Problem Set 9

Due Date .... November 8, 2022
Name .... **Alex Ojemann**
Student ID **109722375**
Collaborators .... **None**

Contents

1 Instructions

- The solutions **should be typed**, using proper mathematical notation. We cannot accept hand-written solutions. (See this short intro to LaTeX plus other resources on Canvas.)
- You should submit your work through the **class Canvas page** only. Please submit one PDF file, compiled using this LaTeX template.
- You may not need a full page for your solutions; pagebreaks are there to help Gradescope automatically find where each problem is. Even if you do not attempt every problem, please submit this document with no fewer pages than the blank template (or Gradescope has issues with it).
- You are welcome and encouraged to collaborate with your classmates, as well as consult outside resources. You must **cite your sources in this document. Copying from any source is an Honor Code violation. Furthermore, all submissions must be in your own words and reflect your understanding of the material.** If there is any confusion about this policy, it is your responsibility to clarify before the due date.
- Posting to **any** service including, but not limited to Chegg, Reddit, StackExchange, etc., for help on an assignment is a violation of the Honor Code.
- You **must** virtually sign the Honor Code (see Section 2). Failure to do so will result in your assignment not being graded.

2 Honor Code (Make Sure to Virtually Sign)

- My submission is in my own words and reflects my understanding of the material.
- Any collaborations and external sources have been clearly cited in this document.
- I have not posted to external services including, but not limited to Chegg, Reddit, StackExchange, etc.
- I have neither copied nor provided others solutions they can copy.

(I agree to the above, Alex Ojemann). □

3 Standard 22: Computability Classification

Problem 1. Classify the following languages as below, and justify your answer:

- (RE) recursive enumerable (r.e.),
- (CO-RE) co-r.e. (i.e. the complement is r.e.),
- (BOTH) both r.e. and co-r.e., or
- (NEITHER) neither r.e. nor co-r.e.

Note this means showing two things each: L is r.e. or not, and $\sim L$ is r.e. or not.

(a) $L = \{M\#x\#y \mid M \text{ halts on } x \text{ and loops on } y\}$

Proof. M halts on x is r.e. and M loops on y is not r.e. If we were to construct a two-tape Turing Machine, the tape with the looping problem on it is not r.e., so L itself must not be r.e.

$\sim L = \{M\#x\#y \mid M \text{ loops on } x \text{ and halts on } y\}$

M loops on x is not r.e. and M halts on y is r.e. If we were to construct a two-tape Turing Machine, the tape with the looping problem on it is not r.e., so $\sim L$ must not be r.e., so L is not co-r.e.

Therefore L is neither r.e. nor co-r.e. □

(b) $L = \{M\#x\#y \mid M \text{ accepts } x \text{ and rejects } y\}$

Proof. M accepts on x is r.e. and M rejects on y is also r.e. If we were to construct a two-tape Turing Machine, both tapes on it are r.e., so L itself must be r.e.

$\sim L = \{M\#x\#y \mid M \text{ rejects or loops on } x \text{ and accepts or loops on } y\}$

M rejects or loops on x is not r.e. and M accepts or loops on y is also not r.e. If we were to construct a two-tape Turing Machine, both tapes are not r.e., so $\sim L$ must not be r.e., so L is not co-r.e. □

4 Novel Construction

Problem 2. For language $L \subseteq \Sigma^*$ and symbol $a \in \Sigma$ we define $\textsc{clip}_a(L)$ as the set

$$\textsc{clip}_a(L) = \{x \in \Sigma^* \mid xa \in L\}.$$

For example if $L = \{0, 0110, 101, 10010\}$ then $\textsc{clip}_0(L) = \{\varepsilon, 011, 1001\}$.

4.1 Standard 11: Novel Construction

(a) Fix $a \in \Sigma$. Show that $\textsc{clip}_a$ is a closure property of regular languages. That is, show that $\textsc{clip}_a(L)$ is regular for all $a \in \Sigma$ and all regular $L \subseteq \Sigma^*$. [**Note:** To earn full credit for Standard 11, your construction should be correct. However, you do not have to prove that your construction is correct to earn credit for Standard 11.]

Answer. Suppose L is regular. So there exists a DFA $M = (Q_M, \Sigma, \delta_M, s_M, F_M)$ such that L(M) = A. We construct an NFA $N = (Q_N, \Sigma, \delta_N, S_N, F_N)$ accepting $\textsc{clip}_a(L)$, as follows.

We let $Q_N := Q_M$ plus any dead states (states that are not final and transition to themselves on any input) added as a result of the changes described in the transition function.

The alphabet for N is the same alphabet as for M.

$s_M = s_N$

We define δ_N as follows. δ_N contains every transition in δ_M, except all transitions in δ_M that result in a final state are revised. For any transition in δ_M that leads to a final state and takes in input a, δ_N has the same transition but it takes in ϵ instead of a. In the given example, this accounts for 0, 0110, and 10010 in L becoming ϵ, 011, and 1001 in $\textsc{clip}_0(L)$ respectively. For any transition in δ_M that leads to a final state and takes in any input in Σ other than a, δ_N has the same transition but it leads to a dead state that is added to Q_N instead of a final state. In the given example, this accounts for 101 in L not having and counterpart in $\textsc{clip}_0(L)$.

$F_N = F_M$ □

4.2 Standards 2/3: Proofs

(b) Carefully prove that your construction from part (a) works. That is, again suppose L is regular, and let K be the language accepted by your construction. Carefully prove that $K = \textsc{clip}_a(L)$.

Proof. There are two relevant cases of strings in L:

Case 1: x is in L and ends with a. Let y have the same characters as x but without the a on the end. y will transition through N, the NFA equivalent to the DFA M that accepts L, the same way x transitions through M until the second to last state. From there y transitions to a final state because δ_N has a transition to a final state on ϵ. N accepts y as $\textsc{clip}_a(L)$ does.

Case 2: x is in L and doesn't end with a. Let y have the same characters as x but without the last character on the end. Since δ_N has a transition to a dead state in place of the transitions in δ_M to final states where the input is any character in Σ other than a, neither x nor y would be accepted, as is the case in $\textsc{clip}_a(L)$.

Thus, the language generated by N is equal to $\textsc{clip}_a(L)$. □