NAME: ____________________________

# CSE 431
# Computational Complexity Theory
# Solved Sample Final Exam

Anup Rao March 11, 2023

DIRECTIONS:

1. (25 points, 5 each) For each of the following assertions:
   - (3 points) State whether they are True, False, or Unknown according to our current state of knowledge of complexity theory, as discussed in class.
   - (2 points) Briefly justify your answer.

   i) $\mathbf{P} \neq \mathbf{NP}$.
   **Solution:** It is an open problem.

   ii) There is a function $f : \{0,1\}^* \to \{0,1\}$ that cannot be computed by any Turing machine.
   **Solution:** True. The halting problem is an example.

   iii) For every $n$, there is a function $f : \{0,1\}^n \to \{0,1\}$ that cannot be computed by any boolean circuit.

   **Solution:** False. All Boolean functions on $n$ bits can be computed by circuits of size $O(2^n/n)$.

   iv) For $n$ large enough, it holds that there is a function $f : \{0,1\}^n \to \{0,1\}$ that can be computed by a circuit of size $2^{\sqrt{n}}$, but cannot be computed by a circuit of size $n^2$.

   **Solution:** True. Consider the set of all Boolean functions that depend only on the first $\sqrt{n}$ bits. All these functions can be computed by circuits of size $2^{\sqrt{n}}$. By the standard counting argument we know that there is a function that requires size at least $2^{\sqrt{n}}/3\sqrt{n} > n^2$ for large enough $n$.

   v) If $\mathbf{P} \neq \mathbf{NP}$, then any algorithm for solving the independent set problem must take time at least $2^n$ on graphs with $n$ vertices.

   **Solution:** Open or not sufficient to conclude or false. $\mathbf{P} \neq \mathbf{NP}$ only implies that there is no polynomial time algorithm for the independent set problem and does not imply that it has to be exponential.

2. (25 points, 5 each) For each of the following assertions:
   - (3 points) State whether they are True, False, or Unknown according to our current state of knowledge of complexity theory, as discussed in class.
   - (2 points) Briefly justify your answer.

i) Given the code of a Turing machine $\alpha$ and an input $x$, there is a machine $M(\alpha, x)$ that outputs 1 if the machine corresponding to $\alpha$ halts on input $x$ within $2^{|x|}$ steps, and outputs 0 otherwise.

**Solution:** True. The Universal Turing machine along with a counter for the number of steps taken will give the desired machine. This machine simulates $\alpha$ only for the required number of steps.

ii) Every function $f : \{0,1\}^n \to \{0,1\}$ can be computed by a circuit of size $O(2^n/n^2)$.

**Solution:** False. We know that there is a function on $n$ bits that requires at least $2^n/3n$ size circuit to compute it.

iii) If there is a deterministic logspace algorithm to check whether or not two vertices in a directed graph are connected, then $\mathbf{L} = \mathbf{NL}$.

**Solution:** True. This is because we can use this algorithm to check if there is a path from the start node to the output node in the configuration graph corresponding to the **NL** machine in log space.

iv) There is a non-deterministic logspace algorithm that takes as input directed graph, a vertex $s$ and a number $k$, and either aborts, or outputs all the vertices at distance $k$ from $s$ in the graph.

**Solution:** True. The desired algorithm was used in the proof that $\mathbf{NL} = co\mathbf{NL}$.

v) There is a function in **BPP** that is not in **PSPACE**.

**Solution:** False. This is because $\mathbf{BPP} \subseteq \mathbf{PSPACE}$.

3. (25 points, 5 each) For each of the following assertions:

- (3 points) State whether they are True, False, or Unknown according to our current state of knowledge of complexity theory, as discussed in class.
- (2 points) Briefly justify your answer.

i) $\mathbf{PSPACE} \subseteq \mathbf{EXP}$.

**Solution:** True. It follows from $DSPACE(s(n)) \subseteq DTIME(2^{O(s(n))})$ with $s(n)$ to be a polynomial in $n$.

ii) $co\mathbf{NL} \neq \mathbf{PSPACE}$.

**Solution:** True. We know that $co\mathbf{NL} = \mathbf{NL}$ and $\mathbf{NL} \subseteq DSPACE(\log^2 n)$. By the space hierarchy theorem we can conclude that $DSPACE(\log^2 n)$ is a strict subset of **PSPACE**.

iii) If $\mathbf{P} = \mathbf{NP}$, then $\mathbf{P}^{\mathsf{3SAT}} \subseteq \mathbf{P}$.

**Solution:** True. $\mathbf{P}^{\mathsf{3SAT}}$ makes polynomial number of oracles calls. since each oracle call is in **P**, the overall algorithm is polynomial time.

iv) Every arithmetic circuit of size $s$ computing a polynomial of degree $d$ can be simulated by a circuit of depth $O((\log(s) + \log d) \log d)$.

**Solution:** True, we saw this in the last class.

v) The problem of determining whether or not a graph can be colored with 3 colors is **NP**-complete.

**Solution:** True. This was proved in class.

4. (25 points, 5 each) For each of the following assertions:

- (3 points) State whether they are True, False, or Unknown according to our current state of knowledge of complexity theory, as discussed in class.
- (2 points) Briefly justify your answer.

i) There is a randomized **RP** algorithm to check whether or not $AB = C$ for three $n$ by $n$ matrices $A, B, C$, that runs in time $O(n^2)$.

**Solution**: True, we saw this in class.

ii) co**RP** $\subseteq$ co**NP**.

**Solution:** True. The RP algorithm that takes as input randomness and input can be used as a verifier. This shows that $\mathbf{RP} \subseteq \mathbf{NP}$, which implies that $co\mathbf{RP} \subseteq co\mathbf{NP}$.

iii) $co\mathbf{RP} \subseteq \mathbf{BPP}$.

**Solution:** True. This follows from the definition of both complexity classes in a straightforward manner.

iv) The class **RP** remains the same if the error probability is made $2^{-n}$ in the definition. (Here, as usual, $n$ is the length of the input.)

**Solution:** True. You can repeatedly run the algorithm $n$ times to make the probability of error smaller and smaller.

v) Every function $f : \{0,1\}^* \to \{0,1\}$ that is computed by a Turing machine can also be computed by a polynomial sized family of circuits.

**Solution:** False. There is a function that is computable by a Turing machine which uses exponential space but cannot be computed by polynomial sized circuits. The Turing machine can brute force search over all functions to find one that cannot be computed by small circuits, and then run that function.

5. (25 points, 5 each) For each of the following assertions:

- (3 points) State whether they are True, False, or Unknown according to our current state of knowledge of complexity theory, as discussed in class.
- (2 points) Briefly justify your answer.

i) If the permanent can be computed in polynomial time, then $co\mathbf{NP} = \mathbf{NP}$.

**Solution:** True. Permanent is complete for the class $\#\mathbf{P}$. $\mathbf{NP}, co\mathbf{NP} \subseteq \#\mathbf{P}$. Therefore, if permanent is computed in polynomial time, then $\mathbf{NP} = co\mathbf{NP}$.

ii) There is an algorithm that can take an undirected graph and two vertices $s, t$ as input and output whether or not there is a path between $s$ and $t$ in $O(\log^2 n)$ space.

**Solution:** True. This is Savitchs algorithm.

iii) $\mathbf{NL} = co\mathbf{NL}$.

**Solution:** True. This statement was proved in class.

iv) A non-zero multivariate polynomial of degree $d$ can have at most a finite number of roots.

**Solution:** False. $f(x, y) = x - y$ has infinitely many roots.

v) A non-zero univariate polynomial of degree $d$ can have at most $d$ roots.

**Solution:** True. This is the fundamental theorem of algebra.

6. (25 points, 5 each) For each of the following assertions:

- (3 points) State whether they are True, False, or Unknown according to our current state of knowledge of complexity theory, as discussed in class.

- (2 points) Briefly justify your answer.

i) There is a function in **IP** which is not computable by polynomial sized circuits.

   **Solution:** This is an open question. If this statement is true, it will imply that $\mathbf{P} \neq \mathbf{PSPACE}$. If this statement is false, it has other interesting consequences to complexity classes that were not discussed in this class.

ii) If the permanent requires arithmetic circuits of size $2^{\Omega(n)}$, then $P \neq PSPACE$.

   **Solution:** True. the permanent can be computed in **PSPACE**, and **P** can be simulated by polynomial sized circuits.

iii) In the definition of **IP**, if the verifier is restricted to being deterministic, then the class becomes the same as **NP**.

   **True**. If the verifier is deterministic, then the prover might as well send the entire transcript of the conversation to the verifier in a single step. This makes the verifier a valid NP verifier.

iv) For any oracle $A$, there is a function $f : \{0,1\}^* \to \{0,1\}$ that cannot be computed in $\mathbf{P}^A$.

   **Solution:** True. In the midterm you showed that there is a function that cannot be computed by any Turing machine with oracle access to $A$.

v) If $\mathbf{P} = \mathbf{NP}$, then $\mathsf{TQBF} \in \mathbf{P}$.

   **Solution**: Unknown or false. There is no known way to use this to get an algorithm for TQBF.