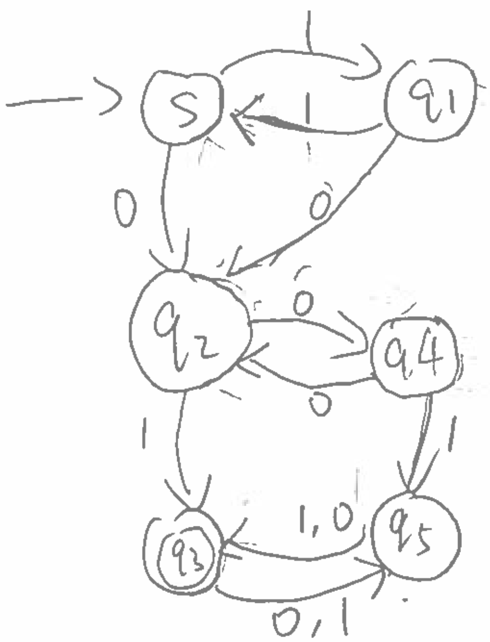
CS/ECE 374, Fall 2018	Gradescope name: Xihruiy 2 Ray Ying
Midterm 1: Problem 1	

Describe a DFA for the language defined below.

$$L = \{w \in \{0,1\}^* \mid w \text{ contains } 01 \text{ as a substring and } |w| \text{ is even}\}.$$

1011

For full credit your DFA must have at most 6 states. Briefly explain the states of the DFA. You may either draw the DFA or describe it formally in tuple notation. If you specify it via tuple notation, the states Q, the start state s, the accepting states A, and the transition function δ must be clearly specified.



S: Start state, doesn't contain 01 and is even (only contain even number of 1s)

q_1: state that contains odd number of 1s only.

q_2: State that end with a 0 and the lengeh is odd, doesn't have 01 substring.

q_3: State that contains substring 01 and has even length.

q_4: state that ends with 0 and it's even length, doesn't have 01 substring.

q_5: State that contain 01 substring but has odd length

CS/ECE 374, Fall 2018	Gradescope name: Xinmiy2
Midterm 1: Problem 2	Ray Ying

Assume $\Sigma = \{0, 1\}$. Recall that a block of 1's in a string is a maximal non-empty substring of 1's; the blocks of 1's are underlined in 01000110111101. Describe a regular expression for the language defined below.

$$L = \{w \in \{0,1\}^* \mid w \text{ has at most one block of 1's of even length}\}.$$

The strings 01110101 and 01101110 are in the language but 1101111 and 11011001001111 are not; in these examples even length blocks of 1s are underlined. Briefly explain your regular expression. It may help you to first consider strings which have no blocks of 1's with even length.

Since we can have at most one block of 1's of even length, that means we can have any number of 1's in odd length.

$$L = 0^*1(11)^* + 0^*(11)^*0^* + (0^*1(11)^*0)^*(11)^*(0^*1(11)^*0)^*$$

$0^*1(11)^*$: contain only one block of 1s that is odd length.

$(0^*1(11)^*0)^*$: contain multiple block of 1s that are all odd length.

$0^*(11)^*0^*$: contain only 0s or contain one block of 1s in even length.

$(0^*1(11)^*0^*)^*(11)^*(0^*1(11)^*0)^*$: contain any number of blocks of odd number of 1s and 0 or 1 block of 1s that is even length. (this also includes $(0^*1(11)^*0)^*$)

CS/ECE 374, Fall 2018	Gradescope name: Ray Ying
Midterm 1: Problem 3	Xinrui Y2

Given a language L over alphabet Σ recall that PREFIX(L) is the language defined over Σ as the collection of all prefixes of strings in L. Formally, PREFIX(L) $= \{u \mid \exists v \in \Sigma^*, uv \in L\}$. In this problem, assuming that L is regular, you will derive an algorithm that generates a regular expression r' for PREFIX(L) from a regular expression r for L. No justification necessary.

- For each of the base cases write a regular expression r' for PREFIX($L(r)$).

  (i) $r = \emptyset$: $r' = \emptyset$

  (ii) $r = \varepsilon$: $r' = \varepsilon$

  (iii) $r = a, a \in \Sigma$: $r' = \{a, \varepsilon\}$.

- Assume $r = r_1 + r_2$ and that r_1' and r_2' are regular expressions for PREFIX($L(r_1)$) and PREFIX($L(r_2)$) respectively. Write a regular expression r' for PREFIX($L(r)$) in terms of r_1, r_2, r_1', r_2'.

  $r' = r_1' + r_2'$

- Assume $r = r_1r_2$ and that r_1' and r_2' are regular expressions for PREFIX($L(r_1)$) and PREFIX($L(r_2)$) respectively. Write a regular expression r' for PREFIX($L(r)$) in terms of r_1, r_2, r_1', r_2'.

  $r' = r_1r_2' + r_1'$

- Assume $r = r_1^*$ and that r_1' is a regular expression for PREFIX($L(r_1)$). Write a regular expression r' for PREFIX($L(r)$) in terms of r_1, r_1'.

  $r' = (r_1)^* r_1'$

$r = r_1^n$

$r' = (r_1)^a r_1' \quad 0 \le a < n$

CS/ECE 374, Fall 2018	Gradescope name: Ray Ying
Midterm 1: Problem 4	xinrui yz

Prove that the language $\{a^i b^j c^k \mid i + j < k\}$ over the alphabet $\{a, b, c\}$ is not regular.

Let's $K = \{a^i b^j$ that $i \geq 0$ and $j \geq 0.\}$

So let $w + z < h$ and $x + y > h$, $(w, z, x, y > 0)$

We can construct $r_1 = a^w b^z \in K$ and $r_2 = a^x b^y \in K$

Then $r_1 c^h = a^w b^z c^h \in L$ since $L = \{a^i b^j c^k \mid j + i < k\}$ and $w + z < h$.

However, $r_2 c^h = a^x b^y c^h \notin L$ since $x + y > h$.

Combine that we have $r_1 \in K$, $r_2 \in K$, $r_1 c^h \in L$ and $r_2 c^h \notin L$, we conclude that K is an infinite fooling set for L, thus L is not regular.

CS/ECE 374, Fall 2018	**Gradescope name:** Ray Ying
Midterm 1: Problem 5	Xinrui Y2

Describe a CFG for the language $\{a^i b^j c^k \mid i + j < k\}$. In order to get full credit you should briefly explain how your grammar works, and the role of each non-terminal.

$i \geq 0$, $j \geq 0$

$S \to aSc \mid S_2$ $\{a^i b^j c^k \mid i+j<k\}$.

$S_1 \to cS_1 \mid c$ c^*.

$S_2 \to bS_2c \mid S_1$ $b^j c^k$ $j<k$

First, we can either have a in the language or not, then we can choose whether we want b in the language or not. Then we choose how many c we want and $c \geq 1$.

CS/ECE 374, Fall 2018	Gradescope name: Ray Ying
Midterm 1: Problem 6	xinruiy2

Let G_1, G_2, G_3 ~~be~~ context free grammars for languages L_1, L_2, L_3 respectively. Let $G_1 = (V_1, T, P_1, S_1)$ and $G_2 = (V_2, T, P_2, S_2)$ and $G_3 = (V_3, T, P_3, S_3)$ and assume that the non-terminal symbols V_1, V_2, V_3 are mutually disjoint (that is, they don't share any symbols). Describe a CFG $G = (V, T, P, S)$ for the language

$$L = L_1 + L_2L_3^*.$$

Hint; You may want to recall how we proved the closure properties of CFGs under union, concatenation and Kleene star.

$V = V_1 \mid V_2 \cup V_3 \cup T_2$

$T = T_1$

$S = S_1 \mid S_2.$

$P = P_1 \mid (P_2 \cup P_3^*)$

$L_1 = (V_1, T, P_1, S_1)$

$L_2 = (V_2, T, P_2, S_2)$

L_3^*: $V = V_3^* = V_3$

$T = T$

$S = (S_3, \text{start})$

$P = P_3^*$

(S, start) means the ~~start~~ start non-terminal symbols at the start L_3.

$L_2L_3^*$

$V = V_2 \cup V_3 \cup T_2$

$T = T.$

$S = S_2.$

$P = P_2 \cup P_3^*$

CS/ECE 374, Fall 2018	Gradescope name: Ray Ying
Midterm 1: Problem 7	Xinrui Yz

Bitstrings are another name for strings over the binary alphabet $\{0,1\}$. Given a bitstring w let $\mathsf{flip}(w)$ be the string obtained by "flipping" each bit of the string, that is changing a 0 to a 1 and a 1 to a 0. For example $\mathsf{flip}(010110) = 101001$. Given a language $L \subseteq \{0,1\}^*$ we define $\mathsf{flip}(L) = \{\mathsf{flip}(w) \mid w \in L\}$. As an example, if $L = \{0, 0110\}$ then $\mathsf{flip}(L) = \{1, 1001\}$. Given a language $L \in \{0,1\}^*$ we define $\mathsf{flipsuffix}(L)$ as follows.

$$\mathsf{flipsuffix}(L) = \{u\,\mathsf{flip}(v) \mid uv \in L\}.$$

flip(ε) = ε.

As an example, if $L = \{0, 0110\}$ then $\mathsf{flipsuffix}(L) = \{0, \underline{1}, 0110, 011\underline{1}, 01\underline{01}, 0\underline{001}, \underline{1001}\}$ where the underlined segments indicate the flipped suffixes.

(a) Given a DFA $M = (Q, \{0,1\}, \delta, s, A)$ for a regular language L, describe a DFA or NFA that accepts the language $\mathsf{flip}(L)$.

(b) Given a DFA $M = (Q, \{0,1\}, \delta, s, A)$ for a regular language L, describe a DFA or NFA that accepts the language $\mathsf{flipsuffix}(L)$. Note that $\mathsf{flipsuffix}(L)$ is not necessarily same as $\mathsf{PREFIX}(L) \cdot \mathsf{flip}(\mathsf{SUFFIX}(L))$. Part (a) is to help you think about part (b). If you are confident about the solution to part (b) you can skip part (a) and get full credit.

(a) DFA $M' = (Q', \{0,1\}, \delta', s', A')$ that accept flip(L).

$Q' = Q$

$S' = S$

$A' = A$

$\delta'(q, a) = \delta(q, b)$ $b \in \text{flip}(L)$; $a \in L$, $b = \text{flip}(a)$
flip(ε) = ε.

(a) Let's make the NFA $M' = (Q', \{0,1\}, \delta', s', A')$

$Q' = Q \times \{\text{before}, \text{after}\}$

$S' = (S, \text{before})$

$A' = (A, \text{after})$

$\delta'((q, r), a) = \begin{cases} (\delta(q,a), \text{before}) & r \in \text{before}, a \notin \varepsilon \\ (\delta(q,a), \text{after}) & r \in \text{after}, a \notin \varepsilon \end{cases}$

(q, before) is before the bit that you start to flip.
(q, after) is after the bit you start to flip.

$\delta'((q, r), \varepsilon) = \begin{cases} (S, \text{after}) & r \in \text{before} \\ (A, \text{after}) & r \in \text{before}, q \in A \end{cases}$

This page for extra work.

This page for extra work.

This page for extra work.

This page for extra work.

This page for extra work.