Homework 10

Yutong Huang (yxh589)

Problem 1

Need to prove: $\overline{A} \in$ **co-NP** $\wedge \forall L \in$ **co-NP** $L \leq_P \overline{A}$

Proof. Assume language A is NP-complete $\implies \forall L \in NP, L \leq_P A \ \wedge A \in NP$.
Then we have $\overline{A} \in$ **co-NP** and a verifier `V_a(w, c)` that runs in polynomial time.
Let B be an arbitrary language form **co-NP**. Then $\overline{B} \in NP$ and $\overline{B} \leq_P A$.
Then there exists a verifier `V_b_complement(w, c)` that verifies $\overline{B}$ in polynomial time.

The verifier for B works by inverting the output of `V_b_complement(w, c)`:

```
function V_b(w,c){
    if (V_b_complement(w,c) accepts){
        reject
    } else {
        accept
    }
}
```

Therefore $B \leq_P \overline{B}$. Similarly, $A \leq_P \overline{A}$.
Therefore $B \leq_P \overline{B} \leq_P A \leq_P \overline{A}$.
$\therefore \forall L \in$ **co-NP** $L \leq_P \overline{A}$

$\overline{A} \in$ **co-NP** $\wedge \forall L \in$ **co-NP** $L \leq_P \overline{A} \implies \overline{A}$ is **co-NP**-complete ☐

Problem 2

Proof. Assume a language L is NP-complete and PSPACE-complete.
Therefore $\forall A \in NP, A \leq_P L \wedge \forall B \in PSPACE, B \leq_P L$
Therefore $\forall A \in NP, B \in PSPACE, A \leq_P B$ and $B \leq_P A$
Therefore NP = PSPACE. ☐

Problem 3

Need to prove:

1. $A_{LBA} \in$ PSPACE

   Proof. Construct a Turing-Machine that decides A_{LBA} within $|w|$ space.

Algorithm 1: $TM_{A_{LBA}}(M, w)$

```
initialize M and w;
let i = 0;
let maxIteration = M.states.size() * w.size() * pow(M.tapeAlphabet.size(), w.size());
while i ≤ maxIteration do
    simulate one step of M on w;
    if head exceeds range of [0, |w|] then
        reject;
    if M accept then
        accept;
reject;
```

This TM decides wether $\langle M, w\rangle$ is in M_{LBA} in $O(|w|)$ space.
$\therefore A_{LBA} \in$ PSPACE. □

2. $\forall L \in$ PSPACE, $L \leq_P A_{LBA}$

Proof. Let L be an arbitrary language in PSPACE, then $\exists M$ that decides L in PSPACE.
Assume the maximum space M can use is n^k, where n is size of input, and k is a constant integer.
Suppose w is an input to M, and l is a symbol in the alphabet of L, then we can construct a string by concatenating $l^{|w|^{k-1}}$ to w: $wl^{|w|^{k-1}}$, such that the size of this string is $|wl^{|w|^{k-1}}| = n^k$.
Then we can reduce M to A_{LBA} by invoking: $TM_{M_{LBA}}(M, wl^{|w|^{k-1}})$.
$\therefore \forall L \in$ PSPACE, $L \leq_P A_{LBA}$ □

$\therefore A_{LBA}$ is PSPACE-complete.