Indian Institute of Technology Kharagpur
Department of Computer Science and Engineering

Foundations of Computing Science (CS60005) **Autumn Semester, 2021-2022**

Test - 2 [Marks: 30] **Date: 06-Oct-2021 (Wednesday), 8:15am – 9:30am** **Venue: Online**

[Instructions: ***There are THREE questions. Answer ALL questions. Be brief and precise.*** **]**

Q1. One of the following languages over the alphabet $\{a, b\}$ is regular; the other one is not regular. Identify which one is what and Justify. *No credit will be given only for correct identification (without justification provided).*

(a) $L_1 = \left\{\alpha\beta\alpha \mid \alpha \in \{a,b\}^+ \text{ and } \beta \in \{a,b\}^+\right\}$ (b) $L_2 = \left\{\alpha\beta\alpha \mid \alpha \in \{a\}^+ \text{ and } \beta \in \{a,b\}^+\right\}$

In particular, do the following: (i) For the one which is *regular*, give the corresponding Regular Expression as well as the minimum-state deterministic finite automaton (DFA) *both*; and (ii) For the one which is *not regular*, give a proof using the Pumping Lemma. **[Marks: (3 + 3) + 4 = 10]**

Solution:

(a) *L_1 is not regular.*

For proving this, suppose that L_1 is regular. Let $k \in \mathbb{N}$ be a pumping lemma constant for L_1. Consider the string $ab^ka^2b^k$ which belongs to L_1 (here $\alpha = ab^k$ and $\beta = a$). Using the notations of the pumping lemma, take $x = a$, $y = b^k$, and $z = a^2b^k$. The pumping lemma gives a decomposition $y = uvw$ with $|v| > 0$ (so $v = b^l$ for some $1 \leq l \leq k$). Pumping out v gives $uw = ab^{k-l}a^2b^k \in L_1$. But this string cannot be written in the form $\alpha\beta\alpha$, and so is not in L_1, a contradiction!

(b) *L_2 is regular.*

Regular Expression: L_2 represents the set of all strings over $\{a, b\}$ of length ≥ 3, that start and end with a. Therefore, the corresponding regular expression is,

$$\mathcal{L}_2 = a(a+b)^+a = a(a+b)(a+b)^*a$$

DFA: The minimum-state DFA description is given as: $\mathcal{M}_2 = \left(\{q_0, q_1, q_2, q_3, q_4\}, \{a, b\}, q_0, \delta, \{q_4\}\right)$ where, δ is defined as,

$$\begin{array}{lll} \delta(q_0,a) = q_2 & \delta(q_2,a) = q_3 & \\ \delta(q_0,b) = q_1 & \delta(q_2,b) = q_3 & \delta(q_4,a) = q_4 \\ \delta(q_1,a) = q_1 & \delta(q_3,a) = q_4 & \delta(q_4,b) = q_3 \\ \delta(q_1,b) = q_1 & \delta(q_3,b) = q_3 & \end{array}$$

Here, q_1 indicates the dead-state which is reached when we receive b as the first symbol of the input string.

Q2. Let $\mathcal{B}_n$ ($n \geq 1$) denote the binary representation (string with leading zeros omitted) corresponding to any $n \in \mathbb{N}$. For example, you may consider, $\mathcal{B}_5 = 101$, $\mathcal{B}_{10} = 1010$, and $\mathcal{B}_{13} = 1101$. Let $\#$ be another symbol not in $\{0, 1\}$. We denote $\mathcal{B}_n^{rev}$ to indicate the reverse of string $\mathcal{B}_n$. For example, $\mathcal{B}_5^{rev} = 101$, $\mathcal{B}_{10}^{rev} = 0101$, and $\mathcal{B}_{13}^{rev} = 1011$.

One of the following languages over the alphabet $\{0, 1, \#\}$ is context-free; the other one is not context-free. Identify which one is what and Justify. *No credit will be given only for correct identification (without justification provided).*

$$(\mathbf{a}) \quad L_3 = \left\{ \mathcal{B}_n \# \mathcal{B}_{n+1} \mid n \geq 1 \right\} \qquad (\mathbf{b}) \quad L_4 = \left\{ \mathcal{B}_n^{rev} \# \mathcal{B}_{n+1} \mid n \geq 1 \right\}$$

In particular, do the following: (i) For the one which is *a context-free language (CFL)*, give the corresponding context-free grammar (CFG) as well as the push-down automaton (PDA) *both*; and (ii) For the one which is *not a CFL*, give a proof using the Pumping Lemma for CFL. **[Marks: (3 + 3) + 4 = 10]**

Solution:

(a) *L_3 is not a CFL.*

For proving this, suppose L_3 is a CFL. Let $k \in \mathbb{N}$ be a pumping lemma constant for L_3. Consider the string $w = 1^k 0^k \# 1^k 0^{k-1} 1$ which belongs to L_3 (here $\mathcal{B}_m = 1^k 0^k$ and so $\mathcal{B}_{m+1} = 1^k 0^{k-1} 1$, where $m = \sum_{i=k}^{2k-1} 2^i = 2^k(2^k - 1) = 4^k - 2^k$). Suppose $w = uvxyz$, where $|vxy| \leq k$ and $|vy| \geq 1$. If vxy does not contain a $\#$, then pumping either way will cause a contradiction (increasing or decreasing one of the numbers without touching the other). If the $\#$ is contained in v or y, then pumping either way leads to a string not even in $(0+1)^*\#(0+1)^*$, i.e., a string definitely outside of L_3. Because of the $|vxy| \leq k$ condition, the only remaining possibility is for $v = 0^j$ and $y = 1^l$, $j, l \geq 1$, to fall on opposite sides of the "$\#$". But pumping up in this case means multiplying the left hand number by some power of two, while it never means multiply the right hand number by some power of two. Thus, the pumped-up strings will not remain with the right number one more than the left, all cases leading to a contradiction!

(b) *L_4 is a CFL.*

CFG: Note that, if we add 1 to n, then in $\mathcal{B}_n$, all the trailing 1's are flipped to 0, then the 0 (the rightmost 0 of $\mathcal{B}_n$) is flipped to 1, and all the other most-significant bits before that 0 remain unchanged.

From this description of binary addition, we can construct the corresponding CFG, $G = \big(\{S, A, B, C\}, \{0, 1, \#\}, P, S\big)$, as follows where the productions rules (P) are:

$$\begin{aligned} S &\rightarrow A \mid B \\ A &\rightarrow 1\#10 \mid 1A0 \\ B &\rightarrow 1B0 \mid 0C1 \\ C &\rightarrow 1\#1 \mid 1C1 \mid 0C0 \end{aligned}$$

The production rule for A handles the case that $\mathcal{B}_n$ is a string of 1's. The production rules for B and C handle the case that $\mathcal{B}_n$ contains 0's and 1's and has a leading 1.

PDA: The machine will read the binary encoding of n from the input, from least-significant digit to most-significant digit, at the same time pushing the binary encoding of $n + 1$ onto its stack. When it encounters the $\#$, it will match the stack against the remaining input.

The PDA description is given as: $\mathcal{M}_4 = \big(\{p_0, p_1, q\}, \{0, 1, \#\}, \{0, 1, \perp\}, \delta, p_1, \phi\big)$
where, δ is defined as,

$$\begin{aligned} (p_1, 0, A) &\mapsto (p_0, 1A) \\ (p_1, 1, A) &\mapsto (p_1, 0A) \\ (p_0, 0, A) &\mapsto (p_0, 0A) \\ (p_0, 1, A) &\mapsto (p_0, 1A) \end{aligned} \qquad \begin{aligned} (p_1, \#, A) &\mapsto (q, 1A) \\ (p_0, \#, A) &\mapsto (q, A) \end{aligned} \qquad \begin{aligned} (q, 1, 1) &\mapsto (q, \epsilon) \\ (q, 0, 0) &\mapsto (q, \epsilon) \\ (q, \epsilon, \perp) &\mapsto (q, \epsilon) \end{aligned}$$

The two states p_0, p_1 represent a *carry* value of 0 or 1, respectively; the machine transitions to state q after reading $\#$; in state q it matches the stack against the remaining input. *[acceptance by empty stack]*

Q3. Consider the language $E = \{w \in \{0,1\}^* \mid$ number of 0's in w is twice the number of 1's$\}$.

(a) Design a total single tape Turing machine that accepts this language. Precisely write down the tape alphabet, the set of states and the transition function.

(b) Write down the sequence of configurations of the Turing machine from part (a) on the input strings 101100 and 010010.

[Marks: 8+(1+1) = 10]

Solution:

(a) Below is the description of one possible Turing machine $\mathcal{M}$ that accepts the language E.

Tape alphabet: $\Gamma = \{\vdash, 0, 1, X, \sqcup\}$
States: $Q = \{s, [q_1, \sqcup, \sqcup], [q_1, 0, \sqcup], [q_1, 1, \sqcup], [q_1, 0, 0], [q_1, 0, 1], [q_1, 1, 0], q_2, t, r\}$

Transition function:

State	$\vdash$	0	1	X	$\sqcup$
s	$([q_1, \sqcup, \sqcup], \vdash, R)$	-	-	-	-
$[q_1, \sqcup, \sqcup]$	-	$([q_1, 0, \sqcup], X, R)$	$([q_1, 1, \sqcup], X, R)$	$([q_1, \sqcup, \sqcup], X, R)$	$(t, -, -)$
$[q_1, 0, \sqcup]$	-	$([q_1, 0, 0], X, R)$	$([q_1, 0, 1], X, R)$	$([q_1, 0, \sqcup], X, R)$	$(r, -, -)$
$[q_1, 1, \sqcup]$	-	$([q_1, 1, 0], X, R)$	$([q_1, 1, \sqcup], 1, R)$	$([q_1, 1, \sqcup], X, R)$	$(r, -, -)$
$[q_1, 0, 0]$	-	$([q_1, 0, 0], 0, R)$	(q_2, X, L)	$([q_1, 0, 0], X, R)$	$(r, -, -)$
$[q_1, 0, 1]$	-	(q_2, X, L)	$([q_1, 0, 1], 1, R)$	$([q_1, 0, 1], X, R)$	$(r, -, -)$
$[q_1, 1, 0]$	-	(q_2, X, L)	$([q_1, 1, 0], 1, R)$	$([q_1, 1, 0], X, R)$	$(r, -, -)$
q_2	$([q_1, \sqcup, \sqcup], \vdash, R)$	$(q_2, 0, L)$	$(q_2, 1, L)$	(q_2, X, L)	-

Once $\mathcal{M}$ enters state t or r it stays in that state. The transitions for states t, r are not provided. For all $a \in \Gamma$, $\delta(t, a)$ could be anything as long as the first component i.e., the destination state is t.

(b) Sequence of configurations of $\mathcal{M}$ on input 101100:

$$
\begin{aligned}
(s, \vdash 101100, 0) &\xrightarrow[\mathcal{M}]{1} ([q_1, \textvisiblespace, \textvisiblespace], \vdash 101100, 1) \xrightarrow[\mathcal{M}]{1} ([q_1, 1, \textvisiblespace], \vdash X01100, 2) \\
&\xrightarrow[\mathcal{M}]{1} ([q_1, 1, 0], \vdash XX1100, 3) \xrightarrow[\mathcal{M}]{1} ([q_1, 1, 0], \vdash XX1100, 4) \\
&\xrightarrow[\mathcal{M}]{1} ([q_1, 1, 0], \vdash XX1100, 5) \xrightarrow[\mathcal{M}]{1} (q_2, \vdash XX11X0, 4) \\
&\xrightarrow[\mathcal{M}]{1} (q_2, \vdash XX11X0, 3) \xrightarrow[\mathcal{M}]{1} (q_2, \vdash XX11X0, 2) \\
&\xrightarrow[\mathcal{M}]{1} (q_2, \vdash XX11X0, 1) \xrightarrow[\mathcal{M}]{1} (q_2, \vdash XX11X0, 0) \\
&\xrightarrow[\mathcal{M}]{1} ([q_1, \textvisiblespace, \textvisiblespace], \vdash XX11X0, 1) \xrightarrow[\mathcal{M}]{1} ([q_1, \textvisiblespace, \textvisiblespace], \vdash XX11X0, 2) \\
&\xrightarrow[\mathcal{M}]{1} ([q_1, \textvisiblespace, \textvisiblespace], \vdash XX11X0, 3) \xrightarrow[\mathcal{M}]{1} ([q_1, 1, \textvisiblespace], \vdash XXX1X0, 4) \\
&\xrightarrow[\mathcal{M}]{1} ([q_1, 1, \textvisiblespace], \vdash XXX1X0, 5) \xrightarrow[\mathcal{M}]{1} ([q_1, 1, \textvisiblespace], \vdash XXX1X0, 6) \\
&\xrightarrow[\mathcal{M}]{1} ([q_1, 1, 0], \vdash XXX1XX, 7) \xrightarrow[\mathcal{M}]{1} (r, \vdash XXX1XX, -)
\end{aligned}
$$

Sequence of configurations of $\mathcal{M}$ on input 010010:

$$
\begin{aligned}
(s, \vdash 010010, 0) &\xrightarrow[\mathcal{M}]{1} ([q_1, \textvisiblespace, \textvisiblespace], \vdash 010010, 1) \xrightarrow[\mathcal{M}]{1} ([q_1, 0, \textvisiblespace], \vdash X10010, 2) \\
&\xrightarrow[\mathcal{M}]{1} ([q_1, 0, 1], \vdash XX0010, 3) \xrightarrow[\mathcal{M}]{1} (q_2, \vdash XXX010, 2) \\
&\xrightarrow[\mathcal{M}]{1} (q_2, \vdash XXX010, 1) \xrightarrow[\mathcal{M}]{1} (q_2, \vdash XXX010, 0) \\
&\xrightarrow[\mathcal{M}]{1} ([q_1, \textvisiblespace, \textvisiblespace], \vdash XXX010, 1) \xrightarrow[\mathcal{M}]{1} ([q_1, \textvisiblespace, \textvisiblespace], \vdash XXX010, 2) \\
&\xrightarrow[\mathcal{M}]{1} ([q_1, \textvisiblespace, \textvisiblespace], \vdash XXX010, 3) \xrightarrow[\mathcal{M}]{1} ([q_1, \textvisiblespace, \textvisiblespace], \vdash XXX010, 4) \\
&\xrightarrow[\mathcal{M}]{1} ([q_1, 0, \textvisiblespace], \vdash XXXX10, 5) \xrightarrow[\mathcal{M}]{1} ([q_1, 0, 1], \vdash XXXXX0, 6) \\
&\xrightarrow[\mathcal{M}]{1} (q_2, \vdash XXXXXX, 5) \xrightarrow[\mathcal{M}]{1} (q_2, \vdash XXXXXX, 4) \\
&\xrightarrow[\mathcal{M}]{1} (q_2, \vdash XXXXXX, 3) \xrightarrow[\mathcal{M}]{1} (q_2, \vdash XXXXXX, 2) \\
&\xrightarrow[\mathcal{M}]{1} (q_2, \vdash XXXXXX, 1) \xrightarrow[\mathcal{M}]{1} (q_2, \vdash XXXXXX, 0) \\
&\xrightarrow[\mathcal{M}]{1} ([q_1, \textvisiblespace, \textvisiblespace], \vdash XXXXXX, 1) \xrightarrow[\mathcal{M}]{1} ([q_1, \textvisiblespace, \textvisiblespace], \vdash XXXXXX, 2) \\
&\xrightarrow[\mathcal{M}]{1} ([q_1, \textvisiblespace, \textvisiblespace], \vdash XXXXXX, 3) \xrightarrow[\mathcal{M}]{1} ([q_1, \textvisiblespace, \textvisiblespace], \vdash XXXXXX, 4) \\
&\xrightarrow[\mathcal{M}]{1} ([q_1, \textvisiblespace, \textvisiblespace], \vdash XXXXXX, 5) \xrightarrow[\mathcal{M}]{1} ([q_1, \textvisiblespace, \textvisiblespace], \vdash XXXXXX, 6) \\
&\xrightarrow[\mathcal{M}]{1} ([q_1, \textvisiblespace, \textvisiblespace], \vdash XXXXXX, 7) \xrightarrow[\mathcal{M}]{1} (t, \vdash XXXXXX, -)
\end{aligned}
$$