If G is a PRG then $\left\{h_x \mid x \in \{0,1\}^k\right\}_n$ is indeed a PRF.*

Proof. Think of the computation of $G_y(x)$ as a binary tree, where the root is x and every vertex v has a left child $G_0(v)$ and a right child $G_1(v)$. This is a complete binary tree of height n, where the computation is done via traversing along the y-path.

Assume towards contradiction that there exists a circuit C making $m = \text{poly}(n)$ distinct queries to h_x and distinguishes it from U_m with probability at least ε. We will show a circuit C' that distinguishes between $G(U_k)$ and U_{2k} with probability at least ε' for some ε' polynomial in ε.

We will partially construct the above binary tree in correspondence with the queries made by C and view the procedure as the way C obtains $h_x(i_1), \ldots, h_x(i_m)$ for some fixed $i_1, \ldots, i_m$. Initially, the tree contains only the root x. Whenever a query $h_x(i_j)$ is made, let v be the lowest point in the i_j-path from the root that is already computed. We then compute all the values along the path from v to the leaf corresponding to i_j. Whenever a labeled vertex is being computed, both its children are re-labeled and the label of the vertex itself is erased. This procedure makes at most $M = m \cdot n$ invocations of G.

For $i = 0, 1, \ldots, M$ define H^i in the following way: H^i is the above procedure for obtaining $h_x(i_1), \ldots, h_x(i_m)$, except that in the first i times when G is supposed to be invoked in order to label the two children of some node v labeled z, we do not do this but instead perform a "uniform" labeling: $G(z) = (z_0, z_1)$ for random $z_0, z_1 \in \{0,1\}^k$, instead of $G(z) = (G_0(z), G_1(z))$. Clearly, H^0 is the original invocation while H^M gives the uniform distribution. For some $i \in [M]$, we can therefore distinguish $C(H^i)$ from $C(H^{i-1})$ with probability at least ε/M.

Before we continue, make the following modification to the uniform labelings of H^i: On a vertex v, we erase its label and mark both its children as "to be chosen at random" and will choose each of these labels at random only when it will be needed in a future time. Note that typically the label for one of the children will be needed in the next step, but the label for the other child may only be required to answer a future query or perhaps never. Even the root x is not chosen initially but rather is initiated with the "to be chosen at random" label. Note that for the first i uniform labelings whenever the value for an internal vertex is used then it is immediately erased, and so in the first i steps all the internal vertices are either untouched or marked "to be chosen at random".

We now describe a circuit C' that distinguishes between $G(U_k)$ and U_{2k}. On input $w \in \{0,1\}^{2k}$, run the above procedure until the i-th uniform invocation. In this invocation we are supposed to take an internal vertex v which is marked "to be chosen at random", and choose a random value z for it. In H^{i-1} we choose $(z_0, z_1) = G(z)$ and use it to label vs children, then erase z. In H^i we choose z_0 and z_1 at random. C' will simply let $(z_0, z_1) = w$ and use this as the labeling.

It is clear that if $w \sim G(U_k)$ then we get H^{i-1} and if $w \sim U_{2k}$ we get H^i. Therefore, the success of C' is the success of distinguishing between $C(H^{i-1})$ and $C(H^i)$, which is ε'. The size of C' is polynomial in the size of C, so we are done. □

We notice that we are in a familiar situation. Once we sample $h \in H$ it defines a function with 2^n outputs, or, equivalently a truth table with 2^n entries. However, the construction is so that computing the value of h on a specific entry $y \in \{0,1\}^n$ can be done in $\text{poly}(k, n)$ time, Thus, the truth table defines an easy function!

4 Natural proofs cannot prove strong lower-bounds on circuit size

Theorem 12. *Let $H \subseteq F_n$ be the GGM construction built upon a PRG G that fools circuits of size s with ε error. Then, if there is a* P*-natural proof against* $\mathsf{P/poly}$ *then there exists a distinguisher running in time* $\mathrm{poly}(2^n)$ *that* $\frac{\varepsilon}{2^n}$*-distinguishes G from uniform.*

Proof. Suppose $\mathcal{P}$ is a P-natural proof against $\mathsf{P/poly}$. As we noticed before, the GGM construction constructs a set $H \subseteq F_n$ of functions h, all of them belong to $\mathsf{P/poly}$. Hence, the intersection between H and $\mathcal{P}$ is empty (at least for every large enough n).

Next, we define a distinguisher: given $f \in F_n$ it checks whether $f \in \mathcal{P}_n$. This distinguisher accepts the uniform distribution with a non-negligible probability δ (because of the largeness property) and rejects all elements $h \in H$ (because the do not belong to $\mathcal{P}$). Thus, it distinguishes between the uniform distribution over F_n and the flat distribution over H. The distinguisher runs in $2^{O(n)}$ time.

This distinguisher by itself is not a contradiction for the PRF property of H, because we look at all the 2^n entries in the table. Yet, a similar hybrid argument can translate it to a distinguisher for a pair with about the same running time, and $\frac{\delta}{2^n}$ advantage, which concludes the proof. □

If we do the parameters right, i.e., we assume a OWF with 2^{n^δ} hardness, we can get a PRG $G : \{0,1\}^k \to \{0,1\}^{2k}$ that is very hard against circuits of size almost exponential in k (say, 2^{-k^α} hard against circuits of size 2^{k^α} for some $\alpha > 0$). Then, we can choose n to be k^β for some $\beta < \alpha$. Thus, if there are such natural proofs against $\mathsf{P/poly}$, G is not as hard as it should be.

In short: If there are PRGs, then GGM gives functions in $\mathsf{P/poly}$ that cannot be naturally separated from uniform. If G and the PRF run in lower classes we should not get natural proofs against those classes. Currently, under widely believed assumptions there are PRFs in TC^0 (constant depth, polynomial size, and, or, threshold gates and unbounded fan-in) so under these beliefs there are no natural lower bounds against TC^0.

References

[1] Theodore Baker, John Gill, and Robert Solovay. Relativizations of the P=?NP question. *SIAM Journal on computing*, 4(4):431–442, 1975.

[2] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of the ACM (JACM)*, 33(4):792–807, 1986.

[3] Alexander A Razborov and Steven Rudich. Natural proofs. *Journal of Computer and System Sciences*, 55(1):24–35, 1997.

[4] R Ryan Williams. Natural proofs versus derandomization. *SIAM Journal on Computing*, 45(2):497–529, 2016.