Instructions

Upload a single file to Gradescope for each group. All group members’ names and PIDs should be on each page of the submission. Your assignments in this class will be evaluated not only on the correctness of your answers, but on your ability to present your ideas clearly and logically. You should always explain how you arrived at your conclusions, using mathematically sound reasoning. Whether you use formal proof techniques or write a more informal argument for why something is true, your answers should always be well-supported. Your goal should be to convince the reader that your results and methods are sound.

For questions that only ask for diagrams, justifications are not required but highly recommended. It helps to show your logic in achieving the answers and partial credit can be given if there are minor mistakes in the diagrams.

Reading Sipser Sections 1.3 and 1.4

Key Concepts DFA, NFA, equivalence of DFA and NFA, regular expressions, equivalence of DFA and regular expressions, regular languages, closure of the class of regular languages under certain operations, the Pumping Lemma, pumping length, proofs of nonregularity
Problem 1 (10 points)

For each of the regular expressions below, give two examples of strings in the corresponding language and give two examples of strings not in the corresponding language.

a. \((000 \cup 1)^* (0 \cup 111)^*\)

b. \((1 \cup 01 \cup 10)^*\)

c. \(\varepsilon^* \cup (0^* \varepsilon) \cup 1\)

Problem 2 (10 points)

In this problem, you will convert the following DFA into a regular expression, using the GNFA construction from Lemma 1.60 in the textbook (Sipser section 1.3, page 69). Note that this DFA recognizes the language of strings that contain an odd number of ones.

![DFA Diagram]

Note: The DFA diagram shows a state labeled q0 with transitions labeled 0 and 1 leading to state q1, and state q1 with transitions labeled 0 leading back to state q0.
Create a 4-state GNFA that is equivalent to the above DFA, using the structure suggested by the proof of Lemma 1.60. To help you get started, we have provided the structure of the GNFA, so you only need to fill in the table below.

![DFA Diagram]

<table>
<thead>
<tr>
<th>Edge</th>
<th>Regular expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>$st \rightarrow q_1$</td>
<td></td>
</tr>
<tr>
<td>$st \rightarrow q_2$</td>
<td></td>
</tr>
<tr>
<td>$st \rightarrow acc$</td>
<td></td>
</tr>
<tr>
<td>$q_0 \rightarrow q_0$</td>
<td>0</td>
</tr>
<tr>
<td>$q_0 \rightarrow q_1$</td>
<td>1</td>
</tr>
<tr>
<td>$q_0 \rightarrow acc$</td>
<td></td>
</tr>
<tr>
<td>$q_1 \rightarrow q_0$</td>
<td>1</td>
</tr>
<tr>
<td>$q_1 \rightarrow q_1$</td>
<td>0</td>
</tr>
<tr>
<td>$q_1 \rightarrow acc$</td>
<td></td>
</tr>
</tbody>
</table>
(b) Create a 3-state GNFA equivalent to the GNFA in part (a), after removing the state \( q_1 \).
Again, we have provided the structure of this GNFA, so you only need to fill in the table below.
Try to simplify each edge’s regular expression as much as possible (for example, using the results of problem 3a).

\[ \begin{array}{|c|c|}
\hline
\text{Edge} & \text{Regular Expression} \\
\hline
st \rightarrow q_0 & \\
\hline
st \rightarrow acc & \\
\hline
q_0 \rightarrow q_0 & \\
\hline
q_0 \rightarrow acc & \\
\hline
\end{array} \]

(c) Create a 2-state GNFA equivalent to the GNFA in part (b), after removing the state \( q_0 \).
Again, we have provided the structure of this GNFA, so you only need to fill in the table below.
Try to simplify the resulting regular expression as much as possible.

(Note: A good way to check your answer is to convince yourself that the regular expression you find in this step is equivalent to the DFA we originally started with.)

\[ \begin{array}{|c|c|}
\hline
\text{Edge} & \text{Regular Expression} \\
\hline
q_0 \rightarrow q_3 & \\
\hline
\end{array} \]
Problem 3 (10 points)

Convert the following DFA into a regular expression. For full credit, you must show a GNFA for each step (five states, four states, three states, two states). You can choose whether to label the edges or use a table.

Problem 4 (10 points)

Given a string $w = x_1 x_2 ... x_n$ its reversal is $w^R = x_n ... x_2 x_1$. For example: $(abbc)^R = cbba$. Given a language $L \subseteq \Sigma^*$ we define its reversal is $L^R = \{ w^R : w \in L \}$. In this problem, we will prove in two different ways that if $L$ is a regular language then $L^R$ is also a regular language. In other words, regular languages are closed under reversal.

(a) First prove that regular languages are closed under reversal using the fact that a language is regular if and only if a DFA or NFA recognizes it. Let $L$ be a regular language and let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA such that $L = L(M)$. Construct an NFA (or DFA) $M'$ such that $L^R = L(M')$ and give a proof that your construction is correct.

(b) Next, prove that regular languages are closed under reversal using the fact that a language is regular if and only if some regular expression describes it. Let $L$ be a regular language and let $R$ be a regular expression such that $L = L(R)$. We need to show that there exists a regular expression $R'$ such that $L^R = L(R')$. Hint: To do this, give a proof by induction on the size of the regular expression $R$. The size of a regular expression is the number of symbols used in it. Use Definition 1.52 (Sipser p.64) and notice that there are three base cases (items 1-3) and three inductive cases (because items 4-6 are defined recursively in terms of smaller regular expressions).
Problem 5 (10 points)

a. Prove that the language $L = \{0^m1^n1^{|m \geq n}\}$ is not regular.

b. Prove that the following language $L$ (which contains all binary strings that are not palindromes) is not regular: $L = \{w \in \{0, 1\}^* | w \neq w^R\}$

Problem 6 (10 points)

According to the statement of the pumping lemma, every regular language has a pumping length $p$ such that every string $x \in L$ can be pumped if $|x| \geq p$. The minimum pumping length is the smallest number $p$ such that $p$ is a pumping length for $L$. For each of the following languages, what is the minimum pumping length $p$? Justify your answer; this means you must also explain why $p-1$ is not a pumping length for $L$.

a. $110^*11$

b. $(01^*0^*) \cup 1111$

c. $L = \{w \in \{0, 1\}^* | w$ has an equal number of occurrences of the substrings 01 and 10 $\}$