Yet Another Assignment
CS203

Disclaimer: I stake no claim on the originality of the problems.

1 Regular Languages

1. Consider languages over some alphabet Σ. Argue whether the following statements are correct.

   (a) If L_1 is not regular and $L_1 \subseteq L_2$, then L_2 is not regular.

   (b) If $L_1 \subseteq L_2$ and L_2 is not regular, then L_1 is not regular.

   (c) If L_1 is not regular, then its complement $\overline{L_1}$ is not regular.

   (d) If L_1 is regular, then $L_1 \bigcup L_2$ is regular for any language L_2.

   (e) If L_1 and L_2 are not regular, then $L_1 \bigcap L_2$ is not regular.

   A nice easy set of problems to start your journey.

2. Consider an arbitrary language L over some alphabet Σ. We can define the language of *suffixes* of the strings in L as

   $$\texttt{SUFF}(L) = \{u \in \Sigma^* \mid vu \in L \text{ for some } v \in \Sigma^*\}.$$

   Prove that if the language L is regular, then so is $\texttt{SUFF}(L)$.

3. Show that if the language L over the alphabet Σ is regular, so is

   $$\frac{1}{q}(L) = \{x \mid \text{ for some } y \text{ such that } |x| = (q-1)|y|,\ xy \text{ is in } L\}.$$

   By now, you should be comfortable with these kind of problems and begin to differentiate between the beauty of intuition and the tediousness of formal description.

4. Let Σ_1 and Σ_2 be disjoint alphabets. Let $L_1 \subseteq \Sigma_1^*$ and $L_2 \subseteq \Sigma_2^*$. Define $\texttt{shuffle}(L_1, L_2)$ as the set of strings $w \in (\Sigma_1 \bigcup \Sigma_2)^*$ with the following properties –

(a) If all symbols in w that belong to Σ_1 are replaced by ϵ, the resulting string belongs to L_2.

(b) If all symbols in w that belong to Σ_2 are replaced by ϵ, the resulting string belongs to L_1.

Intuitively, you take two strings, w_1 from L_1 and w_2 from L_2, and shuffle the symbols of w_1 and w_2 (like you shuffle two halves of a pack of cards) to get strings in $\texttt{shuffle}(L_1, L_2)$.

Show that if L_1 and L_2 is regular, so is $\texttt{shuffle}(L_1, L_2)$.

As an example, if $L_1 = \{ab\}$ *and* $L_2 = \{c, cd\}$*, then*

$$\texttt{shuffle}(L_1, L_2) = \{cab, acb, abc, cdab, cadb, cabd, acdb, acbd, abcd\}$$

2 Context Free Languages

5. For some language L over alphabet Σ, define $\texttt{graft}(L)$ as

$$\{w \mid w = xyz,\ x, y, z \in \Sigma^*,\ xz \in L \text{ and } y \in L\}.$$

Is $\texttt{graft}(L)$ context-free?

6. Let L_1 and L_2 be languages over the alphabet Σ. Define the *right quotient* of L_2 by L_1 as $L_1 \backslash L_2$

$$\{w \in \Sigma^* \mid \exists y \in L_1 \text{ and } yw \in L_2\}.$$

If L_1 is regular and L_2 is context-free, is $L_1 \backslash L_2$ necessarily context-free?

7. Show that each of the following languages is not context-free.

(a) $\left\{ww^R w \mid w \in \{0,1\}^*\right\}$. ($w^R$ *is of course the reverse of* w).

(b) $\left\{0^n 1^n 2^i \mid 0 \leq i \leq n\right\}$.

3 Turing machines

8. Construct a Turing machine that does not accept the language

$$\{a^n b^n c^n \mid n \geq 0\}.$$

You are required to give a complete description of the Turing machine.

This question is to convince me, and indeed yourself, that if asked nicely (or maybe paid enough), you can actually write out the entire description of a Turing machine.

9. Construct a Turing machine that accepts the language

$$\left\{ww \mid w \in \{a, b\}^*\right\}.$$

You are **not** required to give a complete description of the transition function. Just describe how your Turing machine will work in the various states that you have defined.

This question will illustrate the power that the Turing machines gain by being allowed to go back and forth over its input. Remember that our Turing machines are deterministic.

10. Given that the languages L_1 and L_2 are recursive, prove that the following two languages are recursive as well –

- $L_1 L_2$.
- $(L_1)^*$.

The previous question should have given you enough insight to tackle the first of the languages. A little bit of ingenuity will help you solve the second problem.

11. Given that the languages L_1 and L_2 are recursively enumerable, prove that the following two languages are recursively enumerable as well –

- $L_1 L_2$.
- $(L_1)^*$.

These languages will seriously test your mettle. The main problem lies in the fact that the machines for L_1 and L_2 need not halt on all inputs, so the corresponding solutions for the recursive languages will not work.

12. Recall from problem 6 the *right quotient* of L_2 by L_1, denoted by as $L_1 \backslash L_2$. If L_1 and L_2 are recursively enumerable, is $L_1 \backslash L_2$ also recursively enumerable?

4 Firing Squad Synchronization Problem

The name of the problem comes from an analogy with real-world firing squads: the goal is to design a system of rules according to which a general can command a squad of troops to fire, and cause them to all fire their guns simultaneously. More formally, the problem concerns an array of finite state machines, called "cells", arranged in a line, such that at each time step each machine transitions to a new state as a function of its previous state and the states of its two neighbors in the line. For the firing squad problem, the line consists of a finite number of cells, and the rule according to which each machine transitions to the next state should be the same for all of the cells interior to the line, but

the transition functions of the two endpoints of the line are allowed to differ, as these two cells are each missing a neighbor on one of their two sides.

Among the states of the cells, there are three three distinguished states: "active", "quiescent", and "firing", and the transition function must be such that a cell that is quiescent and whose neighbors are quiescent remains quiescent. Initially, at time $t = 0$, all cells are in quiescent states except for the cell at the far left (the general), which is active. The goal is to design a set of states and a transition function such that, no matter how long the line of cells is, there exists a time t such that every cell transitions to the firing state at time t, and such that no cell transitions to the firing state prior to time t.

Promise that even at gun-point you would firmly yet politely refuse to consult internet for solutions, and I guarantee you countless hours of frustration and joy as you grope around in the dark for a solution.