“CS/ECE 374”: Algorithms and Models of Computation, Fall 2018

Midterm 1: October 1, 2018

Real name:	
NetID:	

Gradescope name:	
Gradescope email:	

- This is a closed-book, closed-notes, closed-electronics exam. If you brought anything except your writing implements, put it away for the duration of the exam. In particular, you may not use *any* electronic devices other than those that are medically necessary.
- Please clearly print your real name, your university NetID, your Gradescope name, and your Gradescope email address in the boxes above. If you are using your real name and your university email address on Gradescope, you do ***not*** need to write everything twice. **We will not scan this page into Gradescope.**
- Please also print **only the name you are using on Gradescope** at the top of every page of the answer booklet, except this cover page. These are the pages we will scan into Gradescope.
- Please do not write outside the black boxes on each page; these indicate the area of the page that the scanner can actually see.
- This answer booklet is **double-sided**!
- If you run out of space for an answer, feel free to use the scratch pages at the back of the answer booklet, but **please clearly indicate where we should look**.
- **Please read the entire exam before writing anything.** There are seven numbered problems and each problem is worth 10 points.
- **You have 150 minutes.**
- **Proofs are required only if we specifically ask for them.**

CS/ECE 374, Fall 2018	Gradescope name:
Midterm 1: Problem 1	

Describe a DFA for the language defined below.

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

. 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.

$L = L_1 \cap L_2$

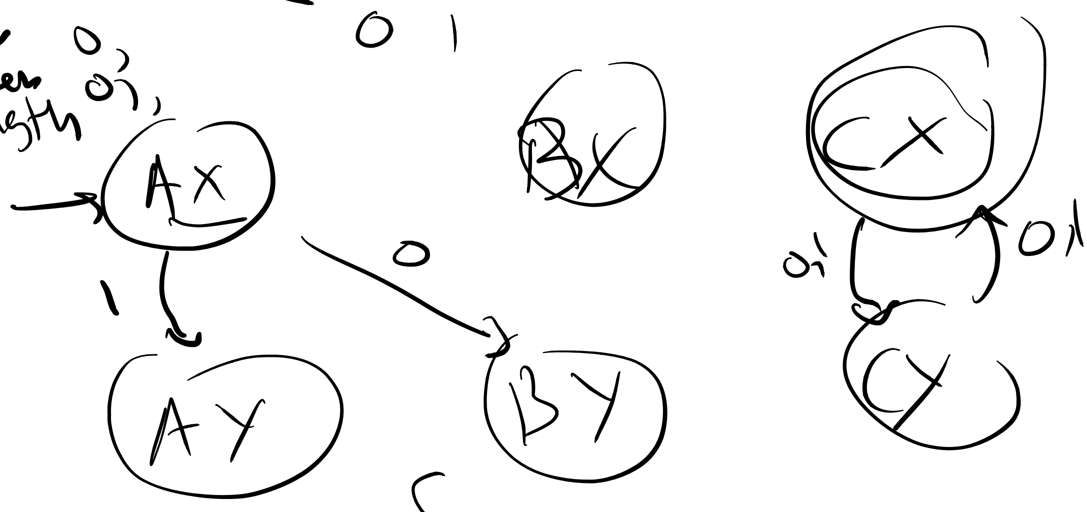
$L_1 = \{$ has 01 substring $\}$

$L_2 = \{ |w|$ is even $\}$

M_1

M_2

BX → last char 0, have not seen 01, even length



δ_1	0	
A	0	B
A	1	A

δ_2		
X	0,1	Y
Y	0,1	X

$$\delta((q_1, a_2)) = (\delta_1(q_1), \delta_2(q_2))$$

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

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 $0\underline{1}000\underline{11}0\underline{1111}0\underline{1}$. 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 $0\underline{11}01110$ are in the language but $\underline{11}0\underline{1111}$ and $\underline{11}0\underline{11}00100\underline{1111}$ are not. Even length blocks of 1s are underlined. Briefly explain your regular expression.

$11(11)^*$ - even block 1's

00^* - block 0's

$1(11)^*$ - odd block 1's

$(\epsilon + 1(11)^*)(00^*\,1(11)^*)^*$

$\frac{(\epsilon + 00^*)}{00^*}$

1 even block of 1's

no even blocks of 1's

A) starts w/ 1

B) doesn't 1

all blocks are odd w/ 1st block being 1

$1(11)^*(00^*\,1(11)^*)^*\,(\epsilon + 00^*)$

$00^*(1(11)^*\,00^*)^*\,(1(11)^* + \epsilon) + \epsilon$

$(1(11)^* + 00^*)^*$

CS/ECE 374, Fall 2018 Midterm 1: Problem 3	Gradescope name:

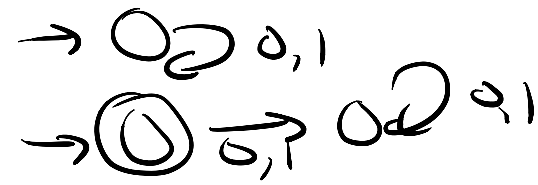
Given a language L over alphabet Σ recall that $\mathsf{PREFIX}(L)$ is the language defined over Σ as the collection of all prefixes of strings in L. Formally, $\mathsf{PREFIX}(L) = \{u \mid \exists v \in \Sigma^*, uv \in L\}$. In this problem, assuming that L is regular, you will derive an algorithm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

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

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

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

  (iii) $r = a$, $a \in \Sigma$: $\quad r' = a + \epsilon$



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

$r' = r_1' + r_2'$

$L_1 \quad L_2$

$\mathsf{PREFIX}(L_1 \cup L_2) =$

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

$r' = r_1' + r_1 r_2'$

→ unless $r_1 = \emptyset$?

$r = (\emptyset + \emptyset)^*$?

$r = ab$ ($r_1 = a$, $r_2 = b$)

$r_1' = \boxed{(a+\epsilon)}$

$r_2' = (b+\epsilon)$

$r' = (a+\epsilon) + a(b+\epsilon)$

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

$r' =$

$r = ab$ ($r_1 = a$, $r_2 = b$)

$r_1' = a + \epsilon$

$r_2' = b + \epsilon$

$r' = r_1 r_2' = a(b+\epsilon)$ × "b"

$r' = r_1' r_2' = (a+\epsilon)(b+\epsilon)$

$r' = r_1' + r_1 r_2'$

$r_1 = a$

$r_2 = \emptyset$

$r = r_1 r_2 = a\emptyset = \emptyset$

$\mathsf{PREFIX}(r) = \emptyset$

$(a+\epsilon) + a(\emptyset)$

CS/ECE 374, Fall 2018	Gradescope name:
Midterm 1: Problem 4	

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

$L =$

$F = a^*$

$a^x \quad a^y \quad x \neq y$

wolog $\underline{x < y}$

$a^x c^y \in L$

$a^y c^y \notin L$

$a^x c^{x+1} \in L$

$a^y c^{x+1} \notin L$

$x+1 \leq y$

$\underline{L} \cap a^* c^*$ not regular

CS/ECE 374, Fall 2018	Gradescope name:
Midterm 1: Problem 5	

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.

more c's than a's, b's

aaa bb ccccccc

$b^n c^n \rightarrow X \rightarrow bXc \mid \epsilon$

$a^m b^n c^n c^m \quad Y \rightarrow aYc \mid X$

$c^n \; n \geq 1 \quad Z \rightarrow c \mid cZ$

$S \rightarrow YZ$

$a^m b^n c^{m+n+k} \quad k > 0$

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

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

$$L = L_1 + L_2 L_3^*.$$

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

CS/ECE 374, Fall 2018	Gradescope name:
Midterm 1: Problem 7	

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 1 and a 1 to a 0. For example $\mathsf{flip}(010110) = 101001$. Given a language $L \subset \{0,1\}^*$ 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.

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

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 $\mathsf{PREFIX}(L) \cdot \mathsf{flip}(\mathsf{SUFFIX}(L))$. The previous part is to help you think about this second part. If you are confident about the solution to this part you can skip the previous part and get full credit.

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.