CSE 355: Intro to Theoretical Computer Science
Recitation #13 Solution (20 pts)

For all the following problems, assume $\Sigma = \{0, 1\}$

1. [5 pts] Given a regular expression R, we consider the problem of deciding whether the language it generated $L(R)$ contains at least one string ω that has 000 as a substring (*i.e.* $\omega = x000y$ for some x and y). Express this problem as a language denoted as SUB_{R-000} and prove that SUB_{R-000} is decidable.

$\boldsymbol{SUB_{R-000}} = \{<\boldsymbol{R}>|\boldsymbol{R\ is\ a\ RE\ that\ generates\ at\ least\ one\ string\ that\ has\ 000\ as\ a\ substring}\}$

Proof-by-construction:

TM Z = "on input < R >"

1. Applies the algorithm we learned in class to get a DFA D from the regular expression R.
2. Applies the algorithm we learned in class to get a DFA E from regular expression $(0 \cup 1)^*000(0 \cup 1)^*$.
3. Apply the cross-product construction/algorithm to get a DFA F such that $L(F) = L(D) \cap L(E)$. (Note: you can only do this because regular languages are closed under intersection)
4. Run Turing decider T (for E_{DFA} problem) on < F >
   If T accepts, Z REJECT
   If T rejects, Z ACCEPT

2. [5 pts] Given a context free grammar G, we consider the problem of deciding whether G generates any strings in the form of 0^*1^*. Express this problem as a language denoted as $ALL_{CFG-0*1*}$ and prove that $ALL_{CFG-0*1*}$ is decidable.

$\boldsymbol{ALL_{CFG-0*1*}} = \{<\boldsymbol{G}>|\boldsymbol{G\ is\ a\ CFG\ that\ generates\ any\ strings\ in\ the\ form\ of\ 0^*1^*}\}$

Proof-by-construction:

TM Z = "on input < G >"

1. Let regular expression $R = 0^*1^*$, then $L(R)$ must be regular.
2. Let $L' = L(G) \cap L(R)$. Since $L(G)$ is CFL and $L(R)$ is regular, then L' must be context free, *i.e.* there must exist a CFG G' generates L'.
3. Run Turing decider R (for E_{CFG} problem) on $< G' >$
   If R accepts, Z REJECT
   If R rejects, Z ACCEPT

3. [5 pts] Given a context free grammar G, we consider the problem of deciding whether G generates all binary strings or not (Σ^*). Express this problem as a language denoted as ALL_{CFG} and prove that ALL_{CFG} is undecidable.

$\boldsymbol{ALL_{CFG}} = \{<\boldsymbol{G}>|\boldsymbol{G\ is\ a\ CFG\ and\ L(G)} = \boldsymbol{\Sigma^*}\}$

Note: This is the first problem proved to be undecidable related with a context free language. The proof is inside the textbook. We will give everyone the points on this question.

4. [5 pts] Given a Turing machine M, we consider the problem of deciding whether the language M accepts is a context free language or not. Express this problem as a language denoted as CFL_{TM}, and prove that CFL_{TM} is undecidable.

$\boldsymbol{CFL_{TM} = \{< M > | M\ is\ a\ TM\ and\ L(M)\ is\ context\ free\}}$

Note: we will prove that CFL_{TM} is undecidable by mapping reduction from A_{TM}. *i.e.*

$$A_{TM} \leq_m CFL_{TM}$$

Proof-by-contradiction:

Step #1: assume CFL_{TM} is decidable, then there must exist a Turing decider R that decides it. More specifically, given a Turing machine X, R accepts if and only if L(X) is context free; and R rejects if and only if L(X) is not context free.

Step #2: we will build a Turing decider S for A_{TM} problem by using R as a sub-routine.

TM S = "on input < M, ω >"

1. Construct the following auxiliary Turing machine M'
   - M'= "On input x"
     - 1.1 if x has the form of $\omega\omega$, M' accepts.
     - 1.2 if x does not have the form of $\omega\omega$, simulate M's running on ω, M' accepts if M accepts ω.
2. Run Turing decider R (for CFL_{TM} problem in step #1) on < M' >
   - If R accepts, S ACCEPT
   - If R rejects, S REJECT

Step #3: we know A_{TM} is undecidable, but from step #2, we successfully build a decider for it, it means something must be wrong with our assumption in step #1, i.e. CFL_{TM} cannot be decidable and it is undecidable.