Homework 3

COMS W3261, Summer A 2022

This homework is due **Tuesday, 6/14/2022, at 11:59pm EST**. Submit to GradeScope (course code: 2KGDW8).

Grading policy reminder: LaTeX is preferred, but neatly typed or handwritten solutions are acceptable. I recommend using the .tex file for the homework as a template to write up your answers. Your TAs may dock points for indecipherable writing.

Proofs should be complete; that is, include enough information that a reader can clearly tell that the argument is rigorous.

If a question is ambiguous, please state your assumptions. This way, we can give you credit for correct work. (Even better, post on Ed so that we can resolve the ambiguity.)

LaTeX resources.

- The website Overleaf (essentially Google Docs for LaTeX) may make compiling and organizing your .tex files easier. Here's a quick tutorial.
- Detexify is a nice tool that lets you draw a symbol and returns the LaTeX codes for similar symbols.
- The tool Table Generator makes building tables in LaTeX much easier.
- The tool Finite State Machine Designer may be useful for drawing automata. See also this example (PDF) (.tex) of how to make fancy edges (courtesy of Eumin Hong).
- The website mathcha.io allows you to draw diagrams and convert them to LaTeX code.
- To use the previous drawing tools (and for most drawing in LaTeX), you'll need to use the package Tikz (add the command "\usepackage{tikz}" to the preamble of your .tex file to import the package).
- This tutorial is a helpful guide to positioning figures.

1 Problem 1 (12 points)

1. (3 points). What is the language of the grammar G_1 below? Express this language as a set or with a sentence and explain your reasoning.

   (Recall that we can express a CFG briefly by writing the rules. When we do this, we interpret the variable at top left as the start variable, read the other variables off the left-hand side, and infer the terminal alphabet from the other symbols listed in the rules. For example: in this case, the start variable is S, the variable set is $\{S, A, B\}$, and the terminal alphabet is $\{x, y, \&, @\}$.)

$$\begin{aligned} S &\to xAy \\ A &\to xAy \mid B \\ B &\to \&\& \mid @@ \end{aligned}$$

   By examining rules 1 and 2, we see that we generate two equal length substrings of x's and y's on either side of the variable A. Because we must use the first rule, each of these substrings has length at least 1. Then, at some point, we must use the production rule $A \to B$ followed by either $B \to \&\&$ or $B \to @@$. The language of this grammar is thus

$$\{x^n w y^n \mid n \geq 1, w = \&\& \text{ or } @@\},$$

   equivalently, "the language of strings consisting of n repeated x's, for some $n > 0$, then either the substring && or @@, then n repeated y's".

2. (3 points). What is the language of the grammar G_2 below? Express this language as a set or with a sentence and explain your reasoning.

$$\begin{aligned} S &\to 0A \mid 1B \mid 2C \\ A &\to 1F \mid 2E \\ B &\to 0F \mid 2D \\ C &\to 0E \mid 1D \\ D &\to 0 \\ E &\to 1 \\ F &\to 2 \end{aligned}$$

   Because the last three variables always produce exactly one terminal symbol, we can simplify this grammar to the following:

$$\begin{aligned} S &\to 0A \mid 1B \mid 2C \\ A &\to 12 \mid 21 \\ B &\to 02 \mid 20 \\ C &\to 01 \mid 10 \end{aligned}$$

Using this new grammar and testing derivations (perhaps following all possible sequences of rules) we determine that this CFG generates every permutation of $\{0,1,2\}$. This is the language

$$\{012, 021, 102, 120, 201, 210\}.$$

3. (3 points). Design a grammar for the language

$$D = \{0^{2n}1^n0^m1^{2m} \mid m, n \geq 0\}$$

and explain why your grammar produces D. (This language includes such strings as 001011, 0000110000111111111, 000000111011, 001, etc.) You may use the brief representation of grammars (i.e., rules-only) or write out the full 4-tuple.

Because n and m are independent, the first rule of our grammar will replace the start symbol with two separate symbols that will generate $0^{2n}1^n$ and $1^{2m}0^m$. We'll make these sublanguages by building two 0's for every 1 and two 1's for every 0, respectively.

This grammar looks like

$$S \to NM$$
$$N \to 00N1 \mid \epsilon$$
$$M \to 0M11 \mid \epsilon$$

4. (3 points). Design a grammar for the language represented by the regular expression

$$R_1 = (01)^* \cup (10)^*$$

and explain why your grammar produces the same language. You may use the brief representation of grammars (i.e., rules-only) or write-out the full 4-tuple.

Here we can explicitly use our tricks for building context-free grammars that mimic regular expressions, or just read R_1 carefully to understand its meaning ("the language of all strings created by concatenating the string 01 with itself 0 or more times, and all strings created by concatenating the string 10 with itself 0 or more times") and build accordingly.

We'll start with a rule that picks which of $(01)^*$ and $(10)^*$ our string will match, then write two additional rules that generate strings matching these expressions. The grammar is as follows:

$$S \to A \mid B$$
$$A \to \epsilon \mid 01A$$
$$B \to \epsilon \mid 10B$$

Rationale: The goal of this question is to practice interpreting and building context-free grammars.

References: Sipser p. 102 and Lightning Review 5 (CFG definition and deriving strings), Sipser p.105 (figuring out the language of a CFG), and Sipser p.106-107 (tips for building CFGs).

2 Problem 2 (12 points)

1. (6 points.) Prove that the language

$$A = \{a^n b^{2n} a^n \mid n \geq 0\}$$

over the alphabet $\Sigma = \{a, b\}$ is nonregular. Hint: we know two ways of proving nonregularity: using the pumping lemma and proof by contradiction and reasoning from the closure of the regular languages under regular operations.

We'll use the pumping lemma to prove that this language is nonregular.

First, assume for contradiction that A is regular. It follows from our assumption and the pumping lemma that there exists some p such that every string $s \in A$ with $s \geq p$ can be divided into 3 substrings x, y, and z such that (1) xy^iz is in the language for all $i \geq 0$, (2) $|y| > 0$, and (3) $|xy| \leq p$.

We'll choose the contradiction string $w = a^p b^{2p} a^p$ as our s in the statement above. Since $|w| > p$, the pumping lemma tells us that w can be written as xyz where x, y, and z are substrings satisfying the pumping conditions.

By (2) and (3), we know that xy consists of only a symbols and that y consists of one or more a symbols. Thus $xy^2z = a^{p+|y|} b^{2p} a^p$ must be in the language by (1). However, this contradicts the language definition.

Thus our assumption leads to a contradiction, and we can conclude that A does not satisfy the pumping lemma and is nonregular.

Alternatively: we could observe that this language is equivalent to $A \circ B$, where $A = \{a^n b^n \mid n \geq 0\}$ and $B = \{b^m a^m \mid m \geq 0\}$. These languages are equivalent

2. (6 points.) Prove that the language

$$B = \{ww \mid w \in \{0,1\}^* \text{ and } w \text{ contains at least one 0 and at least one 1}\}$$

over the alphabet $\Sigma = \{0, 1\}$ is nonregular. You may use the pumping lemma and/or closure properties.

It's possible to prove this directly using the pumping lemma. However, there's a simpler way using closure under regular operations. First, we define the language

$$C = \{ww \mid w \in \{0,1\}^* \text{ and } w \text{ does not contain both a 0 and a 1}\},$$

which can also be written as the regular expression $(00)^* \cup (11)^*$. We can then observe that $B \cup C = D$, where

$$D = \{ww \mid w \in \{0,1\}^*\}.$$

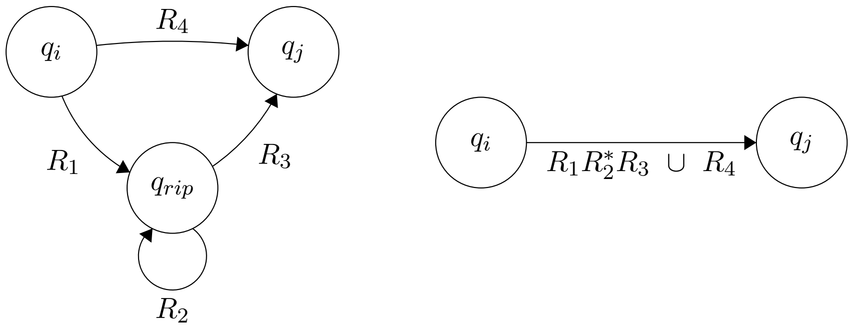
Now, we know that C is regular (it can be expressed as a regular expression.) Suppose for contradiction that B is also regular. Then $B \cup C = D$ is regular because of closure under union. However, this is a contradiction - we already know D is nonregular because we proved it in class using the pumping lemma. Thus B must also be nonregular.

Rationale: The goal of this question is to practice using the pumping lemma and closure properties to show that languages are nonregular.

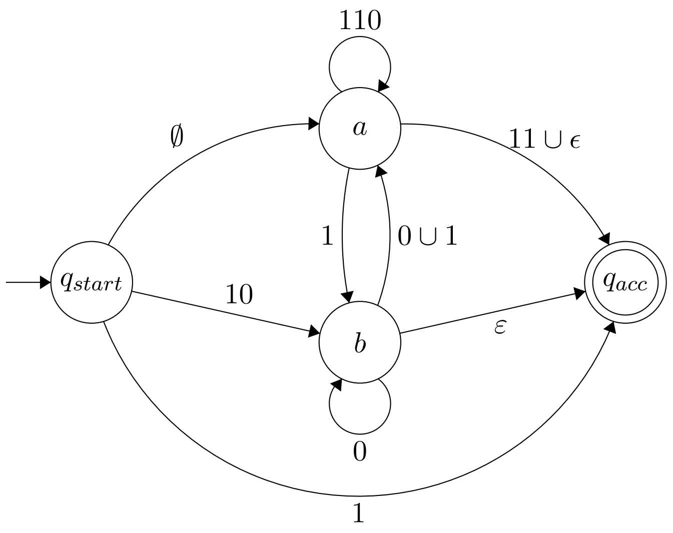
References: Sipser p. 78-79 and Lightning Review 4 (the pumping lemma), Sipser p.80-82 and Lightning Review 5 (using the pumping lemma).

3 Problem 3 (6 points)

1. (6 points). Recall the conversion procedure that we used in class to remove a state from a GNFA without changing its function: for every state pair (q_i, q_j) distinct from our removal state q_{rip}, we rerouted traffic by transforming the first picture below into the second picture:



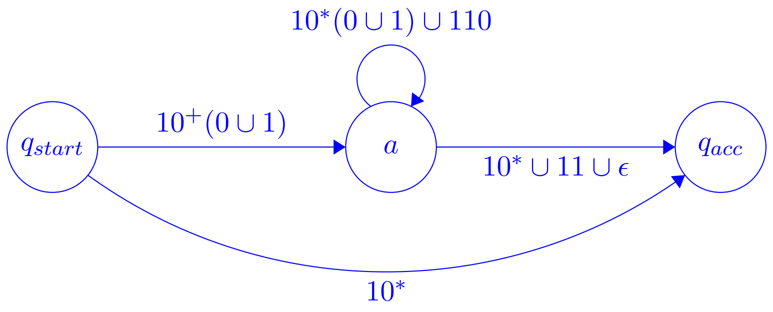
Using our procedure, remove state b from the state diagram below and show the resulting state diagram. If you like, you can use the provided table to compute $R_1R_2^*R_3 \cup R_4$.



State pair (q_i, q_j)	Regular expression $R_1R_2^*R_3 \cup R_4$
(q_{start}, q_{acc})	$100^*\epsilon \cup 1 = 10^+ \cup 1 = 10^*$
(q_{start}, a)	$100^*(0 \cup 1) \cup \emptyset = 10^+(0 \cup 1)$
(a, a)	$10^*(0 \cup 1) \cup 110$
(a, q_{accept})	$10^*\epsilon \cup (11 \cup \epsilon) = 10^* \cup 11 \cup \epsilon$

Above, we've filled in the table with the value of $R_1R_2^*R_3 \cup R_4$ for all pairs (q_i, q_j) (we don't include pairs ending in q_{start} or starting at q_{acc} because these edges are forbidden by the GNFA rules, and we don't include pairs with b because we are removing this state.)

This gives us the four edge labels for the simplified automaton below.



Rationale: This question is intended to provide practice with GNFAs and the recursive simplification of GNFAs to smaller GNFAs that eventually produces an equivalent regular expression.

References: Sipser pp. 70-73 (GNFA rules and definition), Sipser pp.72-74 and video Example 3: DFAs to GNFAs to Regular Expressions (reducing GNFAs by removing states)

4 Problem 4 (7 points)

1. (4 points) In general, applying regular operations to nonregular languages is not guaranteed to result in a nonregular language.

   Suppose A is a nonregular language. Is the complement $\overline{A}$ also nonregular? Explain why or why not. [Hint: think about our proof that the regular languages are closed under complement.]

   Yes, if A is nonregular, $\overline{A}$ is nonregular. To see this, suppose for contradiction that $\overline{A}$ is regular. Then, by closure under complement, A is regular, which is a contradiction.

   This works because the complement operation maps A to $\overline{A}$ and vice versa, while this is not true for binary regular operators $(\cup, \cap, \circ)$ or even other unary regular operators (like $*$).

2. (3 points) Read the statement of the pumping lemma carefully. Does the pumping lemma guarantee that nonregular languages can't be pumped (i.e., that we can show any nonregular language is nonregular by demonstrating that it doesn't satisfy the pumping conditions?)

   No. The pumping lemma tells us something about *all regular languages*, but it doesn't tell us anything in particular about nonregular languages.

   What this means is that, if a language doesn't satisfy the pumping lemma, it is not regular. But it is still possible that some nonregular languages *do* satisfy the PL!

   In fact, there are indeed some nonregular languages that satisfy the PL, and other tools can be used to discover them. Google 'Myhill-Nerode theorem' if you'd like to learn more.

Rationale: The goal of this question is to practice logical thinking and careful interpretation of theorems.

References: Our proof that the regular languages are closed under complement: "If a DFA D recognizes L, then converting all accept states into reject states and vice versa creates a new DFA D' that recognizes $\overline{L}$". Sipser p. 78-79 and Lightning Review 4 (the pumping lemma).

5 Problem 5 (2 Extra Credit Points)

The Myhill-Nerode theorem says that a language L is a regular language if and only if L has a finite number of equivalence classes (i.e., L would not be a regular language if it had an infinite number of equivalence classes).

Consider the language L, defined over the alphabet $\Sigma = \{0\}$:

$$L = \{w | \text{ length of } w \text{ is divisible by } 3\}$$

so strings in the language include $000, 000000, \varepsilon$; and strings not in the language include $0, 00000$.

Is L regular or nonregular? Prove your claim using the Myhill-Nerode theorem. (Hint: You should define all the equivalence classes for L in terms of distinguishing strings, or prove that there are an infinite number of equivalence classes.)

Yes, this is a regular language. Consider the three equivalence classes

$$(\mod 3 \equiv 1, \quad \mod 3 \equiv 2, \quad \mod 3 \equiv 0).$$

Pairs of strings in each of these three classes cannot be distinguished from each other. For example, consider strings x and y such that $x \pmod 3 = 1$ and $y \pmod 3 = 1$. For any extension z, we have $xz, yz \in L$ if and only if $z \pmod 3 = 2$. The situation is similar in the other two equivalence classes. Moreover, the three equivalence classes partition the set of all strings over $\Sigma = \{0\}$.

Because this is a finite number of equivalence classes, by the Myhill-Nerode theorem, L must be regular.