

COMP 481: Automata, Formal Languages, and Computability

Spring 2009

Solutions to Homework Assignment #9

1. For a string $y = y_1 \dots y_n$, the polynomial DTM builds an $(n + 1) \times (n + 1)$ matrix A , such that $A(i, j) = 1$ if $y_i \dots y_j \in L^*$ and $A(i, j) = 0$ otherwise (for $i \leq j$). The values of A can be computed as

$$A(i, j) = (A(i, i) \wedge (y_{i+1} \dots y_j \in L)) \vee (A(i, i+1) \wedge (y_{i+2} \dots y_j \in L)) \vee \dots \vee (A(i, j-1) \wedge (y_j \in L)).$$

For $i = 0$, we have $A(i, j) = 1$ for any $j \geq i$.

The second term of each conjunction can be decided in polynomial time, since $L \in P$. The first element amounts to an $O(1)$ table lookup. Therefore, A can be computed in $O(n^3 \cdot p)$, where p is a polynomial in n that is the time of the decider of L .

2. Since we assume $P=NP$, then every language $A \in P$ is also NP. Let A' be a language in NP. We show that $A' \leq_p A$. The reduction function $\tau : \Sigma^* \rightarrow \Sigma^*$ is defined as follows: $\tau(w) = x$, for some $x \in A$ if $w \in A'$, and $\tau(w) = z$ for some $z \notin A$ if $w \notin A'$. Since $A \neq \emptyset$ and $A \neq \Sigma^*$, the existence of such strings x and z is guaranteed. Since we assume $P=NP$, there is a polynomial decider for A' , which can be used to check whether $w \in A'$. Therefore, the reduction is computable in polynomial time, and we have

$$w \in A' \text{ iff } \tau(w) \in A.$$

3. Assuming $P=NP$, there is a polynomial decider for SAT; i.e., a polynomial DTM M such that $M(\phi) = 1$ if ϕ is satisfiable and $M(\phi) = 0$ if ϕ is not satisfiable. Using M , we can build a polynomial DTM M' that computes a satisfying assignment for a given satisfiable Boolean formula ϕ . Let x_1, \dots, x_k be the variables in ϕ . We use the notation $\phi|_{x_i=1}$ to denote the Boolean formula resulting from assigning value 1 to the variable x_i and $\phi|_{x_i=0}$ to denote the Boolean formula resulting from assigning value 0 to the variable x_i . Then, M' works as follows on input formula ϕ , returning vector y , with $y(i)$ denoting the truth assignment to variable x_i :

$M'(\text{Input} : \phi)$

- Choose a variable x_i in ϕ .
- If $M(\phi|_{x_i=1}) = 1$,
 - $y(i) = 1$.
 - $M'(\phi|_{x_i=1})$.
- else
 - $y(i) = 0$.

$$- M'(\phi|_{x_i=0}).$$

Verify that this procedure is polynomial (under the assumption that a polynomial decider M exists).

4. It is easy to show that both languages are in NP . To show that they are also NP-hard, we reduce VERTEX-COVER.

(a) FEEDBACK-VERTEX-SET: Let $\langle G = (V, E), k \rangle$ be an instance of VERTEX-COVER.

We construct instance $\langle H = (V', E'), k' \rangle$ for FEEDBACK-VERTEX-SET as follows:

$$V' = V$$

$$E' = \{(u, v), (v, u) : \text{for every undirected edge between } u \text{ and } v \text{ in } E\}$$

$$k' = k.$$

Verify the correctness of this reduction.

(b) DOMINATING-SET: Let $\langle G = (V, E), k \rangle$ (with $E = \{e_1, \dots, e_k\}$) be an instance of VERTEX-COVER. We construct instance $\langle H = (V', E'), k' \rangle$ for DOMINATING-SET as follows:

$$V' = V \cup \{w_1, \dots, w_k\}$$

$$E' = \{e_i, (u, w_i), (w_i, v) : \text{for every undirected edge } e_i \text{ in } E\}$$

$$k' = k.$$

Verify the correctness of this reduction.