CS 332: *Theory of Computation* — Prof. Mark Bun
Boston University — January 28, 2025

Homework 2 – Due Tuesday, February 4 at 11:59 PM

Reminder Collaboration is permitted, but you must write the solutions *by yourself without assistance*, and be ready to explain them orally to the course staff if asked. You must also identify your collaborators and write "Collaborators: none" if you worked by yourself. Getting solutions from outside sources such as the Web or students not enrolled in the class is strictly forbidden.

Problems There are 6 required problems and two bonus problems.

1. For each of the following languages, (i) give a plain English description of the language, and (ii) give two examples of strings in the language and two examples of strings that are in Σ^* but outside the language.

   (a) $L_1 = \{w \in \{0,1\}^* \mid w = w^R\}$

   (b) $L_2 = \{x\mathsf{aa}y \mid x, y \in \{\mathsf{a}, \mathsf{b}\}^*\} \cup \overline{\{x\mathsf{bb}y \mid x, y \in \{\mathsf{a}, \mathsf{b}\}^*\}}$

   (c) $L_3 = \{x\#y\#z \mid x, y, z \in \{0,1\}^* \text{ and } x + y = z\}$. (Here, you should view x, y, z as the binary representations of non-negative integers, with the most significant bit on the left.)

2. For each of the following languages, (i) describe the language using set-builder notation and union/intersection/complement/reverse/concatenation operations (the notation used in Problem 1), and (ii) give two examples of strings in the language and two examples of strings that are in Σ^* but outside the language.

   (a) L_4 = the set of all strings over alphabet $\{\mathsf{a}, \mathsf{b}, \mathsf{c}\}$ that have length at least 3 and have b as their third symbol.

   (b) L_5 = the set of all strings over alphabet $\{0, 1\}$ that either start with 0 and have odd length, or start with 1 and have even length.

   (c) L_6 = the set of all strings over alphabet $\{\mathsf{a}, \mathsf{b}\}$ that contain neither aaab nor baaa as substrings.

3. Which of the following statements are true or false, for all alphabets Σ? For each, provide either a proof or a counterexample.

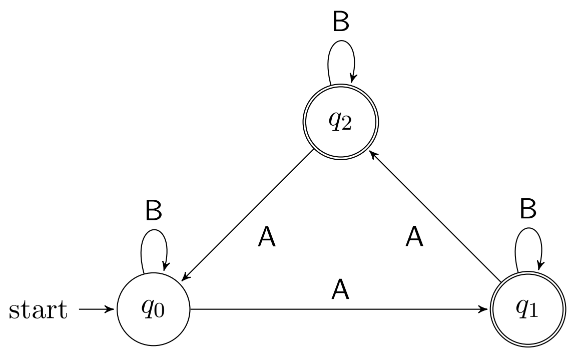
   (a) For all strings $x, y, z \in \Sigma^*$, we have $|x \circ (yz)^R| = |x| + |y| + |z|$. (Recall that $\circ$ denotes concatenation.)

   (b) For all languages $L_1, L_2 \subseteq \Sigma^*$, we have $(L_1 \circ L_2)^R = L_2^R \circ L_1^R$.

   (c) For all languages $L \subseteq \Sigma^*$, we have $L \circ \{\varepsilon\} = L \circ \emptyset$.

   (d) For all languages $L_1, L_2, L_3 \subseteq \Sigma^*$, we have $L_1 \circ (L_2 \cap L_3) = (L_1 \circ L_2) \cap (L_1 \circ L_3)$.

4. Consider the following state diagram of a DFA M using alphabet $\Sigma = \{\mathsf{A}, \mathsf{B}\}$.



   (a) What is the start state of M?
   (b) What is the set of accept states of M?
   (c) Give a formal description of the machine M (i.e., as a 5-tuple).
   (d) What is the language recognized by M? (Hint: It has a simple English description using modular arithmetic.)

5. This problem will be autograded using AutomataTutor. Visit `https://automata-tutor.live-lab.fi.muni.cz/` and register for an account using your name and `@bu.edu` email address (it is important for recording grades that the information for your account match the information on the course list provided by the university). We'll the link to enroll for this course on Piazza.

   Give state diagrams of DFAs with as few states as you can recognizing the following languages. You may assume that the alphabet in each case is $\Sigma = \{0, 1\}$.

   (a) $L_1 = \{w \mid w \text{ begins with 1 and ends with 00}\}$.
   (b) $L_2 = \{w \mid w \text{ contains at most three 1's}\}$.
   (c) $L_3 = \{w \mid w \text{ contains the substring 010}\}$.

   Give state diagrams of NFAs with as few states as you can recognizing the following languages. You may assume that the alphabet in each case is $\Sigma = \{0, 1\}$.

   (d) $L_4 = \{xyz \mid$ string x consists only of 0's, string y consists only of 1's, and string z consists only of 0's and contains at least one 0$\}$.
   (e) $L_5 = \{w \mid w \text{ contains substrings 100 and 10 which do not overlap}\}$.

6. Draw (and include in the PDF you submit to Gradescope) state diagrams of DFAs with as few states as you can recognizing the following languages. You may assume that the alphabet in each case is $\Sigma = \{0, 1\}$.

   (a) $L_6 = \{w \mid w$ is a string of the form $x_1y_1x_2y_2 \ldots x_ny_n$ for some integer $n \geq 0$ such that $x_i, y_i \in \{0, 1\}$ and $x_i = y_i$ for all $i\}$.
   (b) $L_7 = \{w \mid w$ represents a binary number that is congruent to 2 modulo 3$\}$. In other words, this number minus 2 is divisible by 3. The number is presented starting from the most significant bit and can have leading 0s.

7. **Bonus Problem.** In this problem, you'll explore how to formally define and analyze properties of strings. Let Σ be an alphabet. A *string* over alphabet Σ is defined recursively as follows. It is either the empty string ε (base case) or takes the form ax where $a \in \Sigma$ and x is itself a string (recursive case). The *length* of the string x can then be defined as follows:

$$|x| = \begin{cases} 0 & \text{if } x = \varepsilon \\ 1 + |z| & \text{if } x = az \text{ for } a \in \Sigma \text{ and } z \in \Sigma^*. \end{cases}$$

Similarly, the concatenation of two strings x, y can be defined as

$$xy = \begin{cases} y & \text{if } x = \varepsilon \\ a(zy) & \text{if } x = az \text{ for } a \in \Sigma \text{ and } z \in \Sigma^*. \end{cases}$$

These definitions let us give inductive proofs of properties of strings. For instance, consider the following claim, which says that the length of the concatenation of two strings is the sum of the lengths of those strings.

Claim. For any two strings x, y, we have $|xy| = |x| + |y|$.

To prove this, let x and y be arbitrary strings. We will prove this by induction on the length $n = |x|$. As our base case, suppose $n = 0$. Then $x = \varepsilon$, so

$$\begin{aligned} |xy| &= |\varepsilon y| && \text{(assumption on } x\text{)} \\ &= |y| && \text{(definition of concatenation)} \\ &= |\varepsilon| + |y| && \text{(definition of length)} \\ &= |x| + |y| && \text{(assumption on } x\text{)} \end{aligned}$$

as we wanted. Now assume as our inductive hypothesis that the claim is true for length n; we want to show it is true for length $n + 1$. In this case, we have $x = az$ for some string z of length n. So

$$\begin{aligned} |xy| &= |a(zy)| && \text{(definition of concatenation)} \\ &= 1 + |zy| && \text{(definition of length)} \\ &= 1 + |z| + |y| && \text{(inductive hypothesis)} \\ &= |x| + |y| && \text{(definition of length).} \end{aligned}$$

(a) Given a string $x \in \{0,1\}^*$, let $\text{sort}(x)$ denote the string obtained by sorting the characters of x so that all 0's appear before all 1's. For example, $\text{sort}(10110) = 00111$. Give a recursive definition of the sort function along the lines of what we did with length above.

(b) Give an inductive proof that $|\text{sort}(x)| = |x|$ for every string $x \in \{0,1\}^*$.

(c) Prove that $\text{sort}(\text{sort}(x)) = \text{sort}(x)$ for every string $x \in \{0,1\}^*$. Hint: Instead of trying to prove this directly by induction, it might be useful to introduce some auxiliary recursively-defined functions and prove statements about those.

8. **Bonus Problem.** Show that for any natural number n, the language $MOD_n = \{w \mid w \text{ represents a binary number that is divisible by } n\}$ is regular. The number is presented starting from the most significant bit and can have leading 0's.