CS/ECE 374 A (Spring 2022)
Midterm 1 Solutions

---

1. (a) False. A counterexample: 01203 is accepted by the NFA but is not generated by $2^*0(10+3)^*$.

    [Note: a correct regular expression for this DFA would be $2^*0(12^*0+3)^*$, or $(2+03^*1)^*03^*$.]

   (b) True. By Kleene's theorem, every language accepted by a DFA is regular.

   (c) False. By reversing the direction of all transitions, the machine may become nondeterministic.

   (d) True. Both $(L_1 \cup L_2)^* L_1^*$ and $(L_1 \cup L_2 L_2^*)^*$ simplify to $(L_1 \cup L_2)^*$.

   (e) True. If x contains $0^n 1^n$ as a substring for some $n \geq 374$, then x contains $0^{374}1^{374}$ as a substring. The converse is trivially true. Thus, a regular expression is $(0+1)^* \cdot 0^{374}1^{374} \cdot (0+1)^*$.

   (f) True. A regular expression is $(0+1)^* \cdot \bigcup_{n=0}^{5} \{0^n 1^n 0^n\} \cdot (0+1)^*$. In fact, because $n = 0$ is not forbidden, the language is just $(0+1)^*$!

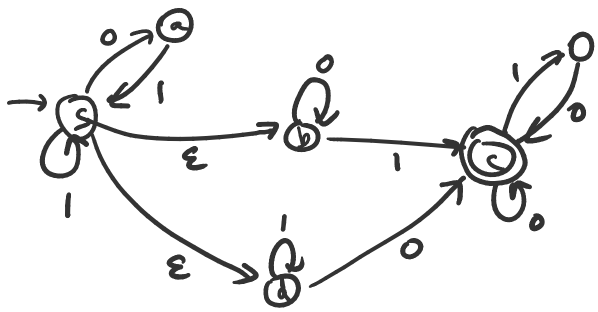
   (g) True. The set $F = \{0^i : 0 \leq i < 2022\}$ is a fooling set of size 2022. To see this, consider two distinct $x, y \in F$, i.e., $x = 0^i$ and $y = 0^j$ for some $0 \leq i, j < 2022$ with $i \neq j$. Pick $z = 0^{2022-i}$. Then $xz = 0^{2022}$ is in the language, but $yz = 0^{2022-i+j}$ is not in the language.

   (h) False. A counterexample: $\{0^n 1^n : n \geq 0\}$ is context-free but is not regular.

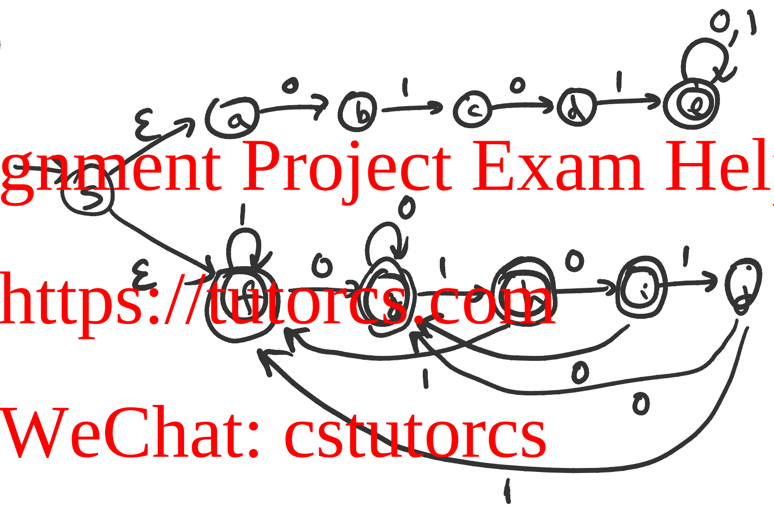
   (i) True. The grammar generates all strings of even length, i.e., $((0+1)^2)^*$, which is clearly regular.

February 21, 2022 5:29 PM

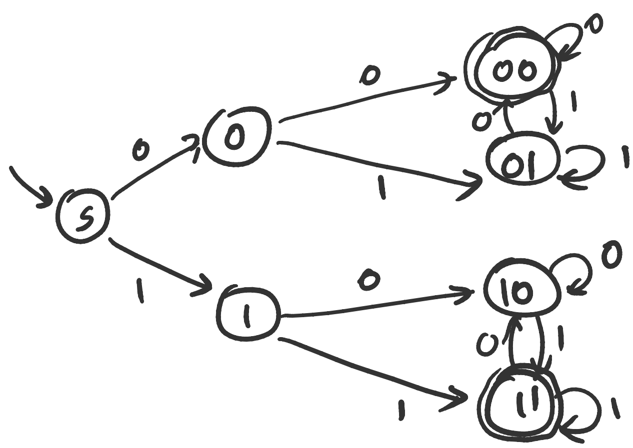
2. a)



b)



c)



2. (c) (Cont'd) Meaning of states:

- s: the start state.
- 0: read one 0.
- 1: read one 1.
- XY: second-to-last symbol is X, and last symbol read is Y.

[One alternative solution is to first draw an NFA (which requires just 4 states) and then apply the subset or power-set construction. Another alternative is to build DFAs for $(0+1)^*00$ and for $(0+1)^*11$ and then apply the product construction.]

3. (a) The language is $L = \bigcup_{i=1}^{374} 0^i(1^i)^*$. Since $0^i(1^i)^*$ is clearly regular for any fixed i and a finite union of regular languages is regular, L is regular.

(b) Define the following DFA $M = (Q, \{0,1\}, s, \delta, A)$:

$$\begin{aligned}
Q &= \big(\{0,1,\ldots,2021\} \times \{\text{BEFORE}, \text{AFTER}\}\big) \cup \{\text{ERR}\} \\
s &= (0, \text{BEFORE}) \\
A &= \{1,\ldots,2021\} \times \{\text{BEFORE}, \text{AFTER}\} \\
\delta((i, \text{BEFORE}), 0) &= ((i+1) \bmod 2022, \text{BEFORE}) \\
\delta((i, \text{BEFORE}), 1) &= ((i+1) \bmod 2022, \text{AFTER}) \\
\delta((i, \text{AFTER}), 1) &= ((i+1) \bmod 2022, \text{AFTER}) \\
\delta((i, \text{AFTER}), 0) &= \text{ERR} \\
\delta(\text{ERR}, 0) &= \text{ERR} \\
\delta(\text{ERR}, 1) &= \text{ERR}
\end{aligned}$$

Meaning of states:

- ERR is the error state.
- State (i, BEFORE) means that we have read only 0's and the number of symbols is congruent to i mod 2022.
- State (i, AFTER) means that we have read a sequence of 0's followed by a nonempty sequence of 1's, and and the total number of 0's plus 1's is congruent to i mod 2022.

4. (a) Choose $F = \{0^i : i \geq 0\}$.
Let x and y be two arbitrary distinct strings in F.
Then $x = 0^i$ and $y = 0^j$ for some $i \neq j$. Without loss of generality, assume $i < j$.
Choose $z = 1^j 2^i$.
Then $xz = 0^i 1^j 2^i \in L$, because $i = \min\{j, i\}$.
On the other hand, $yz = 0^j 1^j 2^i \notin L$, because $i < \min\{j, j\}$.
Thus, F is a fooling set.
Since F is infinite, L cannot be regular.

(b)

$$
\begin{aligned}
S &\rightarrow A \mid BC \\
A &\rightarrow 0A2 \mid A2 \mid D \\
D &\rightarrow 1D \mid \varepsilon \\
B &\rightarrow 0B \mid \varepsilon \\
C &\rightarrow 1C2 \mid C2 \mid \varepsilon
\end{aligned}
$$

Meaning of non-terminals:

- D generates 1^*.
- A generates $\{0^i 1^j 2^k : k \geq i\}$.
- B generates 0^*.
- C generates $\{1^j 2^k : k \geq j\}$.
- S generates the given language (since BC generates $\{0^i 1^j 2^k : k \geq j\}$.

5. (a) Since $L = \{\varepsilon, 01, 0101, 010101, 01010101, 0101010101, \ldots\}$,

$$
\begin{aligned}
\text{INSERT-FIFTH}(L) &= \{01010, 0101001, 010100101, 01010010101, \ldots\} \cup \\
&\quad \{01011, 0101101, 010110101, 01011010101, \ldots\},
\end{aligned}
$$

which can be described by the regular expression $(01010 + 01011) \cdot (01)^*$.

(b) Let L be a regular language over $\Sigma = \{0, 1\}$. By Kleene's theorem, L is accepted by some DFA $M = (Q, \Sigma, s, \delta, A)$. We construct an NFA $M' = (Q', \Sigma, s', \delta', A')$ accepting INSERT-FIFTH(L) (which would imply that INSERT-FIFTH(L) is regular by Kleene's theorem). The construction is as follows:

$$
\begin{aligned}
Q' &= \{(q, i, \text{BEFORE}) : q \in Q,\ i \in \{0,1,2,3,4\}\} \ \cup \\
&\quad \{(q, \text{AFTER}) : q \in Q\} \\
s' &= (s, 0, \text{BEFORE}) \\
A' &= \{(q, \text{AFTER}) : q \in A\}
\end{aligned}
$$

$$
\begin{aligned}
\delta'((q, i, \text{BEFORE}), c) &= \{(\delta(q, c), i+1, \text{BEFORE})\} && \forall q \in Q,\ c \in \Sigma,\ i \in \{0,1,2,3\} \\
\delta'((q, 4, \text{BEFORE}), a) &= \{(q, \text{AFTER})\} && \forall q \in Q,\ a \in \Sigma \\
\delta'((q, \text{AFTER}), c) &= \{(\delta(q, c), \text{AFTER})\} && \forall q \in Q,\ c \in \Sigma
\end{aligned}
$$

(This machine is in fact a DFA!)

Explanation: The idea is to divide the process into two phases: BEFORE (reading the prefix x) and AFTER (reading the suffix ay). At the same time, we simulate M on the string xay.