Theory of Computation

Spring 2024, Homework # 4 Reference Solutions

---

1. As stated in the hint, for any $L \in NP$, there exists a polynomial time TM M and a polynomial $p : N \to N$ such that for every $x \in \{0,1\}^*$, $x \in L \Leftrightarrow \exists u \in \{0,1\}^{p(|x|)}$ s.t. $M(x,u) = 1$. Consider the following reduction f from L to $HALT_{TM}$: "On input w:

   (i) Use $\langle M \rangle$ to construct $M' =$ "On input x:

      (1) Nondeterministically select $u \in \{0,1\}^{p(|x|)}$.

      (2) If $M(x,u) = 1$, accept. Otherwise loop forever. "

   (ii) Output $\langle M', w \rangle$. "

   We can see that $w \in L \Leftrightarrow M'$ halts on w (i.e., $f(w) = \langle M', w \rangle \in HALT_{TM}$). And also M' can be constructed in polynomial time. Therefore $L \leq_p HALT_{TM}$.

   Since $HALT_{TM}$ is not in NP (because $HALT_{TM}$ is undecidable) and $L \leq_p HALT_{TM}$ for any $L \in NP$, $HALT_{TM}$ is NP-hard.

2. Let w^R represent the reversal of w where w is a string. Let L be a language where $L \in NP$.

   (a) Let M be the nondeterministic polynomial time Turing machine that decides L. Construct a NTM $M_R =$ "On input w:

      (i) Nondeterministically generate a string x where $x = w^R$.

      (ii) Run M on x.

      (iii) If M accepts, accept. If M rejects, reject. "

   Since $(w^R)^R = w$, M_R accepts w^R iff M accepts w. So $L(M_R) = (L(M))^R$. Since M is a decider, M_R is also a decider. Finally, since step (i) and (ii) can both be done in polynomial time (w.r.t. $|w|$), M is a polynomial time decider. Therefore $L(M_R) \in NP$.

   (b) Let V be the polynomial time verifier for L. That is, for $w \in L$, V accepts $\langle w, c \rangle$ for some c. Consider
   $V' =$ "On input $\langle w, c \rangle$:

      (i) Simulate V on $\langle w^R, c \rangle$.

      (ii) If V accepts, accept; otherwise reject. "

   For $w \in L$, V accepts $\langle w, c \rangle$ so V' accepts $\langle w^R, c \rangle$ and vice versa. Since w^R can be constructed in polynomial time (w.r.t. $|w|$) and V is a polynomial time verifier, step (i) can be done in polynomial time. So V' is a polynomial time verifier for L^R hence $L^R \in NP$.
   Note that it is okay not to reverse c since it only needs to be a certificate and not necessarily the solution itself.

3. If $A_{TM} \leq_m L_1$, then $\overline{A_{TM}} \leq_m \overline{L_1}$. Since $\overline{A_{TM}}$ is not Turing-recotnizable, if we can prove that $\overline{L_1}$ is Turing-recognizable, then A_{TM} cannobe be many-one reducible to L_1.
Construct a TM $M' =$ "On input $\langle M \rangle$:

   (i) If $\langle M \rangle$ is not a TM, accept.
   (ii) Run M on $\langle M \rangle$.
   (iii) If M accepts $\langle M \rangle$, accept.
   (iv) If M rejects $\langle M \rangle$, reject. "

   Since $\overline{L_1} = \{\langle M \rangle \,|\, (M \text{ is not a TM}) \text{ OR } (M \text{ is a TM and } \langle M \rangle \in L(M))\}$, clearly $L(M') = \overline{L_1}$. Therefore $\overline{L_1}$ is Turing-recognizable.

4. We prove this by showing that $\overline{HALT_{TM}} \leq_m L_2$. Consider $F =$ "On input $\langle M, w \rangle$:

   (i) Use M and w to construct
   $M' =$ "On input x:
      (1) If $\langle M, w \rangle$ does not encode a TM and a string, accept.
      (2) Run M on w for $|x|$ steps.
      (3) If M halts on w within $|x|$ steps, loop forever.
      (4) If M doesn't halt on w within $|x|$ steps, accept. "
   (ii) Output $\langle M' \rangle$. "

   F is clearly computable. We then prove the correctness of our reduction by showing $\langle M, w \rangle \in \overline{HALT_{TM}} \Leftrightarrow \langle M' \rangle \in L_2$.

   - If $\langle M, w \rangle \in \overline{HALT_{TM}}$, then M' accepts all strings. This means M' halts on all palindromes and therefore $\langle M' \rangle \in L_2$.
   - If $\langle M, w \rangle \in HALT_{TM}$, assume M accepts w in n steps. We can find a palindrome p where $|p| \geq n$. Then M' will loop forever on p, hence $\langle M' \rangle \notin L_2$.

5. We prove this by showing $\overline{A_{TM}} \leq_m P$. Consider $F =$ "On input $\langle M, w \rangle$:

   (i) Use M and w to construct
   $M_w =$ "On input x:
      (1) In parallel, run M_{in} on x, M_{out} on x, and M on w (if $\langle M, w \rangle$ does not encode a TM and a string, don't run M and just assume that M does not accept w).
      (2) If M accepts w and M_{out} accepts x, accept.
      (3) If M_{in} accepts x, accept. "
   (ii) Output $\langle M_w \rangle$."

   F is clearly computable. We then prove the correctness of our reduction by showing $\langle M, w \rangle \in \overline{A_{TM}} \Leftrightarrow \langle M_w \rangle \in P$:

   - If $\langle M, w \rangle \in \overline{A_{TM}}$, then $L(M_w) = L(M_{in})$ since M_w can only accept in step (3). Based on the property of P, $\langle M_w \rangle \in P$ since $\langle M_{in} \rangle \in P$.
   - If $\langle M, w \rangle \notin \overline{A_{TM}}$, then $L(M_w) = L(M_{out}) \cup L(M_{in})$. Since $L(M_{in}) \subset L(M_{out})$, $L(M_w) = L(M_{out})$. So $\langle M_w \rangle \notin P$ since $\langle M_{out} \rangle \notin P$.