CS46 Homework 3

This homework is due at 11:59pm on Sunday, February 17. Submission instructions are at the end of this file.

This is an individual homework. It’s ok to discuss approaches at a high level, but you should not reveal specific details of a solution. Your write-up should be your own: do not share it, and do not read other people’s write-ups. If you use any out-of-class references (anything except class notes, the textbook, or asking Lila), then you must cite these in your README.md file. Please refer to the course webpage or ask me any questions you have about this policy.

The main learning goal of this homework is to develop the skills to design, understand, and analyze NFAs and regular expressions.

Part 1 — These problems should be completed and submitted on Automata Tutor. You are allowed three attempts at each problem. I recommend that you first try to solve the problems on paper, then use the site to debug your solutions.

1. (Sipser 1.7d) Construct an NFA with two states for the language \( \{0\} \) over the alphabet \( \{0, 1\} \).

2. (Sipser 1.7c) Construct an NFA with six states for the language \( \{ w \mid w \text{ contains an even number of 0s, or contains exactly two 1s} \} \) over the alphabet \( \{0, 1\} \).

3. Let \( \Sigma = \{a, b\} \) and consider languages:
   \[
   L_1 = \{ w \mid \text{every } a \text{ in } w \text{ is immediately followed by a } b \} \\
   L_2 = \{ w \mid w \text{ contains an odd number of } b \text{s} \}
   \]
   Construct:
   (a) an NFA recognizing \( L_1 \cup L_2 \).
   (b) an NFA recognizing \( L_1 L_2 \).

4. Let \( \Sigma = \{a, b\} \) and let \( L_3 = \{ w \mid w \text{ contains an even number of } a \text{s and an odd number of } b \text{s} \} \).
   Construct an NFA recognizing \( L_3^* \).
   (Hint: You can use the construction of Theorem 1.49 to get a correct answer, but this NFA will be capable of being simplified. Try describing \( L_3^* \) in English first.)

5. (Sipser 1.16)
   (a) Construct a DFA equivalent to the following NFA:

   ![NFA Diagram]


---

1If you want to use late days on this assignment, you will need to submit solutions to these problems via github. The automata tutor site has only one deadline.
(b) Construct a DFA equivalent to the following NFA:

\[ \text{\begin{align*}
q_1 & \xrightarrow{a} q_2 \\
q_2 & \xrightarrow{a, b} q_3 \\
q_3 & \xrightarrow{b} q_3
\end{align*}} \]

6. Write a regular expression for the language

\[ \{ w \mid w \text{ contains exactly two } a \text{ s or at least two } b \text{ s} \} \]

over the alphabet \( \Sigma = \{a, b\} \).

7. Let \( \Sigma = \{0, 1\} \) and \( L = \{ w \mid w \text{ contains the substring } 0ab0 \text{ or } 1ab1 \text{ where } a, b \in \Sigma \} \).

   (a) Construct a DFA that recognizes \( L \).

   (It is strongly recommended that you plan on paper first!)

   \textbf{For fun}: Take a screenshot of your masterpiece, email it to Lila, treasure it forever.

   (b) Construct an NFA that recognizes \( L \).

   Your NFA should have substantially fewer states than your DFA. (Phew!)

8. Let \( L = \{ w \mid \text{if } w \text{ contains an } a, \text{ then it contains at least three } a \text{ s in a row} \} \) over \( \Sigma = \{a, b\} \).

   So for example, \( L \) contains \( abaa, bbaabaabaa, \varepsilon \), and \( bb \). \( L \) does not contain \( baa, abaaba, \) or \( bbba \).

   • Construct a DFA that recognizes \( L \).
   • Construct an NFA that recognizes \( L \).
   • Construct a regular expression that recognizes \( L \).
Part 2 — These problems should be typeset in \LaTeX\ and submitted using github.

9. Understanding regular expressions.
   For each of the following regular expressions over $\Sigma = \{a, b\}$, explain in English what language they describe. (Show your thought process.)
   
   (a) $b^*a(b^*a)^*$
   (b) $(a^*\emptyset b \cup ab \cup b^*\emptyset a)(b \cup \emptyset)$
   (c) $\epsilon \cup a(ba)^* \cup b(ab)^*$

10. Writing regular expressions.
   Give regular expressions for the languages accepted by NFAs $M_1$ and $M_2$. (Hint: use Lemma 1.60.)

11. C++ comments
   In C++, it is possible to write a comment as a string beginning with ‘/*’ and ending with ‘*/’. In between, it can contain any characters, including ‘/’ and ‘*’, as long as it does not contain the substring ‘*/’ before the end of the string. For example, ‘/*ab/ab/c/b****ac*/’ is a valid comment, but ‘/*cab/baaaa***/a//bb*/’ is not.
   
   Construct a regular expression for the language $L$ of this style of well-formed C++ comments over the alphabet $\Sigma = \{a, b, c, *, /\}$.

   $$L = \{w \mid w = /*x*/, \text{ where } x \in \Sigma^* \text{ does not contain the substring ‘*/’}\}$$

   You should explain your reasoning/show your work. A formal proof is not required, but you should be able to justify (at a high level) why your regular expression is correct.

12. Regular expression identities.
   Let $R$ and $S$ be regular expressions. Prove or disprove the following “identities”. To prove the identity argue that a string in the language defined on the left-hand side is in the language defined on the right-hand side, and vice versa. To disprove the identity, give a small counterexample string with real examples of regular expressions for $R$ and $S$.

   (a) $(R^*)^* = R^*$
   (b) $(R \cup S)^* = R^* \cup S^*$
   (c) $(R*S*)^* = (R \cup S)^*$
13. **Extra credit (for the detail-oriented).**

Write a regular expression for the language

\[ \{ w \mid w \in \{ a, b \}^* \text{ contains an odd number of } bs \text{ and an even number of } as \} \]

Either start from a DFA for this language and show your work for converting the DFA to a regular expression, or justify how you came up with the regular expression and why it is correct. A formal proof is not required, but you should explain at a high level what is happening in your regular expression.

14. **Extra credit (self-referentially cool).**

We have a set of rules that describe how to build regular expressions, given an alphabet \( \Sigma \) and some additional symbols, ‘∅’, ‘∪’, ‘◦’, ‘∗’, ‘(’, and ‘)’. Consider the language \( L \) over the alphabet given by \( \Sigma \) with these added symbols, defined as:

\[ L = \{ w \mid w \text{ is a regular expression over } \Sigma \} \]

Prove that \( L \) is not regular.

(Thinking dizziness warning: be careful not to think yourself in circles with this one! A cool corollary of this claim is that there is no regular expression that matches all regular expressions.)

**Submission**

Solutions to part 1 are collected on the automatatutor site.

Solutions to part 2 should be written in this tex file, above.

Once you have finished parts 1 and 2, please fill out the short survey in the file README.md. Your answers to the questions in this document will not impact your grade, but provide valuable feedback on the difficulty of the assignment and common sources of confusion or roadblocks. Your responses will help make the course better.

Once you have edited the files, you should publish your changes:

```
git add hw03.tex README.md
git commit -m "completed homework 3, yippee!"
git push
```

If you make changes to files after you push and want to submit these changes, repeat the add-commit-push loop again to update your submission. You can push as often as you like; only the most recent push will be graded.