CS 4510 Automata and Complexity

Exam 1: Regular languages

Date: 3:30pm-4:45pm Tuesday, September 20

- Name: ____ GTID#: ____
- **When handing in this test, please go to a TA with your BuzzCard to verify the name on the test.**
- Do not open this exam until you are directed to do so. Read all the instructions first.
- This booklet contains 4 **questions on** 5 **pages**, including this one.
- **Write your name and GTID# (numbers, not letters) on the top of every page.** Your GTID# can be found on your BuzzCard.
- By submitting this exam, you agree that your actions during this exam conform with the Georgia Tech Honor Code.
- Write your solutions in the space provided. If you run out of space, continue your answer on the back of the last page, and make a notation on the front of the sheet.
- There is a page of scratch paper at the end of the exam. We can provide additional scratch paper if you need it. The only outside material you may use on the exam is one (1) double-sided, hand-written, 8.5"x11" page of notes.
- Calculators are NOT permitted.
- You may use any of the theorems/facts/lemmas from the lecture notes, homeworks, or textbook without re-proving them unless explicitly stated otherwise.
- If you have a question, you may ask a TA. You should not communicate with anyone other than teaching staff during the exam.
- Do not spend too much time on any one problem! If you get stuck, move on and come back to that one later.
- Good luck!

Name: ________________________ GTID#: ________________________

0. **Following Instructions** (1 point extra credit)

   Please write your name and GTID# (numbers, not letters) at the top of EACH page of this exam. Your GTID# can be found on your BuzzCard.

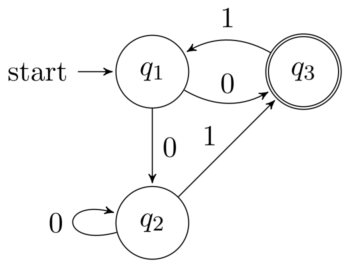
1. **Circle the answers**

   (a) (4 points each) Circle one. No explanation necessary.

   i. Let R be a regular expression. Then $R \cup \emptyset^*$ ($\boxed{\text{ALWAYS}}$ / SOMETIMES / NEVER) generates at least one string. **Answer:** ALWAYS

   ii. Suppose L is a language. Then there (ALWAYS / $\boxed{\text{SOMETIMES}}$ / NEVER) exists a DFA that recognizes L. **Answer:** SOMETIMES

   (b) (4 point) Suppose we wished to convert the following NFA to a DFA as shown in class and in the book:



   Which of the following will be states in the resulting DFA? (Circle all that apply, if any. No explanation necessary.)

   - q_1
   - $\boxed{\{q_1, q_3\}}$ **Answer:** CIRCLE THIS
   - $\boxed{\emptyset}$ **Answer:** CIRCLE THIS
   - $q_{\textbf{FAIL}}$

   (c) (8 points) You are using the pumping lemma to show that the language $\{w \in \{0,1\}^* \mid w$ has an equal number of 0s and 1s$\}$ is not regular. You have already assumed that L is regular, and let p be the pumping length. Which of the following choices for s allow you to successfully complete the proof, where s is the counterexample string which you show cannot be pumped? (Circle all that apply, if any. No explanation necessary.)

   - $\boxed{0^p1^p}$ **Answer:** CIRCLE THIS
   - $0^{p+1}1^p$
   - 00001111
   - $(01)^p$

Name: ____ GTID#: ____

2. **Pumping Lemma** (30 points)
Prove that **one** of the following languages is not regular using the Pumping Lemma.

- $L = \{w \mid w$ contains an even number of 0's or every odd position of w is a $1\}$.
- $\Delta = \{1^j 0^k 1^\ell \mid k > 4, j = \ell\}$.
- The language described by the regular expression $(0 \cup \varepsilon)(10)^*(1 \cup \varepsilon)$.

Answer: In the next problem I give a regular expression for L. The third language is directly described by a regular expression. As a language is regular if and only if it can be described by a regular expression, we must pick Δ.

Suppose Δ is regular and let $p \in \mathbb{Z}^{\geq 0}$ be its "pumping length". Define $w = 1^p000001^p$. Note that $|w| = 2p+5 > p$ and $w \in \Delta$ (because $j = \ell = p$ and $k = 5$). So the pumping lemma would claim there exists x, y, z so that $w = xyz$, $|xy| \leq p$, $|y| > 0$, and $xy^iz \in L$ for all i.

The conditions $w = xyz$ and $|xy| \leq p$ tells us that y must be contained entirely in the run of 1's at the beginning of w. But then consider the $i = 2$ case for the final condition. We have $xy^2z = 1^{p+|y|}0^51^p$. The condition that $|y| > 0$ give $p + |y| \neq p$ and so $xy^2z \notin \Delta$.[1] This contradicts the pumping lemma and hence our assumption that Δ is regular. Thus Δ is not regular.

[1] We could also have used $xy^0z = 1^{p-|y|}0^51^p$ or indeed any choice of i other than 1.

Name: ____ GTID#: ____

3. **Regular Expressions** (20 points - 10 each)
Give a regular expression for **two** of the following binary languages. Use **only** the symbols $\emptyset$, ε, 0, 1, $\circ$, $\cup$, *,), and (. You may leave $\circ$'s implicit. Please box your final answers.

- $L = \{w \mid w$ contains an even number of 0's or every odd position of w is a 1$\}$.
- $P = \{w \mid w$ contains zero 1's and n^2 0's for some $n \in \mathbb{Z}^{\geq 0}\}$
- The set of binary strings containing at least three 1's.

Answer: P is not regular, so we must select the other languages. To write a regular expression for L we can first write them for "contains an even number of 0's" and "every odd position of w is a 1" then combine with a union. The former can be described by $1^*(01^*01^*)^*$ and the latter by $\varepsilon \cup 1((0 \cup 1)1)^*(\varepsilon \cup 0 \cup 1)$. For understanding we can break the latter expression down into pieces and build it back up.

- $((0 \cup 1)1)^*$ gives even length strings with a 1 in every even position.
- $1((0 \cup 1)1)^*$ makes it odd length strings with a 1 in every odd position.
- $1((0 \cup 1)1)^*(\varepsilon \cup 0 \cup 1)$ allows it to be odd length (pick ε at the end) or even length (pick 0 or 1 at the end).
- $\varepsilon \cup 1((0 \cup 1)1)^*(\varepsilon \cup 0 \cup 1)$ to add the length 0 string.

Thus we get $\boxed{1^*(01^*01^*)^* \cup \varepsilon \cup 1((0 \cup 1)1)^*(\varepsilon \cup 0 \cup 1)}$ for L.[2]

For the third language, we have $\boxed{0^*10^*10^*1(0 \cup 1)^*}$.

There are, of course, a wide variety of ways to write regular expressions for either language.

[2] $1^*(01^*01^*)^* \cup 1((0 \cup 1)1)^*(\varepsilon \cup 0 \cup 1)$ works just as well as the first term anyway includes ε.

Name: ______________________________ GTID#: ______________________________

4. **Closure** (30 points)
Here are three operations on languages.

- $\mathsf{MIX}(A, B) = \{a_1 b_1 \cdots a_n b_n \mid a_1 \cdots a_n \in A, b_1 \ldots b_n \in B\}$
- $\mathsf{WEAVE}(A, B, C) = \{a_1 b_1 c_1 \cdots a_n b_n c_n \mid a_1 \cdots a_n \in A, b_1 \ldots b_n \in B, c_1 \ldots c_n \in C\}$
- $\mathsf{2WEAVE}(A, B, C)$ is defined the same as WEAVE except it includes any string where (at least) two of $a_1 \cdots a_n \in A$, $b_1 \ldots b_n \in B$, and $c_1 \ldots c_n \in C$ hold.

Let $A = \{\texttt{cat}\}$, $B = \{110\}$, and $C = \{\triangle\Box\triangle\}$. Then $\mathsf{MIX}(A, B) = \{\texttt{c1a1t0}\}$ and $\mathsf{WEAVE}(A, B, C) = \{\texttt{c1}\triangle\texttt{a1}\Box\texttt{t0}\triangle\}$. $\mathsf{2WEAVE}(A, B, C)$ in includes $\texttt{c1}\Box\texttt{a1}\Box\texttt{t0}\Box$, $\texttt{d1}\triangle\texttt{o1}\Box\texttt{g0}\triangle$, $\texttt{c1}\triangle\texttt{a1}\Box\texttt{t0}\triangle$, and more. It does not contain are $\texttt{c1a1t0}$, $1\texttt{c}\Box 1\texttt{a}\Box 0\texttt{t}\Box$, or $\texttt{c1}\Box\texttt{a1}\Box\texttt{t1}\Box$. If $A' = \{\texttt{cats}\}$ then $\mathsf{MIX}(A', B) = \mathsf{WEAVE}(A', B, C) = \emptyset$

(25 points) Prove that if A, B, and C are regular, then so is $\mathsf{WEAVE}(A, B, C)$. (Hint: you might consider trying to prove closure for MIX first and then generalize your idea.)

(5 points) Prove that if A, B, and C are regular, then so is $\mathsf{2WEAVE}(A, B, C)$. (Hint: modify your proof for WEAVE.)

General Idea: Roughly the idea is that we want to run machines for the different languages in parallel, similar to the DFA-based proof that regular languages are closed under union. However, unlike in that proof we want to alternate which machine processes each character, instead of having every character processed by every machine. For this we need an extra "variable" to track which machine is "next".

Answer: Let $M_A = (Q^A, \Sigma, q_0^A, \delta^A, F)$, $M_B = (Q^B, \Sigma, q_0^B, \delta^B, F)$, and $M_B = (Q^C, \Sigma, q_0^C, \delta^C, F)$ be DFAs for A, B, and C respectively.[3] We define out new DFA $M = (Q, \Sigma, q_0, \delta, F)$ as follows,

$$\begin{aligned} Q &= \{\mathcal{A}, \mathcal{B}, \mathcal{C}\} \times Q^A \times Q^B \times Q^C \\ q_0 &= (\mathcal{A}, q_0^A, q_0^B, q_0^C) \\ F &= \{(\mathcal{A}, q^A, q^B, q^C) \mid q^A \in F^A, q^B \in F^B, q^C \in F^C\} \\ \delta((\mathcal{D}, q^A, q^B, q^C), x) &= \begin{cases} (\mathcal{B}, \delta^A(q^A, x), q^B, q^C) & \text{if } \mathcal{D} = \mathcal{A} \\ (\mathcal{C}, q^A, \delta^B(q^B, x), q^C) & \text{if } \mathcal{D} = \mathcal{B} \\ (\mathcal{A}, q^A, q^B, \delta^C(q^C, x)) & \text{if } \mathcal{D} = \mathcal{C}. \end{cases} \end{aligned}$$

M runs M_A, M_B, and M_C alternating between them and accepts if all three machines have read equal length strings that they accept. (Equal length because every accepting state starts with $\mathcal{A}$.) Hence $L(M) = \mathsf{WEAVE}(A, B, C)$, so it is regular.

To modify M for $\mathsf{2WEAVE}$ only requires changing the definition of F to instead be the set of tuples $(\mathcal{A}, q^A, q^B, q^C)$ where (at least) two out of three of $q^A \in F^A, q^B \in F^B, q^C \in F^C$. Call this M_2. Then $L(M_2) = \mathsf{2WEAVE}(A, B, C)$, so it is regular.

[3]Note that we have assumed, without loss of generality, that the three DFAs have the same alphabet. See Piazza post "Union of Languages with Distinct Characters" for details why this is wlog. Student will not lose points for assuming the alphabets are the same without comment.

...............SCRATCH PAPER..................