HW1 - Closure Properties of Regular Languages and NFAs

COMP361 — Suhas Arehalli

Spring 2025

Instructions

As with previous homeworks,

- Deadlines are posted on the course website.
- Solutions should be turned in as a typeset pdf, preferably using LaTeX. I recommend a tool like Overleaf.
- Assignments are individual, and you should not share solutions with other students. However, you are free to discuss problems within the bounds of the academic integrity policy in the syllabus, and should indicate in your assignment the people you got assistance from (including instructors, peers, and preceptors!).
- Proofread your work before submission, and consult the proof style guide to make sure your submissions are both correct and clear.
- Graded problems will be indicated by an asterisk (*) next to the problem number. You are still expected to solve all of the problems.

Questions

1. *(Sipser 1.31/32/36)*
   (a) For languages A and B, let the *perfect shuffle* of A and B be the language

   $$L_1 = \{w \in \Sigma^* \mid w = a_1 b_1 \dots a_k b_k \text{ where } a_1 \dots a_k \in A \text{ and } b_1 \dots b_k \in B, \text{ each } a_i, b_i \in \Sigma\}$$

   Show the regular languages are closed under the perfect shuffle operation.

   (b) *For languages A and B, let the *shuffle* of A and B be the language

   $$L_1 = \{w \in \Sigma^* \mid w = a_1 b_1 \dots a_k b_k \text{ where } a_1 \dots a_k \in A \text{ and } b_1 \dots b_k \in B, \text{ each } a_i, b_i \in \Sigma^*\}$$

   Note the difference with a perfect shuffle — each a_i, b_i is a string, not a symbol!
   Show that the regular languages are closed under the shuffle operation.

(c) For a language L, let the *reverse* of L be

$$L^R = \{w^R \mid w \in L\}$$

Recall that for $w = w_1 \dots w_k$, $w^R = w_k \dots w_1$, $w_i \in \Sigma, \forall 1 \leq i \leq k$. Show that the regular languages are closed under the reverse operation.

2. *(Extending Sipser 1.25/1.37)* Let

$$\Sigma_3 = \left\{ \begin{bmatrix} a \\ b \\ c \end{bmatrix} \mid a, b, c \in \{0, 1\} \right\}$$

and

$$B = \{w \in \Sigma_3 \mid \text{ the bottom row of } w \text{ is the sum of the top two rows}$$

That is, you should consider the sequence of first, second, and third elements of each column vector as binary strings, and the binary string constructed from the third elements should represent the sum of the binary strings represented by the first and section. See the example in Sipser to confirm your understanding.

(a) *Prove that B^R is regular. Conclude from problem 1c that B must be regular.

***HINT**: Consider how you'd add binary strings, and modify that technique to verify that the third row contains the correct digit in that position. How much state/memory do you need to maintain?*

(b) Read the informal definition of a **Finite State Transducer (FST)** in Sipser 1.24. Construct a formal definition of an FST as a 5-tuple and a formal definition of a computation in an FST, following the definitions of a DFA we saw earlier. Assume the input and output alphabets can be distinct (labeled Σ and Γ), and there are no accept/final states.

***HINT**:* Following Sipser, the transition function should be of the form $\delta : Q \times \Sigma \to Q \times \Gamma$

(c) *Using the previous parts, construct a reverse binary addition FST. Define an appropriate input alphabet Σ_2 than encodes two binary strings a and b of any arbitrary finite length n, and have the transducer output a binary string c such that $c^R \equiv a^R + b^R \pmod{2^n}$. Equivalently, $c^R = (a^R + b^R) \; \% \; 2^n$, or consider this addition on n-bit unsigned integers with no handling of overflow - pick your favorite way to ground this!