(2 points) Question 1. Let $L$ be a language over the alphabet $\Sigma = \{0, 1\}$. Let $u \in \Sigma^*$ be a word.

Consider the following “algorithm” $A$ which is defined in terms of $L$ and $u$:

INPUT: $w$.

- If $u \in L$, ACCEPT.
- If $u \notin L$, REJECT.

Note that $L$ is not necessarily a decidable language. In fact, $L$ can be any language.

(a) Can $A$ be considered an algorithm in the sense that there is a Turing machine that is equivalent to it?

(b) What is $L(A)$? Is $L(A)$ decidable?

Here $L(A)$ is defined as the language $\{w \mid A$ accepts $w\}$.

Solutions for Question 1. For question (a), $A$ cannot be considered as algorithm, because we cannot guarantee that the step $u \in L$ can be performed by TM.

However, we can define a TM $B$ that works according to $u$ and $L$ as follows. If $u \in L$, define $B$ as follows.

INPUT: $w$

- ACCEPT.

If $u \notin L$, define $B$ as follows.

INPUT: $w$

- REJECT.

In the first case $L(B) = \Sigma^*$, while in the second case $L(B) = \emptyset$.

For question (b), by definition, $L(A)$ is either $\Sigma^*$ or $\emptyset$. Both of which are decidable.
(2 points) Question 2. Recall the language $\text{HALT} := \{\langle M \rangle\$w \mid M \text{ accepts } w\}$. Let $L$ be a decidable language. Prove that $L \leq_m \text{HALT}$.

Solution for question 2. If $L$ is decidable language, there is a DTM $M$ that decides $L$.

Define the function $F$ as follows.

$$F(w) = \langle M \rangle\$w$$

Note that by definition, $w \in L$ if and only if $M$ accepts $w$ if and only if $\langle M \rangle\$w \in \text{HALT}$. Thus, the reduction is correct.

It is rather straightforward that $F$ is computable. Algorithmically, it works as follows. On input $w$, append the string $M\$ in front of $w$ and output it.

If you insist on TM, the following is acceptable. We define a 2-tape TM $A$ for $F$ that works as follows. On input $w$:

- Write the input $w$ onto tape-2.
- Write the word $\langle M \rangle\$ onto tape-1, and after that copy $w$ back to tape-1.

Here $w$ is written immediately after $\langle M \rangle\$.

- Output the string on tape-1.

NOTE: Here we may not know exactly the TM $M$, but such TM exists. Thus, there is also a TM $A$ that works as above.
(2 points) Question 3. Consider the following problem.

<table>
<thead>
<tr>
<th>CFL-REG</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input:</strong> A CFG $G$ and a DFA $A$</td>
</tr>
<tr>
<td><strong>Task:</strong> Output True, if $L(G) = L(A)$. Otherwise, output False.</td>
</tr>
</tbody>
</table>

Prove that CFL-REG is undecidable.

**Solution for Question 3.** We will show that CFL-UNIVERSALITY $\leq_m$ CFL-REG.

Let $A$ be a DFA such that $L(A) = \Sigma^*$. Define the following computable function $F$. On input CFG $G$:

- Output $(G, A)$.

For every CFG $G$, the following holds. $G \in$ CFL-UNIVERSALITY if and only if $L(G) = \Sigma^*$ if and only if $L(G) = L(A)$ if and only if $(G, A) \in$ CFL-REG.
(2 points) Question 4. Prove that the class \textbf{coNP} is closed under union and intersection. That is, prove the following.

- If \( L_1, L_2 \in \text{coNP} \), then \( L_1 \cup L_2 \in \text{coNP} \).
- If \( L_1, L_2 \in \text{coNP} \), then \( L_1 \cap L_2 \in \text{coNP} \).

Hint: Use the definition of \textbf{coNP} and de Morgan’s law.

Solution for Question 4. Suppose \( L_1, L_2 \in \text{coNP} \). By definition, \( \Sigma^* - L_1, \Sigma^* - L_2 \in \text{NP} \). Consider the following identity. (de Morgan’s law)

\[
\begin{align*}
\Sigma^* - (L_1 \cup L_2) &= (\Sigma^* - L_1) \cap (\Sigma^* - L_2) \\
\Sigma^* - (L_1 \cap L_2) &= (\Sigma^* - L_1) \cup (\Sigma^* - L_2)
\end{align*}
\]

Since \textbf{NP} is closed under union and intersection, both \((\Sigma^* - L_1) \cap (\Sigma^* - L_2)\) and \((\Sigma^* - L_1) \cup (\Sigma^* - L_2)\) are in \textbf{NP}. Therefore, both \( \Sigma^* - (L_1 \cup L_2) \) and \( \Sigma^* - (L_1 \cap L_2) \) are in \textbf{NP}, which by definition, both \( L_1 \cup L_2 \) and \( L_1 \cap L_2 \) are in \textbf{coNP}. 
(2 points) Question 5. Consider the following problem.

<table>
<thead>
<tr>
<th>CFL-Reversal</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input:</strong> A CFG $G$.</td>
</tr>
<tr>
<td><strong>Task:</strong> Output True, if $L(G) = L(G)^r$, i.e., $L(G)$ is closed under reversal. Otherwise, output False.</td>
</tr>
</tbody>
</table>

Prove that CFL-Reversal is undecidable.

Note: $L(G)$ is closed under reversal, if for every word $w \in L(G)$, $w^r \in L(G)$, where $w^r$ denotes the reverse of $w$.

Solution for Question 5. We will show that CFL-UNIVERSALITY $\leq_m$ CFL-Reversal.

Define the following computable function $F$. On input CFG $G = \langle \Sigma, V, R, S \rangle$:

- Construct the following $G' = \langle \Sigma', V', R', S' \rangle$, where:
  \[
  \Sigma' = \Sigma \cup \{ \$ \} \\
  V' = V \cup \{ X \} \\
  R' = R \cup \{ X \to aX : a \in \Sigma \} \cup \{ X \to \epsilon \} \cup \{ S' \to S\$X \} \\
  S' = \text{the start variable}
  \]

  Here $\$ is a new symbol not in $\Sigma$ and $X$ is a new variable not in $V$.

- Output $G'$.

Proof that $F$ is computable: We can write a computer program that generates the set:

\[
\{ X \to aX : a \in \Sigma \} \cup \{ X \to \epsilon \} \cup \{ S' \to S\$X \}
\]

Thus, the CFG $G'$ can be constructed algorithmically (or by a TM).

Note that in your solution, you should at least explicitly write down the rules for the CFG $G'$. There is a -0.5 point, if you write $G'$ in terms of language like: Let $G'$ be such that $L(G') = \{ u\$v : u \in G \text{ and } v \in \Sigma^* \}$. Or worse, you write like: Let $G' = G\$\Sigma^*$ (Though I understand what you are trying to say, what does $G\$\Sigma^*$ even mean???)

If you write like that, it is not that clear to “me” whether you understand your own solution, or whether you know how to write a program to do it algorithmically, or you just copy from the internet. You should know that the majority of your solutions are almost identical, so I have my doubt here. Note that in the final exam, the grading will be more strict.

Proof that the reduction is correct: Note that the language $L(G')$ contains all the words of the form:

\[
u\$v \quad \text{where } u \in L(G) \text{ and } v \in \Sigma^*
\]

Thus, $L(G) = \Sigma^*$ if and only if $L(G')$ is closed under reversal.