COT 5310 – Fall 2001 – Homework on Intractable problems [H5], **Solutions**

1.
   - Use Kruskal's algorithm to find a minimum-weight spanning tree when the weight 10 on edge $(1,3)$ of the graph discussed in class is changed to 25.
     Answer: MWST looks like Z
   - Use Kruskal's algorithm to find a minimum-weight spanning tree when the weight 10 on edge $(1,3)$ of the graph discussed in class is changed to 16.
     Answer: MWST looks like |\|.

2. Define the following notions: P, NP, $PSPACE$, $NPSPACE$, NP-complete.
   P is the class of all languages accepted in polynomial time by a deterministic TM.
   NP is the class of all languages accepted in polynomial time by a nondeterministic TM.
   $PSPACE$ is the class of all languages accepted by a polynomial space bounded TM.
   $NPSPACE$ is the class of all languages accepted by a polynomial space bounded TM.
   L is NP-complete if

   - $L \in NP$,
   - $L' \in NP \Rightarrow L' \leq_{ptime} L$.

   Here $\leq_{ptime}$ stands for polynomial time reducibility: that is $L' \leq_{ptime} L$ if there exists a polynomial time algorithm that maps instances of L' into instances of L in such a way that: $w \in L' \Leftrightarrow f(w) \in L'$.

   Show that,

   - If L_1 is NP-complete and if there is a polynomial-time reduction $L_1 \leq_{ptime} L_2$, then L_2 is NP-complete.
     Let L' be any language in NP. Then by definition $L' \leq_{ptime} L_1$. Then since $L_1 \leq_{ptime} L_2$, and since the composition of reductions is a reduction, $L' \leq_{ptime} L_2$.
   - If some NP-complete language L is in P, then $P = NP$.
     Let L' be any language in NP. Then by definition $L' \leq_{ptime} L$. Since $L \in P$, there is a polynomial time algorithm that decides L. The composition of a polynomial-time reduction and the polynomial time algorithm is a polynomial time algorithm. So $L' \in P$.

3. **Closure properties for P.** Show that P is closed under each of the following operations:

   - Reversal.
     Let $L \in P$ and M be a deterministic TM with $L = L(M)$ and time complexity n^k. For any input w, we can build a TM M' that produces the reversal w^r. This machine is polynomially bounded: $\mathcal{O}(n^2)$ when $|w| = n$. Then we input it to M. The combined time complexity is: $\mathcal{O}(n^2 + n^k)$, which is polynomial time.
   - Union.
     Let $L_1, L_2 \in P$ and M_1, M_2 be deterministic TMs with $L_1 = L(M_1)$, $L_2 = L(M_2)$ and time complexities n^{k_1}, n^{k_2}.
     Then we can test if $w \in L_1 \cup L_2$ as follows: first test membership in L_1 and then test membership in L_2. The time complexity is $n^{k_1} + n^{k_2}$, which is polynomial time. Thus $L_1 \cup L_2 \in P$.
   - Concatenation.
     Let $L_1, L_2 \in P$ and M_1, M_2 be deterministic TMs with $L_1 = L(M_1)$, $L_2 = L(M_2)$. Suppose we are given an inpiut $w = w_1 w_2 \cdots w_n$ of length n to check for membership in $L_1 L_2$. For each $i = 0, 1, \ldots, n$ test if $w_1 w_2 \cdots w_i \in L_1$ AND $w_{i+1} \cdots w_n \in L_2$ ($w_0 = \varepsilon$). If so accept, else reject. If the time complexities for L_1, L_2 are n^{k_1}, n^{k_2}, then the overall cost is: $(n+1)(n^{k_1} + n^{k_2})$, which is polynomial time. Thus $L_1 L_2 \in P$.
   - Closure (star).
     Let the input be w of length n. It can be at most in L^n, not more. So we check membership for $L^0, L^1, \cdots, L^n$, i.e., in $\cup_0^n L^i$. We know from above that $L^2 = LL$ is in P. So repeat the argument

to get that $L^3 = L^2L$, ..., L^n are in P. Then use the property that P is closed w.r. to unions. It follows that we can check membership of w in $\cup_0^n L^i$ in polynomial time.

- Complementation.
  Given a polynomial time deterministic TM M for L we can check membership in in L^c, by using the TM M' for which: M' accepts input w iff M' does not accepts input w. The run time is the same (the machines are deterministic),

4. **Closure properties for NP.** Show that NP is closed under each of the following operations:

   - Reversal.
     The argument is essentially the same. First reverse (deterministically), then check membership. The composition is a nondeterministic procedure.
   - Union.
     Identical argument.
   - Concatenation.
     Similar argument. Here we can save some time by guessing nondeterministacally the split. So the overall cost is: $x^{k_1} + x^{k_2}$.
   - Closure (star). Similar argument: again there is some saving in time complexity, from the previous remark.

   [We do not have NP closure for Complementation.]

5. Suppose that there is an NP-complete problem that has a deterministic solution that takes time $\mathcal{O}(n^{\log_2(n)})$. What could you say about the running time of any problem in NP. Explain.
   [Note that this function lies between the polynomials and the exponentials, and is in neither class of functions.]

   Let L be an NP complete language that is in P and let L' be any language in NP.
   A first approach: There must be a reduction $L' \leq_{ptime} L$ because $L \in NP$. Suppose this takes time n^k. Then solve the corresponding problem in L. This takes time $\mathcal{O}(n^{\log_2(n)})$. Combine the two to get: $\mathcal{O}(n^k + n^{\log_2(n)}) = \mathcal{O}(n^{\log_2(n)})$. Problem: the reduction from L' to L may have increased the size of the input from n to m. So we should take $\mathcal{O}(m^{\log_2(m)})$.
   Second approach: Assume that the reduction is accomplished by a polynomial time TM. If the time complexity of this machine is n^k then the size of the input cannot have grown more than that, that is m is at most n^k. Now we get the correct complexity:

   $$\mathcal{O}(n^k + m^{\log_2(m)}) = \mathcal{O}(n^k + n^{k\log_2(n^k)}) = \mathcal{O}(n^{k\log_2(n^k)}) = \mathcal{O}(n^{k^2\log_2(n)}) = \mathcal{O}(n^{c\log_2(n)}),$$

   where $c = k^2$ is a constant.

   [Note that this function lies between the polynomials and the exponentials, and is in neither class of functions.]