Homework 9

Yutong Huang (yxh589)

Problem 1

To prove that a language is NP-complete, we need to prove that it is NP and that every language L can be reduced to it in polynomial time.

Proof. Assume $P = NP$, then $\forall A \in P$ such that $A \neq \phi \ \wedge A \neq \Sigma^* \implies A \in NP$. Then $\exists x \in L \wedge b \notin L$
Let L be an arbitrary language from $NP = P$ such that $L \neq \phi \ \wedge L \neq \Sigma^*$, and there must a decider D_L that decides L in polynomial time.
Consider the following algorithm:

```
function reduction(w: string){
    run D_L on input w;
    if (D_L accepts){
        return x;
    }else {
        return y;
    }
}
```

This algorithm maps words from L to words from A in polynomial time.
Case 1: $w \in L \implies D_L$ accepts $\implies$ `reduction` returns $x \in A$ in polynomial time;
Case 2: $w \notin L \implies D_L$ rejects $\implies$ `reduction` returns $y \notin A$ in polynomial time;
$\therefore \forall L, L \leq_P A \wedge A \in NP \implies A$ is NP-complete. □

Problem 2

We need to prove:

1. $LPATH \in NP$
2. $\forall L \in NP, L \leq_P LPATH$

Proof. $LPATH \in NP$:
Build a verifier for $LPATH$ that runs in polynomial time:
Consider the following algorithm $V(w, c)$, where $w \in \langle G, a, b, k\rangle$ and c is a path:

```
function V(w: <G,a,b,k>, c: path){
    if (c does not contain duplicate nodes && every node in c is also in G){
        if (c.first == a && c.last == b){
            if (c.length >= k){
                return true;
            } else {
                return false;
            }
        } else {
            return false;
        }
    }else {
        return false;
    }
}
```

This algorithm will run in $O(CW)$ time, where $C = |c|,\ W = |w|$, which is polynomial. $\square$

Proof. $\forall L \in NP, L \leq_P LPATH$
Consider the $UHAMPATH = \{\langle G, s, t\rangle|$ G is an undirected graph, s and t are two distinct vertices, and there is a path from s to t in G that passes through each vertex of G exactly once$\}$, which is NP-complete. Then we need to show that $UHAMPATH \leq_P LAPTH$.
Consider the following reduction:

```
function reduction(G: Graph, s: Vertex, t: Vertex){
    let k <- G.vertices.size() - 1;
    return <G, s, t, k>;
}
```

If $G = \langle V, E\rangle$, then this mapping takes $O(|V| + |E|)$ to complete.
Assume that $\langle G, s, t\rangle \in UHAMPATH$,
$\implies$ there is path in G from s to t that passes through each vertex of G exactly once.
then if $k = |V| - 1$, $\langle G, s, t, k\rangle \in LAPTH$.
Assume that $\langle G, a, b, k\rangle \in LAPTH$,
$\implies$ there exists a simple path between a and b.
Because $k = |V| - 1 \implies$ this path must pass through all vertices exactly once.
$\implies \langle G, a, b\rangle \in UHAMPATH$
$\therefore \langle G, a, b\rangle \in UHAMPATH \iff \langle G, a, b, k\rangle \in LAPTH$
$\therefore UHAMPATH \leq_P LAPTH$ $\square$

$\therefore LPATH$ is NP-complete.

Problem 3

1. $SET-SPLITTING \in NP$

   Proof. We can verify if each $C_i \in C$ is monochromatic in $O(|C_i|)$ time,
   $\implies$ we can verify if C contains any monochromatic sets in $O(|C|)$ time.
   $\therefore SET-SPLITTING \in NP$. $\square$

2. $\forall L \in NP, L \leq_P SET-SPLITTING$

   Proof. We can prove this by reducing another NP-complete language to $SET-SPLITTING$: $3SAT$.
   Suppose a formula $F \in 3SAT$, $F = (x_1 \vee x_2 \vee x_3) \wedge \cdots \wedge (x_{n-2} \vee x_{n-1} \vee x_n)$.
   Create a Set S and a set of its subsets C' as follows:

   (a) For each x_i, create $x_i, \overline{x_i}$;

   (b) Then create a set S=$\{x_1, \overline{x_2}, x_2, \overline{x_2}, \ldots x_i, \overline{x_i} \ldots x_n, \overline{x_n}\}$;

   (c) create a variable $f = false$;

   (d) for every clause C_i in F, create a set of 4 variables C'_i containing elementes from C_i and f

   Now we color every true variale in blue and every false one in red. If F is satisfiable, then every clause C_i in F must have at least one blue variable. This means that every C'_i must have at least one of either color.
   $\therefore \langle S, C'\rangle \in SET-SPLITTING$;

   If F is not satisfiable, then at least on of the clause C_i in F evaluates to false. Then we can find these clauses and for each one of them replace one of the x_i with $\overline{x_i}$ to create a new set of clauses C''.
   $\therefore \langle S, C''\rangle \in SET-SPLITTING$

   $\therefore 3SAT \leq_P SET-SPLITTING$

$\therefore \forall L \in NP, L \leq_P SET-SPLITTING$ $\square$