03684155: On the P vs. BPP problem. 8/1/2017

Take-Home Exam

Amnon Ta-Shma and Dean Doron

General instructions:

1. The deadline for the exam and the take home project is 15/2/17.
2. Solve at least 3 questions.
3. Submit your (typed) solution by mail to amnon@tau.ac.il and deandoron@mail.tau.ac.il.
4. Some questions are based on published papers. We do not give the due credit, because we want to encourage you to try it yourself.
5. Please sign the attached statement that you indeed did it alone. Also, for each question you solved with the help of some external source (be it electronic or human) please mention the source explicitly.
6. If you find a mistake (or a typo), or you have a suggestion that may benefit others, please let us know as soon as possible.
7. You have more than a month, use it!
8. Enjoy!

1 Non-uniform lower bounds implies non-trivial derandomization

Question 1.1. *Prove that if either* $\mathsf{NEXP} \not\subseteq \mathsf{P/poly}$ *or* $\mathrm{PERM} \notin \mathsf{AP/poly}$*, then arithmetic circuit identity testing for size n circuits computing polynomials of degree and maximum coefficient size at most* $\mathrm{poly}(n)$ *is in* $\mathsf{io\text{-}NTIME}(2^{n^{\varepsilon}})/n^{\varepsilon}$ *for every constant* $\varepsilon > 0$.

2 Derandomization under uniform assumptions

Question 2.1. *Prove that there are functions in* EXP *that are not in* $\mathsf{io\text{-}HeurDTIME}_{2/3}(2^{o(n)})/o(n)$.

Use what we proved in class to conclude that:

Question 2.2. *Prove: if* $\mathsf{EXP} \cap \mathsf{P/poly} = \mathsf{BPP}$ *then* $\mathsf{EXP} = \mathsf{BPP}$*, i.e., if* $\mathsf{BPP} \neq \mathsf{EXP}$ *then* EXP *contains a language in* $\mathsf{P/poly}$ *that is not in* BPP.

3 Lower bounds imply derandomization of space-bounded classes

In class we constructed, for every ℓ, m, an explicit $(\ell, a = \log m)$-design $Z_1, \ldots, Z_m \subseteq [t]$ where $t = O(\ell^2)$. Moreover, given $i \in [m]$ and $j \in [\ell]$, we could output the j-th element of S_i in space $O(\ell \log m)$. Note that if we insist on $\ell = O(\log m)$, t is not logarithmic in m and the construction is not fully-explicit. We want to aim higher, and construct a design that achieves just that.

Question 3.1. *Prove that for every m there exist constants c_1, c_2 and c_3 for which exists a $(c_1 \log m, c_2 \log m)$-design $Z_1, \ldots, Z_m \subseteq [t]$ where $t = c_3 \log m$. Moreover, we can generate each element of S_i in space $O(\log m)$.*

Hint: Choose the S_i-s in a pairwise independent way.

Now, show that the techniques we have seen in class can be adopted to prove a "hardness implies derandomization" result in the space-bounded regime. As usual, we denote $\mathsf{L} = \mathsf{DSPACE}(O(\log n))$. A probabilistic space-bounded Turing machine is similar to the deterministic one (and so requires to halt in polynomial-time) except that it also has a read-only, uni-directional random coins tape. We denote $\mathsf{BPL} = \mathsf{BPSPACE}(O(\log n))$. Prove:

Question 3.2. *If there is a Boolean function computable in $\mathsf{DSPACE}(O(n))$ such that $\mathsf{Size}(f) \geq 2^{\Omega(n)}$ then there exists a Boolean function computable in $\mathsf{DSPACE}(O(n))$ that is average-case hard for circuits of size $2^{\Omega(n)}$.*

And,

Question 3.3. *If there is a Boolean function computable in $\mathsf{DSPACE}(O(n))$ such that $\mathsf{Size}(f) \geq 2^{\Omega(n)}$ then $\mathsf{BPL} = \mathsf{L}$.*

4 Approximate-counting is in the polynomial-time hierarchy

We know that $\mathsf{PH} \subseteq \mathsf{P}^{\#\mathsf{P}}$, but what about *approximating* a $\#\mathsf{P}$-function? You will show that it can be done in $\mathsf{BPP}^{\mathsf{NP}}$ (that is, already in the third level of PH).

Question 4.1. *Let #CSAT be the problem where given a circuit, count the number of accepting inputs. Let f be an arbitrary function in $\#\mathsf{P}$. Prove that if we have a polynomial-time algorithm A and a constant c such that for every circuit C,*

$$\frac{1}{c} \cdot \#\mathrm{CSAT}(C) \;\leq\; A(C) \;\leq\; c \cdot \#\mathrm{CSAT}(C),$$

then given $\varepsilon > 0$ there exists an algorithm B such that for every x,

$$(1-\varepsilon) \cdot f(x) \;\leq\; B(x) \;\leq\; (1+\varepsilon) \cdot f(x),$$

and runs in time $\mathrm{poly}(|x|, 1/\varepsilon)$.

Question 4.2. *Let $\mathcal{H} \subseteq \{0,1\}^n \to \{0,1\}^m$ be a two-universal family of hash functions, and let $S \subseteq \{0,1\}^n$. Prove that for every t,*

$$\Pr_{h \in \mathcal{H}}\left[\left| |\{a \in S : h(a) = 0^m\}| - \frac{|S|}{2^m} \right| \geq t\right] \;\leq\; \frac{|S|}{t^2 2^{2m}}.$$

Question 4.3. *For every $k \geq 0$, define the promise problem* CSAT_k*:*

- *Yes instances: Circuits C for which* $\#\mathrm{CSAT}(C) \geq 2^{k+1}$.
- *No instances: Circuits C for which* $\#\mathrm{CSAT}(C) < 2^k$.

Prove that for every k, CSAT_k *can be solved in* $\mathsf{BPP}^{\mathsf{NP}}$.

Question 4.4. *Prove that for every $f \in \#\mathsf{P}$ and $\varepsilon, \delta > 0$ there is a probabilistic algorithm A such that*

$$\Pr_A[(1-\varepsilon)A(x) \leq f(x) \leq (1+\varepsilon)A(x)] \;\geq\; 1-\delta,$$

the algorithm runs in time $\mathrm{poly}(|x|, 1/\varepsilon, \log(1/\delta))$ *using an oracle for* NP.