Homework 3 Solutions

COMS W3261, Summer B 2021

This homework is due **Monday, 7/19/2021, at 11:59PM EST**. Submit to GradeScope (course code: X3JEX4).

Grading policy reminder: LaTeX is preferred, but neatly typed or handwritten solutions are acceptable. 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.

The tool http://madebyevan.com/fsm/ may be useful for drawing finite state machines.

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.)

1 Problem 1 (12 points)

1. (6 points.) Prove that the language

$$A = \{w \mid \text{ For all } y \in \{0,1\}^*, w \neq yy\}$$

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

Recall that the complement of a language L over the alphabet $\{0,1\}$ is the language $\{0,1\}^* \setminus L$: that is, all strings over $\{0,1\}$ except those in the language L. The complement of A is

$$\overline{A} = \{w \mid w = yy \text{ for some } y \in \{0,1\}^*\} = \{ww \mid w \in \{0,1\}^*\}.$$

We proved in class that this language is nonregular using the pumping lemma.

Assume for contradiction that A is regular. Because the class of regular languages is closed under complement, this implies that $\overline{A}$ is regular. Because this is a contradiction, our assumption is false and A is nonregular.

(We mentioned in class that the class of regular language is closed under complement. To see this, observe that any DFA D that recognizes a language L can be converted into a DFA that recognizes $\overline{L}$ by turning every reject state into an accept state and vice versa.)

(Proving the statement using the pumping lemma is also fine.)

2. (6 points.) Prove that the language

$$B = \{1^n 0^m 1^n \mid n \geq 0, m \geq 1\}$$

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

Assume for contradiction that B is regular. Thus B satisfies the pumping lemma, and there exists some pumping length p such that for every string $s \in B$ with $|s| \geq p$, s can be split into substrings $s = xyz$ such that $xy^iz \in B$ for all $i \geq 0$, $|y| > 0$ and $|xy| \leq p$.

Consider the string $s = 1^p01^p$, which is in the language B. As $|s| \geq p$, s can be split into substrings x, y, and z satisfying the conditions above if B is regular. We will show that each division of s into substrings fails the conditions of the pumping lemma.

- Case 1: s is divided into x, y, and z such that y does not contain a 0.
  In this case, $xyyz = 1^{p+|y|}01^p$ (if y is a substring of the first p ones) or $xyyz = 1^p01^{p+|y|}$ (if y is a substring of the second p ones). To satisfy our three conditions, it must be true that $|y| > 0$. However, this implies that $xyyz \notin B$ because the two substrings of zeroes are not the same length.
- Case 2: s is divided into x, y, and z such that y contains a 0.
  In this case, $xy^0z = xz$ is a string of all zeroes. Because every string in B has a nonzero number of ones, $xz \notin B$.

Thus there is no way to divide s into substrings x, y, and z in a way that satisfies the conditions of the pumping lemma. This contradicts our assumption that B is regular.

(To simplify these cases, one could also note that the condition $|xy| \leq p$ implies that y must be a substring of the first string of ones in any valid split.)

2 Problem 2 (8 points)

1. (8 points.) Is the language

$$C = \{a^i b^j c^k \mid i, j, k \geq 0; i \leq j\} \cup \{a^i b^j c^k \mid i, j, k \geq 0; j \leq k\} \cup \{a^i b^j c^k \mid i, j, k \geq 0; k \leq i\}$$

   over the alphabet $\Sigma = \{a, b, c\}$ a regular language? Prove your answer.

Yes, this language is regular (even though the three component languages may not be.) To see this, observe that C is a subset of the language

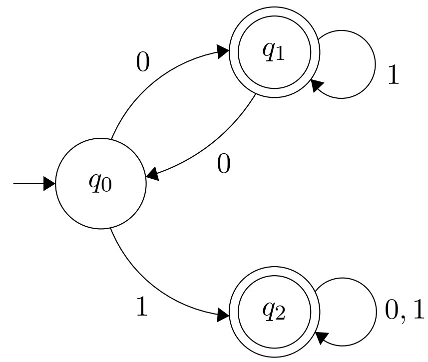
$$L = \{a^i b^j c^k \mid i, j, k \geq 0\}.$$

When is a string in L but not in C? Precisely when the three additional conditions $i \leq j$, $j \leq k$, and $k \leq i$ all fail. This occurs when $i > j > k > i$: in other words, never. Thus $C = L$.

It remains to show that L is regular. To see this, observe that the regular expression $a^* b^* c^*$ evaluates to L.

3 Problem 3 (10 points)

1. (4 points). Convert the DFA below into a GNFA state diagram using the procedure outlined in class. (This procedure is also outlined in the textbook on page 71.)



The procedure outlined in class has several steps:

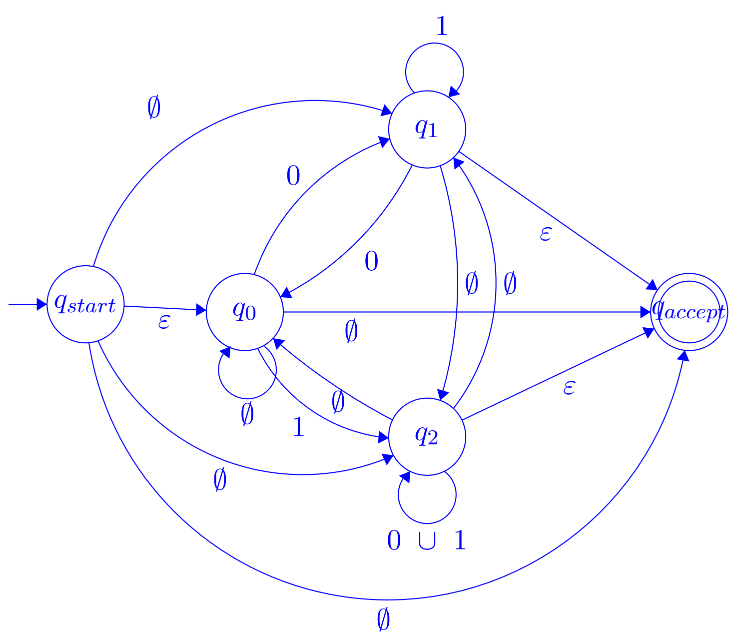
(a) Add a new start state q_{start} connected by an ε-arrow to the old start state.

(b) Add a new accept state q_{accept} with ε-arrows from the old accept state(s).

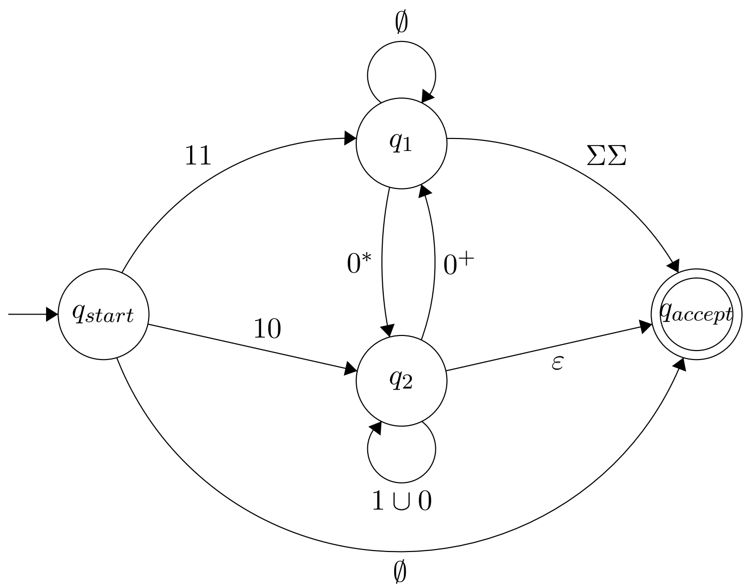
(c) If there are multiple edges connecting any pair of states, merge them with a union operation.

(d) Finally, add $\emptyset$-arrows between every ordered pair of states not connected by a transition, excepting transitions *to* the start state and *from* the accept state. (This includes transitions from states to themselves.)

The resulting state diagram is as follows:



2. (6 points). Use the procedure $CONVERT(G)$ outlined in class (and on page 73 of the textbook) to compute the values of the transitions $\delta'(q_{start}, q_2)$, $\delta'(q_{start}, q_{accept})$, and $\delta'(q_2, q_{accept})$ after removing state q_1 from the GNFA below. Hint: Recall that $\emptyset^*$ evaluates to the language $\{\varepsilon\}$.



The procedure $CONVERT(G)$ reduces the number of states of G one by one. At each step, we select a state $q_{rip} \neq q_{start}, q_{accept}$ and remove it from the state set. For each pair of states (q_i, q_j) such that $q_i \neq q_{accept}, q_j \neq q_{start}$, we set

$$\delta'(q_i, q_j) = R_1 R_2^* R_3 \cup R_4,$$

where $R_1 = \delta(q_i, q_{rip})$, $R_2 = \delta(q_{rip}, q_{rip})$, $R_3 = \delta(q_{rip}, q_j)$, and $R_4 = \delta(q_i, q_j)$.

We designate $q_{rip} = q_1$ and consider each possible pair of states (q_i, q_j) in turn:

1. $(q_i, q_j) = (q_{start}, q_2)$. In this case, $R_1 = 11, R_2 = \emptyset, R_3 = 0^*, R_4 = 10$. We set $\delta'(q_{start}, q_2) = R_1 R_2^* R_3 \cup R_4 = 11\emptyset^* 0^* \cup 10 = 110^* \cup 10$.

2. $(q_i, q_j) = (q_{start}, q_{accept})$. In this case, $R_1 = 11, R_2 = \emptyset, R_3 = \Sigma\Sigma, R_4 = \emptyset$. We set $\delta'(q_{start}, q_{accept}) = R_1 R_2^* R_3 \cup R_4 = 11\emptyset^*\Sigma\Sigma \cup \emptyset = 11\Sigma\Sigma$.

3. $(q_i, q_j) = (q_2, q_{accept})$. In this case, $R_1 = 0^+, R_2 = \emptyset, R_3 = \Sigma\Sigma, R_4 = \varepsilon$. We set $\delta'(q_2, q_{accept}) = R_1 R_2^* R_3 \cup R_4 = 0^+\emptyset^*\Sigma\Sigma \cup \epsilon = 0^+\Sigma\Sigma \cup \epsilon$.

4 Problem 4 (12 points)

1. (3 points). What is the language of the grammar G_1 below? Here S, A, and B are the variables and 0 and 1 are the terminals. Explain your reasoning.

$$S \to 1A1$$
$$A \to S \mid B$$
$$B \to 0B \mid \varepsilon$$

By examining rules 1 and 2, we see that we generate two equal length strings of 1's on either side of the variable A until we use the production rule $A \to B$. At that point, we generate some number of 0's before producing a terminal empty string. The language of this grammar is thus

$$\{1^n 0^m 1^n \mid n \geq 1, m \geq 0\}.$$

2. (3 points). What is the language of the grammar G_2 below? Here A and B are the variables and x, y, and z are the terminals. Explain your reasoning.

$$A \to xAx \mid yAy \mid zAz \mid B$$
$$B \to x \mid y \mid z \mid \varepsilon$$

The first rule produces pairs of x's, y's, and z's in any order on either side of the variable A until we use the production rule $A \to B$. At this point, we finish our string with an x, y, z, or empty string in the middle. The result is the language of all palindromes on the alphabet $\Sigma = \{x, y, z\}$. (To see this, recall that w^R denotes the reverse of w and observe that even-length palindromes have the form ww^R for some string $w \in \Sigma^*$ and that odd-length palindromes have the form wxw^R, wyw^R, or wxw^R for some string $w \in \Sigma^*$.)

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

$$D = \{a^i b^j c a^j b^i \mid i, j \geq 1\}$$

and explain why your grammar produces D.

Define the grammar $G_3 = \{\{A, B\}, \{a, b, c\}, R, A\}$. The set of rules R is

$$A \to aAb \mid aBb$$
$$B \to bBa \mid bca$$

The first rule generates two equal-length substrings of a's and b's on either side of the variable B. These substrings have length at least one, as guaranteed by the production rule $A \to aBa$. The second rule generates two equal-length substrings of b's and a's on either side of the terminal c. The final production rule $B \to bcb$ ensures that these substrings have length at least 1.

4. (3 points). Design a grammar for the language

$$L = I(saw \cup met \cup loved)(the \cup a)(very)^*(large \cup tiny \cup red)(frog \cup dog)$$

and explain why your grammar produces L. (You can treat each word as a single terminal symbol.)

Define the grammar $G_4 = \{\{S, A, B, C, D, E\},$
$\{I, saw, met, loved, the, a, very, large, tiny, red, frog, dog\}, R, S\}$. The set of rules R is

$$\begin{aligned}
&S \rightarrow IABCDE \\
&A \rightarrow saw \mid met \mid loved \\
&B \rightarrow the \mid a \\
&C \rightarrow veryC \mid \epsilon \\
&D \rightarrow large \mid tiny \mid red \\
&E \rightarrow frog \mid dog
\end{aligned}$$

This language is a long concatenation, as is its grammar. Variables S, A, B, D, and E are straightforward replacements with terminals. The variable C is replaced with some number of concatenations of the terminal 'very.'