CSE 2001: Introduction to the theory of computation
Assignment 3 Solutions

Question 1

Suppose L is a regular language. Consider the language L' consisting of all subwords of L. A subword of a word w is a word obtained by deleting zero or more characters from the beginning and deleting zero or more characters from the end of w. Is L' regular? Prove your answer.
Solution: Assume that there is one accept state (if the given FA does not, we can use the standard method of converting to a NFA with 1 accept state). Add an epsilon transition from the start state to every other state and from each state to the accept state, eliminating duplicates.
If the original DFA on input $w = a_1a_2 \dots a_k$ is accepted after the DFA goes through states $s_0, s_1, \dots, s_k$, then for a subword $w' = a_i \dots a_j$, where $1 \leq i \leq j \leq k$, the new NFA goes through the exact same transitions by using the newly added epsilon-transitions where needed.

Question 2

Given a language L, define $L_1 = \{vw | v \in L, w \notin L\}$. Prove that if L is regular, so is L_1. Prove also that the converse is not true.
Solution:
L_1 is equal to the concatenation of L and $\overline{L}$. The latter is regular because regular languages are closed under complementation. The concatenation of two languages is regular. So L_1 is regular if L is regular.
For the converse, consider $L = \{1^p | p \text{ prime}\} \cup \{1\}$. It was proved in the tutorials that $\{1^p | p \text{ prime}\}$ is not regular. The same proof holds even when the string 1 is added to that set.
We claim that $L_1 = \{1^k | k \in \mathbb{N}, k > 0\} - \{1111\}$. In other words, for any positive integer $k \neq 4$, $1^k \in L_1$. To prove this, we consider two cases.
Case 1: $k \geq 5$ is an odd number. Then $k - 1$ is even. Since every even number greater than 2 is not a prime, $1^k \in L_1$. The number 5 can be written as
Case 2: $k \geq 5$ is an even number. In this case $k - 2$ is even and not prime. So $1^k \in L_1$.
Case 3: $k = 1, 2, 3$. $1^k \in L_1$ for these k, because $1 = 1 + 0, 2 = 2 + 0, 3 = 3 + 0$.
We cannot break 1111 into 2 parts such that one part is in L and the other part is not.
It is not hard to see that L_1 is regular. A regular expression for this set is $1 \cup 11 \cup 111 \cup 111111^*$.
Thus we have shown that there exists an L that is not regular but for which $L \cdot L^c$ is regular.

Question 3

Prove that the languages $\{a^m b^n | n \geq m \geq 0\}$ and $\{a^m b^n | 0 \leq n \leq m\}$ are not regular. Infer from these results that the union of two non-regular languages may be regular.
Solution: The standard pumping lemma based arguments work for each of the given sets. Let us assume that $L_1 = \{a^m b^n | n \geq m \geq 0\}$ is regular. Then there is a pumping length p. Let us choose string $s = xyz = a^p b^p \in L_1$. From condition 3 of the Pumping Lemma we know that y will consist only of a's. If we pump up, we increase the number of a's only, and the string created has more a's that b's. This is a contradiction. Therefore L_1 is not regular.
Similarly, Let us assume that $L_2 = \{a^m b^n | m \geq n \geq 0\}$ is regular. Then there is a pumping length q. Let us choose string $s = xyz = a^q b^q \in L_2$. From condition 3 of the Pumping Lemma we know that y

will consist only of a's. If we pump down, we decrease the number of a's only, and the string created has fewer a's that b's. This is a contradiction. Therefore L_2 is not regular.
Note that $L_1 \cup L_2 = a^*b^*$ which is a regular expression and therefore regular.

Question 4

Consider the CFG $S \to \epsilon | aSb | SS$. Prove that the words generated by this language satisfy the following conditions:

1. The number of a's and b's are equal, and
2. In each prefix, the number of a's is no less than the number of b's.

Solution: The two parts are done separately here for clarity. They can be combined into one.
Part 1.
This is a simple proof by strong induction on the length of the derived string s. Let us assume that of all possible derivations for a given string we choose the shortest one. For the base case, $|s| = 0$: ϵ satisfies the condition.
For the inductive step, assume that all strings with lengths less than or equal to k satisfy the condition. For a string of length $k+1$, let us consider the first rule in the derivation. If the rule was $S \to aSb$, then the S on the right hand side inductively derived a string of length $k-1$ with equal number of a, b's. Adding an a and a b preserves this condition.
If the first rule applied was $S \to SS$ then either each S on the right produced a string of length k or smaller, or one of the S's were expanded using the $S \to \epsilon$ rule. In the former case, each of these strings have the same number of a, b's and so their concatenation also has an equal number of a, b's. The latter case contradicts the assumption that the derivation is the smallest possible, because this step can be eliminated from the derivation without changing the outcome.
Part 2
Consider any prefix of a string s produced by this grammar. Again, we assume that we choose the shortest derivation of s. We prove the statement using strong induction on the length of s. The base case $s = \epsilon$ is trivial. For the inductive step, assume that all strings with lengths less than or equal to k satisfy the condition. For a string of length $k+1$, let us consider the first rule in the derivation. If the rule was string of length $k-1$ in which any prefix has no more b's than a's. Adding an a at the beginning and a b at the end preserves this condition.
If the first rule applied was $S \to SS$ then either each S on the right produced a string of length k or smaller, or one of the S's were expanded using the $S \to \epsilon$ rule. In the former case, each prefix of each of these strings have the at least as many a's as b's and so their concatenation preserves the given condition.
The latter case contradicts the assumption that the derivation is the smallest possible, because this step can be eliminated from the derivation without changing the outcome.

Question 5

Is the language $\{ww^Rw | w \in \{a, b\}^*\}$ context-free? Prove your answer.
Solution: We assume that the language is context-free. That implies we are given a pumping length p. Let us choose $w = a^pb^p$, so $ww^Rw = a^pb^{2p}a^{2p}b^p = uvxyz$, where $|vxy| \leq p$. Note that the format ww^Rw implies that each third has the same number of a's (and thus b's). This in turn implies that the number of a's in each third is half that in the remaining two thirds combined.

If v, y are both a's and we pump up, the number of a's in the first third i s no longer half of the number of a's in the latter two thirds. The same thing happens if If v, y are both b's.
If v or y contains both a, b's then pumping up creates more alternations of a, b's in one of the thirds compared to the other two thirds, and this destroy s the $ww^R w$ format.
Therefore the given language is not context-free.