ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Final Exam, Computational Complexity 2018

- You are only allowed to have a handwritten A4 page written on both sides.
- Communication, calculators, cell phones, computers, etc... are not allowed.
- Your explanations should be clear, well motivated, and proofs should be complete.
- The solutions to the questions of the exam are rather short. If you end up writing a solution requiring a lot of pages then there is probably an easier solution.
- **Do not touch until the start of the exam.**

Good luck!

Name: ______________________ **N° Sciper:** ______________

Problem 1 / 30 points	Problem 2 / 20 points	Problem 3 / 25 points	Problem 4 / 25 points

Total / 100

1 *(30 pts)* Basic questions with short answers.

1a *(10 pts)* Consider the operation of copying a qubit x by performing the map $|xy\rangle \mapsto |xx\rangle$ where y is an arbitrary qubit. Can this map be implemented as a quantum operation?

Solution:

No because quantum operations are reversible and the above map is not (there is no way to recover $|xy\rangle$ from $|xx\rangle$).

1b *(10 pts)* Briefly explain why no proof can resolve the **P** vs **NP** question if it uses only these two facts about Turing Machines:

1. The existence of an effective representation of Turing machines by strings.
2. The ability of one Turing Machine to simulate any other without much overhead in running time or space.

Solution: Such a proof would also work for oracle Turing Machines and be oblivious to the oracle. But we know that there exist an oracle A such that $\mathbf{P^A} = \mathbf{NP^A}$ and an oracle B such that $\mathbf{P^B} \neq \mathbf{NP^B}$. Hence any proof would need to depend on more properties.

1c *(10 pts)* Let $g : \{0,1\}^n \to \{0,1\}$ be a function that is $(1-\epsilon)$-close to a Walsh-Hadamard code. In other words, g is $(1-\epsilon)$-close to a linear function $f : \{0,1\}^n \to \{0,1\}$. Describe a "decoding" algorithm that on input g and $x \in \{0,1\}^n$ has the following guarantees:

1. It outputs $f(x)$ with probability at least $(1-2\epsilon)$. (We emphasize however that the algorithm has *no* access to f, only to g.)
2. It evaluates g on two inputs.

Description of algorithm (no analysis needed):

1. Select $r \in \{0,1\}^n$ uniformly at random.
2. Output $f(x+r) + f(r)$.

2 *(20 pts)* **Circuits.** In the last lecture, we saw that there are functions that require circuits of linear depth *assuming that AND and OR gates have fan-in* 2 (and NOT gates have fan-in 1). In this problem, we are going to consider circuits where AND and OR gates are allowed to have unbounded fan-in. In that case, any function $f : \{0,1\}^n \to \{0,1\}$ can be calculated using a circuit of *constant* depth. While this shows that the required depth changes dramatically if we have unbounded fan-in, this is not the case for the size of the circuit. Indeed, your task is to prove the following:

> For every n large enough, there exists an n-ary function $f : \{0,1\}^n \to \{0,1\}$ not computable by circuits of size $2^{n/3}$ even if AND and OR gates have unbounded fan-in.

Hint: recall that most functions f require circuits of large size when fan-in is bounded by 2*. In particular, you are allowed to use the statement proved in class about the bounded fan-in circuit size of most functions.*

Solution:

- We know from class that there exists a function $f : \{0,1\}^n \to \{0,1\}$ that requires $\Omega(2^n/n)$ gates of fan-in at most 2.
- Now consider a circuit C of size $2^{n/3}$ where gates have unbounded fan-in.
- Since C has size $2^{n/3}$ the fan-in of any gate is bounded by $2^{n/3}$.
- Now observe that we can replace every AND (OR) gate of fan-in t with binary tree of fan-in-2 AND (OR) gates with t leaves and thus at most $2t$ gates in total.
- Hence, by doing this replacement for every gate, we can obtain a circuit C' that is identical to C and has at most $2^{n/3} \cdot 2 \cdot 2^{n/3} = 2^{2n/3+1}$ gates each of fan-in at most 2.
- As $2^{2n/3+1}$ is much smaller than $2^n/n^2$ for large enough n the statement of the exercise follows.

3 *(25 pts)* **Closed under reductions.** Prove that $\mathbf{DTIME}(2^{n^{100}})$ is *not* closed under many-to-one polynomial time reductions. In your proof you may assume the Time Hierarchy Theorem without proving it.

(Recall that a language A has a many-to-one polynomial time reduction (aka Karp reduction) to a language B, written $\leq_p$, if there is a polynomial time computable function $f(\cdot)$ such that for every instance $x \in \{0,1\}^*$ we have $x \in A \Leftrightarrow f(x) \in B$. Moreover, a class $\mathbf{C}$ *of languages is* closed *under polynomial many-to-one reductions if $A \leq_p B$ and $B \in \mathbf{C}$ implies $A \in \mathbf{C}$. A famous example of a class that is closed under such reductions is* **NP**.*)*

Solution:

- By the Time Hierarchy Theorem, there is a language $L \in \mathbf{DTIME}(2^{n^{102}}) \setminus \mathbf{DTIME}(2^{n^{100}})$.
- Now define the language $L' = \{\langle x, 0^{|x|^2}\rangle : x \in L\}$.
- We claim that $L' \in \mathbf{DTIME}(2^{n^{50}}$. To see this observe first that, given $y \in \{0,1\}^*$, we can in polynomial time verify that y is in the form $\langle x, 0^{|x|^2}\rangle$ for some x. Now since $L \in \mathbf{DTIME}(2^{n^{102}})$, we can verify that $x \in L$ in time $2^{|x|^{102}}$ which is at most $2^{|y|^{51}}$.
- Finally, there is a trivial Karp reduction from L to L'. Take any x and output $\langle x, 0^{|x|^2}\rangle$.
- We conclude that $\mathbf{DTIME}(2^{n^{100}}$ cannot be closed under Karp reductions since $L' \in \mathbf{DTIME}(2^{n^{100}})$ and $L \notin \mathbf{DTIME}(2^{n^{100}})$.

4 *(25 pts)* **Cryptography.** Let (E,D) be any polynomial-time computable encryption scheme with key length $\leq m/2$ on messages of length m that satisfies $\mathsf{D}_k(\mathsf{E}_k(x)) = x$ for every key k and message x.

In this problem we are going to show that (E,D) is not computationally secure if $\mathbf{P} = \mathbf{NP}$. Specifically, prove the following: Assuming $\mathbf{P} = \mathbf{NP}$, there is a polynomial time algorithm A such that for every input length m, there is a pair of messages $x_0, x_1 \in \{0,1\}^m$ satisfying:

$$\Pr_{b\in\{0,1\},k\in\{0,1\}^n}[A(\mathsf{E}_k(x_b)) = b] \geq 1 - \frac{1}{2}\cdot\frac{1}{2^{m/2}},$$

where n denotes the key length and by assumption $n \leq m/2$.

Solution:

- Let (E,D) be an encryption for messages of length m and with key length $n \leq m/2$.
- Let $S \subseteq \{0,1\}^*$ denote the support of $\mathsf{E}_{U_n}(0^m)$. Note that $y \in S$ if and only if $y = \mathsf{E}_k(0^m)$ for some k. Hence, if $\mathbf{P} = \mathbf{NP}$, then membership in S can be verified efficiently (by "guessing" the right k).
- Algorithm A will be very simple: on input y, it outputs 0 if $y \in S$ and it outputs 1 otherwise.
- We set $x_0 = 0^m$ and we will find some x_1 satisfying the statement of the lemma.
- If we let D_x denote the distribution $\mathsf{E}_{U_n}(x)$, then

$$\begin{aligned}\Pr_{b\in\{0,1\},k\in\{0,1\}^n}[A(\mathsf{E}_k(x_b)) = b] &= \frac{1}{2}\Pr[A(D_{x_0}) = 0] + \frac{1}{2}\Pr[A(D_{x_1}) = 1] \\ &= \frac{1}{2} + \frac{1}{2}\Pr[A(D_{x_1}) = 1].\end{aligned}$$

- It thus suffice in finding an x_1 such that $\Pr[A(D_{x_1}) = 1] \geq 1 - \frac{1}{2^{m/2}}$ or, equivalently, $\Pr[D_{x_1} \in S] \leq 1/2^{m/2}$.
- To see that such an x_1 exists, define $S(x,k)$ to be 1 if $\mathsf{E}_k(x) \in S$ and 0 otherwise. Then

$$\mathbb{E}_{x\in\{0,1\}^m}\mathbb{E}_{k\in\{0,1\}^n}[S(x,k)] = \mathbb{E}_{k\in\{0,1\}^n}\mathbb{E}_{x\in\{0,1\}^m}[S(x,k)] \leq 1/2^{m/2},$$

  where the last inequality follows from the fact that, for any fixed k, E_k is one-to-one and hence at most $2^n \leq 2^{m/2}$ of the x's can be mapped to the set S of size 2^n.
- Hence there must exist an x_1 such that $\mathbb{E}_{k\in\{0,1\}^n}[S(x_1,k)] \leq 1/2^{m/2}$ which is equivalent to $\Pr[D_{x_1} \in S] \leq 1/2^{m/2}$.