Final Exam, Theory of Computation 2019-2020

- Books, notes, communication, calculators, cell phones, computers, etc... are not allowed.
- Your explanations and proofs should be clear enough and in sufficient detail so that they are easy to understand and have no ambiguities.
- You are allowed to refer to material covered in the class including theorems without reproving them.
- **Do not touch until the start of the exam.**

Good luck!

Name: ____________________ **N° Sciper:** ____________

Problem 1 / 18 points	Problem 2 / 12 points	Problem 3 / 20 points	Problem 4 / 15 points	Problem 5 / 20 points	Problem 6 / 15 points

Total / 100

1 *(consisting of subproblems **a**-**b**, 18 pts)* **Basic questions.**

1a *(8 pts)* Write the formal definition of the class **NP**. If you use the concept of a verifier or a non-deterministic Turing machine, then explain what it is.

Solution:

A language is in NP if it has a polynomial-time verifier or equivalently a polynomial-time nondeterministic decider.

A polynomial-time verifier is a Turing Machine M such that given an input x, if x is in the language, then there exists a certificate C such that M accepts (x, C); if x is not in the language, then for every certificate C, M rejects (x, C).

A nondeterministic decider is an NTM N such that for each input x, every computation of N on x halts, and if x is in the language, then some computation of N on x accepts; if x is not in the language, then every computation of N on x rejects.

1b *(10 pts)* Which of the following statements are true?

1. All regular languages are in **NP**.
2. All languages in **P** are regular.
3. All irregular languages are decidable.
4. All decidable languages are in **NP**.
5. All decidable languages are recognizable.
6. If a language L is decidable, then its complement $\bar{L}$ is decidable.
7. If a language L is recognizable, then its complement $\bar{L}$ is recognizable.
8. If a language is **NP**-hard, then it is **NP**-complete.
9. The language $\{w \in \{0,1\}^* \mid \text{number of 0's in } w \text{ is divisible by } 2020\}$ is regular.
10. The language $\{0^n 1^n \mid n \geq 0\}$ is in **NP**.
11. $\{\langle M \rangle \mid M \text{ is a TM that halts on all inputs of length at least } 2020\}$ is recognizable.
12. If languages L_1 and L_2 are unrecognizable then $L_1 \cup L_2$ is unrecognizable.

*(A complete solution identifies **all** true statements. A fully correct solution is worth* 10 *points. A solution with one mistake is worth* 9 *points. A solution with two mistakes is worth* 7 *point. A solution with three mistakes is worth* 4 *points. A solution with four mistakes is worth* 1 *point. Solutions with more mistakes are worth* 0 *points. A mistake is to either indicate falsely that a false statement is true or to not indicate that a true statement is true.)*

Solution:

Among the above statements, the following are true 1, 5, 6, 9, 10

(In this problem you do not need to justify your answer.)

2 *(12 pts)* **NP-completeness.** Prove that the following language LONGCYCLE is **NP**-complete:

$$\text{LONGCYCLE} = \{\langle G\rangle \mid G \text{ is an undirected graph with a cycle that visits at least half the vertices}\}.$$

In your proof you may use that the following language is **NP**-complete:

$$\text{HAM} = \{\langle G\rangle \mid G \text{ is an undirected graph that is Hamiltonian}\}.$$

An undirected graph $G = (V, E)$ is said to be Hamiltonian if it contains a cycle of length $|V|$, i.e., a cycle that visits every vertex. Two examples of Hamiltonian graphs are the complete graph and the graph consisting of a single cycle of length $|V|$.

(In this problem you are asked to prove that the language LONGCYCLE is **NP***-complete. Recall that you are allowed to use that HAM is* **NP***-complete and to refer to material covered in the class including theorems without reproving them.)*

3 *(consisting of subproblems **a**-**b**, 20 pts)* **Reductions.** In the next two subproblems, we are going to consider reductions between languages INDSET and ODD. Recall from class that INDSET denotes the ***NP**-complete* language

$$\text{INDSET} = \{\langle G, k\rangle \mid G \text{ is an undirected graph with an independent set of size } k\}$$

and we define ODD to be the following *regular language*

$$\text{ODD} = \{w \in \{0,1\}^* \mid w \text{ has an odd number of 1's}\}\,.$$

3a *(10 pts)* In this subproblem, we are going to show that if INDSET is *poly-time* mapping reducible to ODD then $\mathbf{P} = \mathbf{NP}$. Specifically, your task is to prove the following statement:

If INDSET $\leq_p$ ODD then $\mathbf{P} = \mathbf{NP}$.

(In this problem you are asked to prove that if INDSET $\leq_p$ ODD then $\mathbf{P} = \mathbf{NP}$. Recall that you are allowed to refer to material covered in the class including theorems without reproving them.)

Solution:
ODD is in P as it is regular. By assumption that INDSET $\leq_p$ ODD and transitivity, INDSET is in P. But, INDSET is an NP-complete language. Hence, by definition, any language L $\in$ NP satisfies L $\leq_p$ INDSET. Therefore, we conclude that P=NP.

3b *(10 pts)* In this subproblem, we are going to show that INDSET is mapping reducible to ODD. Specifically, your task is to show that INDSET $\leq_m$ ODD.

(In this problem you are asked to give a mapping reduction from INDSET to ODD. Recall that you are allowed to refer to material covered in the class including theorems without reproving them.)

Solution: We first give a decider M for INDSET.

On input $\langle G, k\rangle$, M iterates through all possible subsets of size k. If there exists a subset where there is no edge between any two vertices in the set, then accept. Otherwise, reject.

It is a decider as it halts in finite amount of time. The correctness is easy to see as there exists an independent set of size k iff M finds it as it exhaustively tries every possibility for such a set.

We now give the function f for proving that INDSET$\leq_m$ODD.

f: On input $\langle G, k\rangle$:

1. Run M on $\langle G, k\rangle$.
2. If M accepts, output “1”, else output “0”.

f is computable as M is a decider.

For correctness, if $\langle G, k\rangle \in$ INDSET, then M accepts and $f(\langle G, k\rangle) =$ “1” $\in$ ODD. On the other hand, if $\langle G, k\rangle \notin$ INDSET, then M rejects and $f(\langle G, k\rangle) =$ “0” $\notin$ ODD.

4 *(15 pts)* **Regular languages.** Given a language $A \subseteq \{0,1\}^*$, let

$$B = \{\text{EVEN}(w) \mid w \in A\},$$

where EVEN(w) denotes the string with only the even-positioned letters of w. For example, EVEN(11101011) = 1001. Describe an NFA that shows that B is regular if A is regular.

(In this problem you are asked to construct an NFA that shows that if A is regular then so is B.)

Solution: Suppose A is regular. Then there is a DFA that recognizes it: $D = (Q, \Sigma, \delta, q, F)$. Take two copies of this DFA: $D_{odd} = (Q_o, \Sigma, \delta_o, q_o, F_o)$ and $D_{even} = (Q_e, \Sigma, \delta_e, q_e, F_e)$. We define a new NFA: $(Q', \Sigma', \delta', q', F')$ where:

- $Q' = Q_o \cup Q_e$
- $\Sigma' = \Sigma$
- $q' = q_o$
- $F' = F_o \cup F_e$
- $\delta'(q_{o,i}, \epsilon) = \{\delta_e(q_{e,i}, 0), \delta_e(q_{e,i}, 1)\}$
- $\delta'(q_{e,i}, 0) = \delta_o(q_{o,i}, 0)$
- $\delta'(q_{e,i}, 1) = \delta_o(q_{o,i}, 1)$

This NFA recognizes B so B is regular if A is regular.

(Informally: take two copies of the states of the DFA D, one for “odd" and one for “even". The starting state is one from the odd copy. The finishing states is the union of the finishing states from each copy. Starting from an odd state q, add an epsilon transition to the corresponding states in the even copy that can be reached from q. Starting from an even state q, add a transition when reading 0 to the corresponding state in the odd copy that is reached by reading 0 from q, and similarly for reading 1. So when we are in the odd copy, we have removed the odd-positioned letter by an epsilon transition to the even copy, and when we are in the even copy, we read the letter by doing the original transition, but to the odd copy. The original accepting states are in both copies, since there can be both odd- and even-length words in A. Note that this NFA would accept the empty string, if there are words of 1 letter in A.)

5 *(20 pts)* **Computability.** Classify the following language into one of: decidable, undecidable but recognizable, unrecognizable.

$$\text{SLOWHALT} = \{\langle M, x\rangle \mid M \text{ is a TM that halts on input } x \text{ after taking at least 2020 steps}\}.$$

Justify your answer with a formal proof.

(In this problem, you are asked to identify whether SLOWHALT is (decidable and recognizable), (undecidable and recognizable), or (unrecognizable) and provide a formal correctness proof. Recall that you are allowed to refer to material covered in the class including theorems without reproving them.)

Solution: The language SLOWHALT is recognizable but undecidable. To prove recognizability we build the following recognizer.

Recognizer $R =$ "on input $\langle M, x\rangle$

1. Simulate M on input x while counting the number of steps.
2. If the number of steps is at least 2020 then accept otherwise reject."

If $\langle M, x\rangle \in$ SLOWHALT then the line 1 will eventually stops after at least 2020 steps and the recognizer will accept the input $\langle M, x\rangle$. If $\langle M, x\rangle \notin$ SLOWHALT, either line 1 stops in less than 2020 steps and line 2 will reject, or line 1 never stops. In both case, the input $\langle M, x\rangle$ is not accepted.

We now prove that SLOWHALT is undecidable by providing a reduction from the language HALT $= \{\langle M, x\rangle \mid M$ is a TM and M halts on $x\}$ which is known to be undecidable by the lectures.

We define the following computable reduction $f(\langle M, x\rangle) = \langle M', x\rangle$ where M' is a TM defined as follows.

$M' =$ "on input y

1. Loop for 2020 steps.
2. Run M on y and output the same result."

We see that f is computable as the input x is unchanged and we only need to define the Turing Machine M'.

As for correctness of the reduction, if $\langle M, x\rangle \in$ HALT, then the machine M' (on input x) will first run 2020 useless steps because of line 1 then will stop in line 2. Hence $\langle M', x\rangle \in$ SLOWHALT. If $\langle M, x\rangle \notin$ HALT then the machine M' will never stop on input x because of line 2 hence $\langle M', x\rangle \notin$ SLOWHALT.

We can conlude that SLOWHALT is recognisable but undecidable.

6 *(15 pts)* **Large DFAs.** Let LARGE $= \{0^n 1^n \mid 1 \leq n \leq 10^6\}$. That is LARGE contains all strings that have n zeros followed by n ones where n is an integer between 1 and 10^6. As LARGE contains a finite number of strings (10^6 many strings), LARGE is a regular language. However, in this problem you are asked to prove that there is no "small" DFA that recognizes LARGE. More specifically, your task is to prove the following statement:

Any DFA that recognizes LARGE has at least 10^5 states.

(In this problem you should give a proof of the statement that any DFA that recognizes LARGE has at least 10^5 states. Recall that you are allowed to refer to material covered in the class including theorems without reproving them.)

Solution: We will show a stronger claim. We will show that any DFA that recognizes LARGE has at least 10^6 states.

Assume for the sake of contradiction that there exists a DFA Q that has fewer than 10^6 states and such that $L(Q) =$ LARGE. Consider a set $S := \{0^k : 1 \leq k \leq 10^6\}$. The number of states of Q is smaller than 10^6 and $|S| = 10^6$ so, by pigeonhole principle, there exist $w_1, w_2 \in S, w_1 \neq w_2$ such that Q after reading w_1 and w_2 ends up in the same state. By definition of S, without loss of generality, we have that $w_1 = 0^i, w_2 = 0^j$ for some $1 \leq i < j \leq 10^6$.

We claim that $0^j 1^i \in L(Q)$ and $0^j 1^i \notin$ LARGE. This would constitute the desired contradiction.

- $0^j 1^i \notin$ LARGE because $i \neq j$,
- $0^j 1^i \in L(Q)$ because Q after reading 0^j is in the same state as after reading 0^i and $0^i 1^i \in$ LARGE.