Solutions Problem Set #5

Solution Problem 1

The proof is by contradiction. Assume $L$ is Context Free. Then by the Pumping Lemma for Context Free Languages, there exists a constant $p \geq 1$, such that for any $w \in L$, $|w| \geq p$, $w$ can be written as $vxyuz$, satisfying the following three conditions: (1) $vx^iyu^iz \in L$, $\forall i \geq 0$; (2) $|xu| \neq 0$; (3) $|xyu| \leq p$. Consider $w = 0^p10^{2p}10^{3p}$. $w \in L$ and $|w| \geq p$. We check all possible partitions of $w$ into $vxyuz$ and show that there exists $i \geq 0$, for which condition (1) does not hold:

1. Case: $|v| = 0$
   By condition (3) $x, y, u$ span the first $p$ symbols of $w$, thus $x = 0^\alpha$, $y = 0^\beta$, $u = 0^\gamma$. By condition (2), $\alpha$ and $\gamma$ can not be both 0. Thus $vx^0yu^0z = 0^{p-\alpha-\gamma}10^{2p}10^{3p} \notin L$.

2. Case: $0 < |v| < p$
   - if $1$ is contained in either $x$ or $u$, $(|xu| = \alpha \geq 1)$, then $w_1 = vx^0yu^0z = 0^p0^{2p-\alpha+1}10^{3p} = 0^{3p-\alpha+1} \notin L$. 
   - if $v = 0^\beta, x = 0^\alpha$ and $u = 0^\gamma$ and $y = 0^*10^*$, then $w_1 = vx^0yu^0z = 0^{p-\alpha}10^{2p-\gamma}10^{3p} \notin L$, because by condition (2) $\alpha$ and $\gamma$ can not be both 0, thus $\frac{2p-\alpha-\gamma}{2} \neq p$ or $p - \alpha \neq p$.

3. Case: $p < |v| < 2p$
   The string $w_1 = vx^0yu^0z = 0^p10^{2p-\alpha-\gamma}10^{3p} \notin L$, because by condition (2) $|xu| = \alpha + \gamma \geq 1$, thus $\frac{2p-\alpha-\gamma}{2} \neq p$.

4. Case: $2p < |v| < 3p$
   - If $x$ or $u$ contain the second 1 of $w$, then $vx^0yu^0z = 0^p10^* \notin L$.
   - If $x$ and $u$ do not contain 1, i.e. $x = 0^\alpha$, $u = 0^\gamma$, and $y = 0^*10^*$, then $w_1 = vx^0yu^0z = 0^p10^{2p-\alpha-\gamma}10^{3p} \notin L$ or $w_1 = vx^0yu^0z = 0^p10^{2p-\alpha-\gamma}10^{3p} \notin L$. Because by condition (2) $\alpha$ and $\gamma$ can not be both 0, then either $\frac{2p-\alpha-\gamma}{2} \neq p$, or $\frac{2p-\alpha}{2} \neq p$, or $\frac{3p-\gamma}{3} \neq p$.

This concludes the proof.
Solution Problem 2

- Formal Definition of Automaton with Queue:
  An automaton with Queue is a 6-tuple \((Q, \Sigma, \Gamma, \delta, q_s, q_a, q_r)\), where \(Q\) is a finite set of states, \(\Sigma\) is the input alphabet (also finite), \(\Gamma\) is a finite set of stack symbols, \(q_s, q_a, q_r\) are the initial, accepting, and rejecting state, respectively. The transition function is defined as \(\delta : (Q \times \Gamma \epsilon) \rightarrow (Q \times \Gamma^*)\).

- The set of all configurations is all possible pairs of the form \((Q \times \Gamma^*)\).
- The initial configuration is \((q_s, w)\), meaning the machine is in state \(q_s\), with the input \(w\) enqueued.
- An accepting configuration is \((q_a, x)\), meaning the finite control is at state \(q_a\), and \(x\) is the content of the Queue.
- A rejecting configuration is \((q_r, x)\).

- Definition of the computation relation:
  We say that a configuration \(S_1 = (q_i, a\alpha)\) yields a configuration \(S_2 = (q_j, \alpha x)\), denoted as \(S_1 \implies S_2\), if \(\delta(q_i, a) = (q_j, x)\).
  If the Queue is empty then \((q_i, \epsilon) \implies (q_j, x)\), if \(\delta(q_i, \epsilon) = (q_j, x)\).

- The language accepted by the Automaton with Queue is the set of all strings \(w\) for which there exists a sequence of configurations beginning with the initial configuration \((q_s, w)\implies^*(q_a, x)\) and each configuration in the sequence is defined by the computation relation, above.

Solution Problem 3

Given a QFA \(M = (Q, \Sigma, \Gamma, \delta, q_s, q_a, q_r)\) we build a Turing machine \(M' = (Q', \Sigma', \Gamma', \delta', q'_0, q'_a, q'_r)\)
- \(\Gamma' = \Gamma \cup \{\uparrow\}\), where \(\uparrow\) is a new tape symbol, added to tape alphabet of \(M'\) so that the tape of \(M'\) can be converted to simulate the First-In-First-Out discipline. \(\uparrow\) is used to mark the left and right end of the queue.
- \(Q' = Q \cup Q_{aux}\), where \(Q_{aux}\) is auxiliary set of states, which \(M'\) needs to carry out the simulation.
- \(q'_s = q_s, q'_r = q_r, q'_a = q_a\).

The initial configuration of \(M\) \((q'_s, w)\) corresponds to \(q_sw\) configuration of \(M'\).
Following is an informal description of the simulation of the QFA $M$ by a Turing Machine $M'$:

1. $M'$ inserts $\uparrow$ symbol at the left end of its tape and marks the right end with $\uparrow$ symbol, i.e., the initial configuration $(q_s w) \xrightarrow{*} (q_s \uparrow w \uparrow)$. GOTO step 2.

2. If the current state is $q'_a$ or $q'_r$ accept, or reject, respectively and halt. Otherwise GOTO step 3.

3. Assuming the current state of $M'$ is $q_i$.
   
   - $M'$ reads the symbol under its tape head, let that be $a$, remembers it, and writes the $\uparrow$ symbol over it.
   
   - Then $M'$ checks $M$’s transition table. $M'$ has a copy of $M$’s transition table. If $\delta(q_i, a) = (q_j, x)$, then $M'$ remembers the state $q_j$ (as its ’next’ state) and $x$ in a buffer.
   
   - $M'$ moves its tape head to the right until it reaches $\uparrow$ symbol. ($M'$ simply reads a symbol under its head, and moves to the right until the symbol read is $\uparrow$; thus the tape head is over the right $\uparrow$ symbol.) $M'$ writes the first symbol of $x$ over it, moves one position to the right, writes the next symbol of $x$, and continues until $x$ is written in its entirety on the tape. $M'$ writes $\uparrow$ symbol, to designate the right end of the ’queue’ on the tape,
   
   - $M'$ moves the tape head to the left until it reads $\uparrow$ symbol. Then $M'$ moves one position to the right.
   
   - $M'$ sets its state to $q_j$. GOTO step 2.