Honors Theory of Computation

Problem Set 2 Solutions

Problem 1

Solution:

(a) Suppose that $L_1 = \{0^n 1^m 0^n\}$ is regular. Let p be the pumping length given by the pumping lemma. Let $s = 0^p 1 0^p$. Clearly $s \in L_1$, and $|s| = 2p + 1 > p$. Write $s = xyz$ satisfying $|xy| \leq p$ and $y \neq \varepsilon$. Then $y = 0^k$ for some $k \geq 1$. Thus $xy^2z = 0^\ell 1 0^p$ where $\ell > p$, and hence $xy^2z \notin L_1$, contradicting the pumping lemma. Therefore, L_1 is nonregular.

(b) We instead show that the language of palindromes is not regular (which suffices since the class of regular languages is closed under complementation). Suppose on the contrary that the palindromes form a regular language. Let p be the pumping length given by the pumping lemma. Let $s = 0^p 1 0^p$. Using exactly the same argument as in part (a), we reach a contradiction.

Problem 2

Solution: Note first that a DFA is also an All-Paths-NFA (DFA has exactly one computation on given input) and hence every regular language is accepted by an All-Paths-NFA.
Now we show that if M is an All-Paths-NFA then the language L recognized by M is regular. Let N be a NFA whose transition function is same as that of M, but the accept/non-accept states are switched. Now

$$\begin{aligned} x \in L &\iff M \text{ accepts } x \\ &\iff \text{every computation of } M \text{ on } x \text{ accepts} \\ &\iff \text{no computation of } N \text{ on } x \text{ accepts} \\ &\iff x \notin L(N) \end{aligned}$$

This shows that $L = \overline{L(N)}$, i.e. L is complement of a regular language. Hence L is regular.

Problem 3

Solution: Given a DFA M that accepts A, we construct an NFA M' that accepts $A_{\frac{1}{2}-}$. The basic idea is as follows: to decide whether a string $x \in A_{\frac{1}{2}-}$, we non-deterministically choose y such that $|x| = |y|$. We simulate M on x and at the same time simulate M backwards on string y. The simulation on x starts with the start state of M (call it q_0) whereas the simulation on y starts with some accept state of M (call it q_f). We accept iff both simulations reach the same state of M (call it q). Thus we accept iff x takes the DFA from q_0 to q and y takes it from q to q_f. Since q_f is an accept state of M, we ensure that $xy \in A$. The simultaneous simulation on x and y is carried out by the cartesian product construction, similar to proof of Theorem 1.2 in Sipser's book.

Formally, if $(Q, \Sigma, \delta, q_0, F)$ describes the DFA M, then the NFA $M' = (Q', \Sigma, \delta', q_{start}, F')$ is defined as follows:

- $Q' = Q \times Q \cup \{q_{start}\}$.
- $F' = \{(q, q) \mid q \in Q\}$.

- There is ϵ-move from q_{start} to all the states in $\{(q_0, q_f) \mid q_f \in F\}$. These are the only moves possible from q_{start}.
- There is a move from (q_1, q_2) to (q_3, q_4) on input symbol $a \in \Sigma$ iff $\delta(q_1, a) = q_3$ and $\delta(q_4, b) = q_2$ for some $b \in \Sigma$. Formally,

$$(q_3, q_4) \in \delta'((q_1, q_2), a) \quad \text{iff} \quad \delta(q_1, a) = q_3 \text{ and } \exists\, b \in \Sigma \text{ s.t. } \delta(q_4, b) = q_2$$

Problem 4

1. $\{w : \text{length of } w \text{ is odd}\}$

$$\begin{aligned} S &\rightarrow 1E \mid 0E \\ E &\rightarrow EE \mid 00 \mid 01 \mid 10 \mid 11 \mid \varepsilon \end{aligned}$$

   The PDA has two states q_{start} and q_{accept} and no stack. The state changes for every input symbol read and the PDA accepts if the end state is q_{accept}.

2. $\{w : w \text{ contains more 1's than 0's}\}$

   Solution:

$$\begin{aligned} S &\rightarrow R1R \\ R &\rightarrow RR \mid 0R1 \mid 1R0 \mid 1 \mid \varepsilon \end{aligned}$$

   The PDA scans across the input. If it sees a 1 and its top stack symbol is 0, it pops the stack. Similarly if it sees a 0 and its top stack symbol is 1, it pops the stack. In all other cases, it pushes the input symbol onto the stack. After it scans the input, if there is a 1 on the top of the stack, it accepts. Otherwise, it rejects.

3. $\left\{w : w = w^R\right\}$

   Solution:

$$S \rightarrow 0S0 \mid 1S1 \mid 0 \mid 1 \mid \varepsilon$$

The PDA begins by scanning across the string and pushing the symbols onto the stack. At some point it nondeterministically guesses the midpoint of the string has been reached. It also nondeterministically guesses if the string has even length or odd length. If it guesses even, then it pushes the current symbol it's reading (at the guessed midpoint) onto the stack. If it guesses odd, it goes to the next input symbol without changing the stack. Now it scans the rest of the string, and compares each symbol it scans with the symbol on the top of the stack. If they are different, it rejects. If the stack becomes empty just after it reaches the end of the input, then it accepts. In all other cases it rejects.

Problem 5

Solution: Let A be a context-free language recognized by a PDA M. We will construct a PDA R that recognizes SUFFIX(A). On input v, the PDA R works in two phases. The first phase operates without looking at the input. The PDA non-deterministically generates a symbol $a \in \Sigma$ and simulates (one or more) steps of M until the symbol a is read. The PDA repeats this sequence of moves (as many times as it wishes). It non-deterministically decides when to switch to second phase. In the second phase, the PDA looks at the input v and simulates M on v.

Note that R accepts v if and only if there exists $u \in \Sigma^*$ such that M accepts uv. The string u (and its length!) is "guessed".

Problem 6

(a) $L_1 = \{0^n 1^n 0^n 1^n : n \geq 0\}$.

Solution: Suppose that L_1 were a CFL. Let p be the pumping length given by the pumping lemma. Let $s = 0^p 1^p 0^p 1^p$ and we show that s cannot be pumped. Write $s = uvxyz$ satisfying $|vy| > 0$ and $|vxy| \leq p$. If v or y contains more than one type of symbols, then uv^2xy^2z does not have the symbols in the correct order as it is not of the form $a^i b^j a^k b^\ell$, and thus is not a member of L_1. If both v and y contain at most one type of symbol, then uv^2xy^2z contains runs of 0's and 1's of unequal length, and thus is not a member of L_1. Therefore, s cannot be pumped without violating the pumping lemma conditions, and hence L_1 is not a CFL.

(a) $L_2 = \{0^i 1^j : i \geq 1, j \geq 1, i = jk \text{ for some integer } k\}$

Solution: Suppose L_2 is CFL and let p be the pumping length. Let $s = 0^{200p} 1^{100p}$ and we show that s cannot be pumped. Write $s = uvxyz$ satisfying $|vy| > 0$ and $|vxy| \leq p$. It can be easily seen that $uv^2xy^2z \notin L_2$ (I got tired of writing the proof).