Complexité avancée - TD 10

Rémy Poulain

27 novembre 2019

Exercise 0 : BPP and oracle machines

Prove that $\mathsf{P}^{\mathsf{BPP}} = \mathsf{BPP}$.

Solution:

Idea : To show that $\mathsf{P}^{\mathsf{BPP}} \subseteq \mathsf{BPP}$ we need to simulate all calls to the oracle BPP by our language BPP, so the difficulty is in the duplication of random words. We have already seen this in the exercice with the self-reduction. The main idea is to use error reduction on the oracle and not on our built language.

Bad idea : Care if you say something like : $\mathsf{BPP}^{\mathsf{BPP}} \subseteq \mathsf{BPP}$ that's true here but that's not obvious, think about $\mathsf{NP}^{\mathsf{NP}}$ and NP.

Proof (scheme) :

$\mathsf{BPP} \subseteq \mathsf{P}^{\mathsf{BPP}}$: Obvious, cause you can just ask to the oracle the answer.

$\mathsf{BPP} \subseteq \mathsf{P}^{\mathsf{BPP}}$: Let $L \in \mathsf{P}^{\mathsf{BPP}}$, by definition $\exists B \in \mathsf{BPP}$ an oracle and M a TM (of execution time lower than p a poly.) which decide together L. We know that for q a poly., we can have M_q a PTM such that M_q decide B with an error lower than $e^{-q(.)}$. So, if we simulate all calls to the oracle by BPP, the error is lower than $p(.)e^{-q(.)}$. Notice that we can set q after that p is given, so we can set q to have a good error. Moreover, for an input of length n, we have a number of random words polynomial (of n) and all are of lenght polynomial (of $p(n)$, so of n), then our big random word used to simulate all random words have a length polynomial of n. Therefore $L \in \mathsf{BPP}$ □

Exercise 1 : Probabilistic Logarithmic Space

Propose a definition of RSPACE.

Let $\mathsf{RL} = \bigcup_{k\in\mathbb{N}} \mathsf{RSPACE}(k \cdot log(n), \infty, 0, 1/2)$ be the class of languages that can be decided in probabilistic logarithmic space (the machine does not necessarily halt).

Show that :

1. For L in RL and M a PTM which decides L , If $x \notin L$, then $\forall r$, $M(x, r) \neq \top$
2. $\mathsf{RL} \subseteq \mathsf{NL}$
3. $\mathsf{RL} \subseteq \mathsf{RP}$

Solution:

Idea : What's difficult here is the possibility that the Machine doesn't stop. Moreover (for the same reason) we don't have a boundary on our random word so we have a probability on an infinite set (as in ZPP). For the definition we won't use the usual definition by contraposition because there is not two but three cases.

Proof : Definition : $\mathsf{RSPACE}(p(n), \infty, reject(n), accept())$ is the class of all languages L such that there is a randomized Turing machine M, working in space $O(p(n))$, that terminates with probability 1, and such that :

— If $x \in L$, then $Pr[M(x,r) = \top] \geq 1 - accept(n)$

— If $x \notin L$, then $Pr[M(x,r) = \bot] \geq 1 - reject(n)$

1. For $x \notin L$, if $\exists r_0$ s.t $M(x, r_0) = \top$ then for all word w $M(x, r_0 w) = \top$ so $Pr[M(x,r) = \top] > 0$, then $Pr[M(x,r) = \bot] < 1$: $\lightning$. □
2. $\mathsf{RL} \subseteq \mathsf{NL}$:
   Given $L \in \mathsf{RL}$, M a RTM which decides L, we build the NDMT M' which follows the algorithm :
   Input: x a word
   Guess r a random word (bit by bit) ;
   return Simulate $M(x,r)$
   Therefore :
   — if $x \in L$ then $Pr[M(x,r) = \top] \geq \frac{1}{2}$, so $(\exists r, M(x,r) = \top)$ then $M'(x) = \top$
   — if $x \notin L$ then $(\nexists r, M(x,r) = \top)$ then $M'(x) = \bot$
   Notice that the first assertion is true if we take $|r| > 1$. Therefore : M' recognize L. Moreover, the resulting NL machine runs in space $klog(n)$ for some k, but may fail to terminate. As in the lectures, we know that any run of more than $a^{klog(n)}$ (where a is the alphabet size) will visit the same configuration twice. So we can stop any run when it exceeds that number of steps, and reject. This requires an counter of size $klog(n)$. Then $L \in \mathsf{NL}$. □
3. $\mathsf{RL} \subseteq \mathsf{NL} \subseteq \mathsf{P} \subseteq \mathsf{RP}$ □

Exercise 2 : BPP-completeness

1. Show that the language $L = \{(M, x, 1^t) \mid M \text{ accepts on input } x \text{ in time at most } t\}$, where M is the code of a non-deterministic Turing machine, x an input of M and t a natural number, is NP-complete.
2. Let L be the language of words $(M, x, 1^t)$ where M designates the encoding of a probabilistic Turing machine and x a string on $M's$ alphabet such that M accepts x in at most t steps, for at least $2/3$ of the possible random tapes of size t.
   Is L BPP-hard ? Is it in BPP ?

Solution:

1. — $L \in \mathsf{NP}$:
   Let M be the code of a non-deterministic Turing machine, x an input of M and t a natural number.
   Notice that the execution time of $M(x)$ is t so lower than the length of $(M, x, 1^t)$. So the algorithm which simulates M on x on the input $(M, x, 1^t)$ is non-deterministic polynomial. Then we can check that $(M, x, 1^t) \in \{(M, x, 1^t) \mid M \text{ accepts on input } x \text{ in time at most } t\}$.
   Therefore, $L \in \mathsf{NP}$.
   — L is NP-hard :
   Given $L' \in \mathsf{NP}$, M a NDTM for L', and p a polynomial associated.
   For an instance x of L' we can build (polynomially) the instance $(M, x, 1^{p(|x|)})$
   And, by definition of L, $(M, x, 1^{p(|x|)}) \in L \Leftrightarrow x \in L'$
   □
2. — L is BPP-hard : (for exactly the same reasons).
   Given $L' \in \mathsf{BPP}$, M a NDTM for L', and p his polynomial associated.
   For an instance x of L' we can build (polynomially) the instance $(M, x, 1^{p(|x|)})$
   And, by definition of L, $(M, x, 1^{p(|x|)}) \in L \Leftrightarrow x \in L'$
   □

— Can't say the same thing cause bpp is not syntactic.

Definition 1 *Recall that* $\mathsf{AM}[f]$ *for a proper function* f *denotes the class of languages* L *such that for any* $\ell \geq 0$*, there exists a game of Arthur and Merlin* (M, A, D) *such that for any* x *of size* n*, letting* $prot = (AM)^{f(n)}$ *:*

1. *Completeness : if* $x \in L$ *then* $prot[A, M]_D = \top$ *with probability at least* $1 - 1/2^{n^\ell}$
2. *Soundness : if* $x \notin L$ *then for any Merlin's function* M'*,* $prot[A, M']_D = \bot$ *with probability at least* $1 - 1/2^{n^\ell}$

Exercise 3 : Arthur-Merlin protocols

Prove the following statements, directly from definition of Arthur-Merlin games :

— $\mathsf{M} = \mathsf{NP}$;

Solution:
Notice that for a language L :
$L \in \mathsf{M} \Leftrightarrow \exists D \in \mathsf{P}, \exists p\ poly, L = \{x \mid \exists y, |y| < p(|x|) \wedge x\#y \in D\}$. It's exactly the certificate definition of NP. □
Notice that we won't usually put the condition on p, but it's implicit.

— $\mathcal{A} = \mathsf{BPP}$;

Solution:
Obvious, just have to write the two definitions :
$\mathsf{BPP} \subseteq \mathcal{A}$: Take $D = \Sigma^*\#\Sigma^*\#\top$, and A which simulates our BPP machines and write the answer.
$\mathcal{A} \subseteq \mathsf{BPP}$: Just simulate A and check if it's in D.
□

— $\mathsf{NP}^{\mathsf{BPP}} \subseteq \mathsf{MA}$;

Solution:
Let $L \in \mathsf{NP}^{\mathsf{BPP}}$, then we have $L' \in P^{\mathsf{BPP}}$ such that $L = \{x \mid \exists y, x\#y \in L'\}$. Moreover we know from the previous TD that $P^{\mathsf{BPP}} = \mathsf{BPP}$, and from the previous answer that $\mathcal{A} = \mathsf{BPP}$. Therefore we have $L' \in \mathcal{A}$ such that $L = \{x \mid \exists y, x\#y \in L'\}$. So $L \in \mathsf{MA}$. □

— $\mathsf{AM} \subseteq \mathsf{BPP}^{\mathsf{NP}}$.

Solution:
Let L be in AM, we have M, A, D given by the definition of AM, we can set the error at $1/3$. Define A' the PTM s.t. for an input x and a random word r, $A'(x, r) = x\#r\#A(x, r)$. Moreover, define $D' = \{z \mid \exists y\ z\#y \in D\}$, $D' \in \mathsf{NP}$ because $D \in \mathsf{P}$. Let M_o be the probabilistic oracle machine which simulates A' and call the oracle D' on the answer, accepting with the BPP way.
— If $x \in L$, $P[M_o(x, r) = \top] = P[x\#r\#A(x, r) \in D'] = P[\exists y\ x\#r\#A(x, r)\#y \in D] \geq \frac{2}{3}$ (it's the definition of AM)
— If $x \notin L$, $P[M_o(x, r) = \bot] = P[x\#r\#A(x, r) \notin D'] = P[\forall y\ x\#r\#A(x, r)\#y \notin D] \geq \frac{2}{3}$
Then $L \in \mathsf{BPP}^{\mathsf{NP}}$ □

Exercise 4 : Collapse of the Arthur-Merlin hierarchy

Recall that, for each $\Pi \in \{A, M\}^*$, the class $\mathbf{\Pi}$ is the class of languages recognized by Arthur-Merlin games with protocol Π.

(a) Without using any result about the collapse of the Arthur-Merlin hierarchy, prove that for all $\Pi_0, \Pi_1, \Pi_2 \in \{A, M\}^*$, we have $\mathbf{\Pi_1} \subseteq \mathbf{\Pi_0\Pi_1\Pi_2}$.

Solution:

This is not a very formal proof but there is the idea : Let $\Pi_0, \Pi_1, \Pi_2 \in \{A, M\}^*$. Let $L \in \Pi_0\Pi_1\Pi_2$, and D a polynomial language given by the definition of the the Arthur-Merlin protocol. We can restrict D to a language containing words likes $x\#x_0\#x_1\#x_2$ where obviously x_0 is "generated" byΠ_0, x_1 by Π_1 and x_2 by Π_2. Let ϕ be the projection s.t. $\phi(x\#x_0\#x_1\#x_2) = x\#x_1$. Notice that ϕ is well defined, because even if the x_i can contain # we know exactly the number of # in each word (that just depends of the number of letters in the Π_i). Moreover ϕ is polynomial.
Now lets $L \in \Pi_1$, Pc a protocol deciding L and D' a polynomial language given by the definition of the the Arthur-Merlin protocol. We simulate it by a protocol of $\Pi_0\Pi_1\Pi_2$ which does the same thing that Pc "in" Π_1, and Arthur does nothing and Merlin does whatever he wants in Π_O and Π_2. Set $D = \{w \ s.t. \ \phi(w) \in D'\}$. D is polynomial because D' and ϕ. So $L \in \Pi_0\Pi_1\Pi_2$. □

(b) Now assume the fact that for all $\Pi \in \{A, M\}^*$, one has $\mathbf{\Pi} \subseteq \mathsf{AM}$. Prove the following statement : For all $\Pi \in \{A, M\}^*$ such that Π has a strict alternation of symbols, and $|\Pi| > 2$, we have $\mathbf{\Pi} = \mathsf{AM}$.

Solution:
Let Π be such a protocol. We already know that $\mathbf{\Pi} \subseteq \mathsf{AM}$, Moreover $\Pi = AMA\Pi'$ or $\Pi = MAM\Pi'$. In both cases, it contains AM. Therefore $\mathsf{AM} \subseteq \mathbf{\Pi}$ □

Exercise 5 : The BP operator

We say that a language B reduces to language C under a randomized polynomial time reduction, denoted $B \leq_r C$, if there is a probabilistic polynomial-time Turing machine such that for every x, $Pr[C(M(x)) = B(x)] \geq \frac{2}{3}$.

Remember the definition of $\mathsf{BP} \cdot \mathcal{C}$:
$L \in \mathsf{BP} \cdot \mathcal{C}$ iff exists A a PTM polynomial and a language D polynomial s.t. for all input x
— if $x \in L$ then $Pr[A(x, r) \in D] \geq \frac{2}{3}$
— if $x \notin L$ then $Pr[A(x, r) \notin D] \geq \frac{2}{3}$

1. Show that $\mathsf{BP} \cdot \mathcal{C} = \{L \mid L \leq_r L', \text{ for some } L' \in \mathcal{C}\}$
2. Show that BPP is closed under randomized polynomial time reduction.
3. Deduce that $\mathsf{BP} \cdot (\mathsf{BP} \cdot \mathcal{C}) = \mathsf{BP} \cdot \mathcal{C}$.

Solution:

1. obvious
2. Let $B \in \mathsf{BPP}$, we know that we can have M_B a PTM which decides B with an error lower than $\frac{1}{12}$. Let $C \leq_r B$, we have M a probabilistic polynomial-time Turing machine such that for every x, $Pr[C(M(x)) = B(x)] \geq \frac{2}{3}$.. Let M_C be the PTM which simulates, for an input x and two random words r and r', $M_B(M(x, r'), r)$. Then :
   — If $x \in C$, $P[M_C(x, r) = \bot] = P_{(r',r)}[M_B(M(x, r'), r) = \bot] \leq P_{r'}[C(M(x, r')) = B(x)] + P_{r,if\ y \in B}[M_B(y, r) = \bot] \leq \frac{1}{12} + \frac{1}{3} \leq \frac{5}{12}$
   — If $x \notin C$, $P[M_C(x, r) = \top] = P_{(r',r)}[M_B(M(x, r'), r) = \top] \leq P_{r'}[C(M(x, r')) = B(x)] + P_{r,if\ y \notin B}[M_B(y, r) = \top] \leq \frac{1}{12} + \frac{1}{3} \leq \frac{5}{12}$
   Therefore $C \in \mathsf{BPP}$ □
3. By composition of reductions □

Exercise 6 : The class BP · NP

1. Show that $\mathsf{BP} \cdot \mathsf{P} = \mathsf{BPP}$

Solution:
Obvious, I do one inclusion :
Let $L \in \mathsf{BPP}$, we have A a PTM given by the definition of BPP, we just simulate it with A' which won't accept or reject but will write $\top$ or $\bot$. Let $D = \top$. By definition of BPP :
— If $x \in L$, $P[A(x,r) \in D] = P[A'(x,r) = \top] \geq \frac{2}{3}$
— If $x \notin L$, $P[A(x,r) \notin D] = P[A'(x,r) = \bot] \geq \frac{2}{3}$
Therefore $\mathsf{BPP} \subseteq \mathsf{BP} \cdot \mathsf{P}$ □

2. Show that $\mathsf{BP} \cdot \mathsf{NP} = \mathsf{AM}$

Solution:
The solution is in the course p.31, and it's the same construction that $\mathsf{AM} \subseteq \mathsf{BPP}^{\mathsf{NP}}$.

3. Show that $\mathsf{BP} \cdot \mathsf{NP} \subseteq \Sigma_3^P$ (give a direct proof, do not use $\mathsf{AM} \subseteq \Pi_2^{\mathsf{P}}$).

Solution:
As in the course, we can see that if $L \in \mathsf{BP} \cdot \mathsf{NP}$, we have M a NDTM polynomial and q a poly. such that : $x \in L \Leftrightarrow \exists t_0 ... t_{\lfloor q(n)/n \rfloor}\ \forall r \bigvee_i M(x, r \oplus t_i)$.
The only difference here is that the Machine M is non deterministic. Then $L \in \Sigma_3^P$ □
In fact $\mathsf{BP} \cdot \Sigma_i^P \subseteq \Sigma_{i+2}^P$

4. Show that $\mathsf{BP} \cdot \mathsf{NP} \subseteq \mathsf{NP}/poly$

Solution:
Same proof that $\mathsf{BPP} \subseteq \mathsf{P}/poly$

5. (bonus) Show that if $\overline{\mathbf{3SAT}} \leq_r \mathbf{3SAT}$ then PH collapses to the third level.

Solution:
Idea : adapt Karp Lipton

Exercise 5 (bonus) : One Merlin to rule them all

Show that the following definition of AM if actually equivalent to the one given in introduction : $L \in \mathsf{AM}$ iff for any $\ell \geq 0$, there exists an Arthur A and a polynomial-time-checkable predicate D such that for any x of size n, letting $prot = (AM)^{f(n)}$:

1. Completeness : if $x \in L$ then there exists some Merlin M such that $prot[A, M]_D = \top$ with probability at least $1 - 1/2^{n^\ell}$
2. Soundness : if $x \notin L$ then for any Merlin M', $prot[A, M']_D = \bot$ with probability at least $1 - 1/2^{n^\ell}$

Solution:
Just the idea : Use the axiom of choice

Exercise 8 (bonus) : Unreliable Merlin

Show that allowing Merlin to use randomness (in a private manner) does not change the class AM.

Solution:
Just the idea : average on all possible merlin