CS3500 HW3 Solutions

1 (10 points each)

a. Yes.
We know $\overline{L_1} = \{w : w \text{ has an occurrence of } 001 \text{ or } |w| \text{ is odd}\}$. Let $A = \{w : w \text{ has an occurrence of } 010\ \}$ and $B = \{w : |w| \text{ is odd}\}$, then $\overline{L_1} = A \cup B$. Clearly, A and B are both regular. So L_1 is regular since the class of regular languages is closed under union and complement.

b. Yes.
Assume the alphabet Σ is $\{0, 1\}$, then the regular language $A = \Sigma\Sigma(\Sigma\Sigma\Sigma)^* = \{w : |w| = 3n + 2 \text{ for some } n \geq 0\}$ and $B = L(0^*1^*)$. Clearly, $L_2 = A \circ B$ and thus L_2 is also regular.

c. No.
Assume L_3 is regular. Let p be the pumping length and s be the string $0^p1^{2p}0^p$. Because s is a member of L and s has length more than p, the pumping lemma guarantees that s can be split into three pieces, $s = xyz$ satisfying the three conditions of the pumping lemma. Because of condition 3 ($|xy| \leq p$), y can only consist of 0s. Then the string xy^2z will not be in L. This is a contradiction.

d. No.
Assume L_4 is regular. Then its complement $\overline{L_4}$ is also regular. Furthermore, the language $\overline{L_4} \cap \{0^*1^*0^*\}$ is also regular since the class of regular languages is closed under intersection. We know that $\overline{L_4} \cap \{0^*1^*0^*\} = L_3$ is not a regular language from part c) and this is a contradiction.
Note: The complement of L_4 is NOT L_3.

2 (10 points each)

a. No.
Assume L_5 is a CFL. Let p be the pumping length and s be the string $a^pb^pc^{p+1}$. Because s is a member of L_5 and s has length more than p, the pumping lemma guarantees that s can be split into $s = uvxyz$ satisfying the three conditions of the pumping lemma. Because of condition 3 ($|vxy| \leq p$), vxy can only consist of at most two different alphabet symbols.

1. If vxy consists of only a's or only b's, then pumping up will make the number of a's and b's unequal.

2. If vxy consists of only c's, then pumping down will reduce the number of c's to below $p+1$ and it is equal or less than the number of a's and b's.

3. If vxy consists of both a's and b's, then pumping up will make the number of a's or b's exceed the number of c's $p+1$.

4. If vxy consists of both b's and c's, then pumping down will either make the number of b's not equal to a's, or the number of c's equal or less than a's.

b. Yes.
We know that the language $A = \{w : w \text{ contains equal number of } a\text{'s and } b\text{'s}\}$ is a CFL (it is easy to construct a PDA for A), and $B = \{a^*b^*a^*b^*\}$ is a regular language. According to Sipser Problem 2.17 (a) (in HW2), the intersection of a CFL and a regular language is a CFL. So $L_6 = A \cap B$ is a CFL.

c. No.
Assume L_7 is a CFL. Let p be the pumping length and s be the string $a^p b^p a^p b^p$. Because s is a member of L_7 and s has length more than p, the pumping lemma guarantees that s can be split into $s = uvxyz$.

1. If v or y consists of more than one symbol, i.e., both a and b, then uv^2xy^2z will not contain the symbols in the correct order.

2. If v and y both consist of only a's, due to condition 3, they can only lie in the same region of a's (i.e., vxy only contains a's), then pumping up will increase the number of a's in this region causing the two a regions to have different numbers of a's. The same reasoning works if v and y both consist of only b's.

3. If v consists of only a's and y consists of only b's, or v consists of only b's and y consists of only a's, then pumping up will cause the number of a's or b's in one region greater than in the other region. The contradiction occurs.

The above cases show that s cannot be pumped and thus L_7 is not a CFL.

d. Yes.
The following CF grammar generates L_8:
$S \rightarrow S_0 \mid S_0C \mid S_1$
$A \rightarrow \texttt{a}A \mid \texttt{a}$
$B \rightarrow \texttt{b}B \mid \texttt{b}$
$C \rightarrow \texttt{c}C \mid \texttt{c}$
$S_0 \rightarrow \texttt{a}S_0\texttt{b} \mid \texttt{A} \mid \texttt{B}$
$S_1 \rightarrow \texttt{a}S_1\texttt{c} \mid \texttt{AB} \mid \texttt{BC} \mid \texttt{A} \mid \texttt{C}$

3 (10 points)
Hint: We'll construct a 4-tape TM. Two tapes are used for input and one for carryover, one for the result. Reverse the two input integers and put them on the input tapes (one for each tape). Write '0' on the carryover tape. Start from the leftmost symbols. Each time read one symbol from the input tapes and carryover tape, respectively. If there is no '1's, write '0' on both the result tape and the carryover tape. If there is a single '1', write '1' on the result tape and '0' on the carryover tape. If there are two '1's, write '0' on the result tape and '1' on the carryover tape. If all three symbols are '1's, write '1' on both the result tape and the carryover tape. Move the tape heads of both input tapes and the result tape right one position and repeat the steps until both input tapes reach the end. Reverse the number on the result tape to get the final result.

4.2 (10 points)
Define $EQ_{DFA,REX} = \{< D, R > | D$ is a DFA and R is a regular expression, $L(D) = L(R)\}$. We can construct the following TM to decide $EQ_{DFA,REX}$:
$T =$ "On input $< D, R >$:
1. Convert R to an equivalent DFA D'.
2. Run TM F from Theorem 4.5 (EQ_{DFA}) on D and D'. *Accept* if F accepts, otherwise *reject*."

4.3 (10 points)
Since the class of decidable languages are closed under complement, we can just prove the complement of ALL_{DFA} is decidable. Since the complement of ALL_{DFA} is E_{DFA}, which is proved in Sipser to be decidable, ALL_{DFA} must also be decidable.

4.5 (10 points)
Suppose the DFA A has n states. If $L(A)$ is infinite, then A must accept a string w with length $|w| \geq n$, otherwise A can only accept at most 2^n strings. So A must go through at least $n + 1$ states to accept w and thus at least one state must be repeated causing a loop on the path. We can construct the following TM:
$T =$ "On input $< A >$, an encoding of a DFA whose number of states is n:
1. Construct a DFA A' for the language $\{w | w \geq n\}$.
2. Construct a DFA F: $L(F) = L(A) \cap L(A')$.
3. Run the decider for E_{DFA} on $< F >$. *Accept* if it rejects, otherwise *reject*."