Homework 10 – Due Tuesday, December 8th, 2020 before 2:00PM

Reminder Collaboration is permitted, but you must write the solutions *by yourself without assistance*, and be ready to explain them orally to the course staff if asked. You must also identify your collaborators. Getting solutions from outside sources such as the Web or students not enrolled in the class is strictly forbidden.

Exercises Please practice on exercises and solved problems in Chapter 7 and 8, and on problem 7.20, 7.23, 7.33, 8.1, 8.4, 8.6. The material they cover may appear on exams. The material they cover may appear on exams.

Problems There are 4 mandatory problems and two bonus problems. Again, collaboration is *allowed* on both mandatory and bonus problems.

1. (**Hamiltonian Path**) Read the reduction from *3SAT* to *HAMPATH* on page 314 of Sipser.

   (a) (**2 points**) A **2cnf-formula** is an AND of clauses, where each clause is an OR of at most two literals. Let 2SAT = $\{\langle\phi\rangle \mid \phi$ is a satisfiable 2cnf-formula$\}$. Is this construction also a valid polynomial-time reduction from *2SAT* to *HAMPATH*?

   (b) (**6 points**) Draw the graph G that the reduction outputs on input formula $\phi = (\bar{x} \vee y) \wedge (x \vee \bar{y})$. For both satisfying assignments of ϕ, give a corresponding Hamiltonian path in G.

   (c) (**8 points**) Draw the graph G that the reduction outputs on input formula $\phi = (x \vee y) \wedge (\bar{x} \vee \bar{y}) \wedge (\bar{x} \vee y) \wedge (x \vee \bar{y})$. Argue that G does not have a Hamiltonian path (not relying on the fact that we already proved that the reduction is correct).

   (d) (**Bonus**) Show that 2SAT $\in$ P.
   Hint: The clause $(x \vee y)$ *is logically equivalent to each of the expressions* $(\overline{x} \to y)$ *and* $(\overline{y} \to x)$. Use a graph to represent the formula.

   (e) (**4 points**) Why would a polynomial-time reduction from *HAMPATH* to *2SAT* have surprising implications, but a reduction in the other direction does not?

2. (**Systems of linear inequalities**) A *linear inequality* over variables $x_1, \ldots, x_k$ is an inequality of the form $c_1x_1 + \cdots + c_kx_k \leq b$, where $c_1, \ldots, c_k$ and b are integers. E.g., $5x_1 - 3x_2 + x_3 \leq -1$ is a linear inequality. A *system of linear inequalities* is a set of inequalities over the same variables. Such a system *has an integer solution* if one can assign integer values to all variables in such a way that all inequalities are satisfied.

   (a) (**4 points**) Formulate as a language LE the problem of deciding whether a given system of linear inequalities has an integer solution.

   (b) (**12 points**) Prove that LE is NP-complete. You may use the following fact without proof: LE $\in$ NP, and the witness for the TM is just an assignment of the variables (in particular, you do not need to prove that there exists a polynomial sized witness).

   (c) (**12 points**) Suppose P = NP, prove that there is some *polynomial-time* Turing machine S, such that on input any integer linear program, outputs (meaning writing to a special output tape) either $\perp$ if there is not a solution, or a solution $x_1, ..., x_k$ to the program.

(d) (**12 points**) An *integer linear program* over variables $x_1, ..., x_k$ consists of a system of linear inequalities (constraints) and an objective function $c_1x_1+\cdots+c_kx_k$. Suppose P = NP, prove that there is some *polynomial-time* Turing machine M, that on input any integer linear program, outputs either $-\infty$ if there is not a solution, $+\infty$ if the program is unbounded, or the maximum value of the program $\max c_1x_1 + \cdots + c_kx_k$. You may use the following fact without proof: for any integer linear program P with value $v \in \mathbb{Z} \cup \{+\infty\}$, there is a polynomial-time computable "dual" integer linear program $D(P)$ with value exactly $-v$.

3. (**coNP**) The complexity class coNP is defined as $\{\overline{L}|L \in \text{NP}\}$.

   (a) (**4 points**) Prove that P $\subseteq$ NP $\cap$ coNP.

   (b) (**4 points**) Prove that NP $\cap$ coNP is closed under complement.

   (c) (**4 points**) Prove that NP is closed under complement if and only if NP = coNP.

   (d) (**8 points**) Prove that if some language $L \in$ coNP is NP-hard, then NP = coNP.

4. (**MIN-FORMULA**) Two Boolean formulas are ***equivalent*** if they have the same set of variables and are true on the same set of assignments to those variables (in other words, they describe the same Boolean function). For example, $\overline{x} \wedge \overline{y}$ and $\overline{x \vee y}$ are equivalent. A Boolean formula is ***minimal*** if no shorter Boolean formula is equivalent to it. Let MIN-FORMULA=$\{\langle\phi\rangle \mid \phi$ is a minimal Boolean formula$\}$.

   (a) (**10 points**) Show that MIN-FORMULA is in PSPACE.

   (b) (**10 points**) Your friend wants to convince you that $\overline{\text{MIN-FORMULA}}$ is in NP, i.e. MIN-FORMULA is in coNP.

   She says that if $\phi \in \overline{\text{MIN-FORMULA}}$ then there exists a formula ψ shorter than ϕ that is equivalent to ϕ. An NTM can verify that $\phi \in \overline{\text{MIN-FORMULA}}$ by guessing ψ.

   Explain why this argument fails to show that $\overline{\text{MIN-FORMULA}}$ in NP.

5* (**Bonus, collaboration is *allowed***) In this problem, we will consider Turing machines with *sublinear* space. In particular, the Turing machine will have two tapes – the first tape holding the input is read-only, the second tape is the working tape and is read-write, and the space used by the Turing machine is the number of cells used on the second tape.

   (a) Consider the language $\{b_k(0)\#b_k(1)\#\cdots\#b_k(2^k-1)|k \geq 0\}$, where $b_k(m)$ is the k-bit binary representation of m. Show that it is not context free, but can be recognized by a Turing machine in time $O(n)$ and space $O(\log\log n)$.

   (b) Show that if L can be recognized by a Turing machine in time $o(n)$, then L is regular.

   (c) Show that if L can be recognized by a Turing machine in space $o(\log\log n)$, then L is regular. *Hint:* Try to see if Savitch's theorem can be useful here.