W12 L2 $P, NP, \leq_p$

Wednesday, April 22, 2020 9:18

TravelingSalesmanProblem, abbreviated TSP

$$\text{TSP} = \left\{ \langle C, d, b \rangle \;\middle|\; \begin{array}{l} C \text{ is a list of cities} \\ d : C \times C \to \mathbb{N} \text{ gives the distance between cities} \\ \text{There is a round-trip path which visits each} \\ \text{non-home city exactly once, and has} \\ \text{total distance } \leq b. \end{array} \right\}$$

Theorem: $TSP \in NP$.

Proof: (version 1)

T is the certificate/witness/hint

Let's give a verifier V for TSP.

$V =$ "On input $\langle C, d, b, T \rangle$:

1. Check that C is a list of cities, d is a distance function for distances between those cities, $b \in \mathbb{N}$, and T is a list of the same cities as C, with no repeated cities, and $|T| = |C|$.
2. distance =0, current=$T[0]$
3. for each city $i \in T$:
   a. distance = distance + d(current, i)
   b. current = i
4. distance = distance + d(T[last item] , T[first item])
5. If distance $\leq b$, accept. Else, reject."

Need to check:

- does V run in polynomial time in the length of the input?
- is $|T|$ polynomial in the length of the input $\langle C, d, b \rangle$?
- the string $\langle C, d, b \rangle \in TSP$ if and only if there is some T such that $\langle C, d, b, T \rangle \in L(V)$

Proof: (version 2)

Let's give a nondeterministic TM to decide TSP:

$M =$ "On input $\langle C, d, b \rangle$:

1. Check the formatting (if it's bad, reject).
2. Nondeterministically pick an order of cities to visit in C, and track the total distance travelled.
3. Once we run out of new cities to visit, go back to the first city we started at (add this to the total distance travelled).
4. If the total distance travelled was $\leq b$, accept. Else, reject."

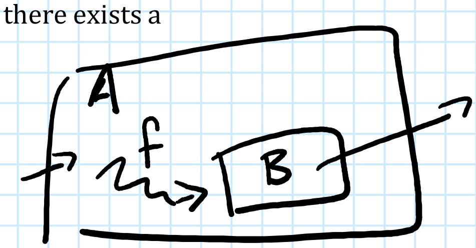
Need to check:

- does M run in polynomial time in the input length?
- is M a decider?
- the string $\langle C, d, b \rangle \in TSP$ if and only if M accepts $\langle C, d, b \rangle$

Def: A function $f: \Sigma^* \to \Sigma^*$ is **polynomial-time computable** if there is some polynomial-time Turing machine that computes it (on input w, this machine halts in time polynomial in $|w|$ with only $f(w)$ on the tape).

Def: For two languages A and B, we say that A is polynomial-time reducible to B if there exists a polynomial-time computable function f such that $w \in A$ if and only if $f(w) \in B$.

<u>Def:</u> For two languages A and B, we say that A is polynomial-time reducible to B if there exists a polynomial-time computable function f such that $w \in A$ if and only if $f(w) \in B$.
We write this as $A \leq_p B$.



Note that this is more restrictive than the mapping reduction $A \leq_m B$.

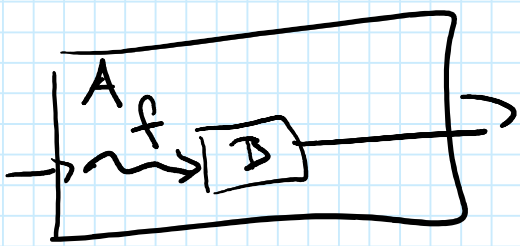
Reminder:
P is the set of languages decidable in polynomial time by a deterministic TM
NP is the set of languages decidable in polynomial time by a nondeterministic TM

<u>Def:</u> A language B is NP −hard if, for every $A \in NP$, we have that $A \leq_p B$.
<u>Def:</u> A language B is NP −complete if B is NP −hard and $B \in NP$.

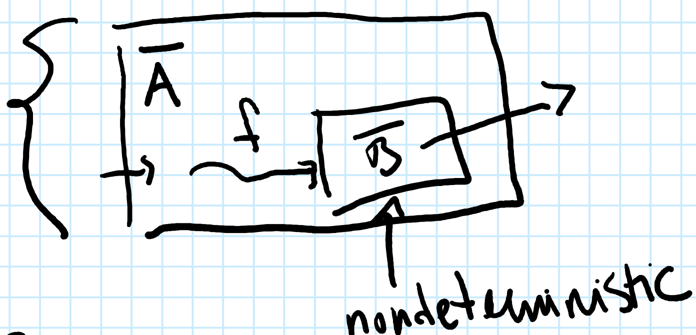
Clicker Q2:

$A \leq_p B$ & $B \in coNP$

$\overline{B} \in NP$



polytime nondet TM



$\overline{A} \in NP$

$A \in coNP$