Dominik Wawszczak
Student ID Number: 440014
Group Number: 1

Languages, Automata and Computation II
Assignment 2

Problem 1

For any $w \in \Sigma^*$, let $P_w(x)$ denote the polynomial obtained by feeding w into the automaton. By definition, this function is a polynomial in one variable x. Let X be the set defined in the problem statement.

We consider two cases:

1. There exists a word $w \in \Sigma^*$ such that $P_w(x)$ is not the zero polynomial.

   In this case, let $R(P_w)$ represent the set of roots of $P_w(x)$. Since $P_w(x)$ is not identically zero, $R(P_w)$ is finite. Furthermore, X must be a subset of $R(P_w)$; otherwise, there would exist some $x \in X$ for which $P_w(x) \neq 0$, contradicting the definition of X. Thus, X is finite.

2. For every word $w \in \Sigma^*$, $P_w(x)$ is the zero polynomial.

   In this scenario, $P_w(x) = 0$ holds for all $x \in \mathbb{Q}$ and every $w \in \Sigma^*$. Consequently, $X = \mathbb{Q}$.

From these cases, we conclude that X is either finite or equal to $\mathbb{Q}$, completing the proof.

Problem 2

This problem was solved in collaboration with Kacper Bal and Mateusz Mroczka.

Let $\mathcal{A}$ denote the automaton from the problem statement, with dimension d and alphabet Σ. For any $a \in \Sigma$, let $M_a \in \mathbb{Q}^{d \times d}$ represent the matrix corresponding to the transition function for symbol a.

We will construct a polynomial automaton $\mathcal{B}$ that defines a function $g : \Sigma^* \to \mathbb{Q}$. The goal is for this function to satisfy the property that, for any word $w \in \Sigma^*$, $g(w) = f(w_{\text{lex}})$, where w_{lex} is w sorted lexicographically, and f is the function computed by $\mathcal{A}$.

Assume $\Sigma = \{a_1, a_2, \ldots, a_k\}$, where $a_1 <_{\text{lex}} a_2 <_{\text{lex}} \ldots <_{\text{lex}} a_k$. Denote the initial state of $\mathcal{A}$ as $q_{\text{ini}} \in \mathbb{Q}^{1 \times d}$, and the final mapping as $q_{\text{fin}} \in \mathbb{Q}^{d \times 1}$. According to the definition of a weighted automaton, for any $w \in \Sigma^*$ such that $w = w_1 w_2 \ldots w_n$, it holds that $f(w) = q_{\text{ini}} \cdot M_{w_1} \cdot M_{w_2} \cdot \ldots \cdot M_{w_n} \cdot q_{\text{fin}}$. Therefore, it should also hold that

$$g(w) \;=\; q_{\text{ini}} \cdot M_{a_1}^{\#_{a_1}(w)} \cdot M_{a_2}^{\#_{a_2}(w)} \cdot \ldots \cdot M_{a_k}^{\#_{a_k}(w)} \cdot q_{\text{fin}}.$$

The automaton $\mathcal{B}$ will have dimension $d^2 \cdot k$. The first d^2 coordinates will correspond to M_{a_1} raised to the power of the number of letters a_1 read so far, the next d^2 coordinates to M_{a_2}, and so on. In the initial state of $\mathcal{B}$, we store the identity matrix I for each letter a_i in the alphabet.

For any $a \in \Sigma$, the transition matrix simulates matrix multiplication by M_a on the appropriate coordinates, while the other coordinates remain unchanged. Note that this requires only linear mappings.

Once the values $M_{a_1}^{\#_{a_1}(w)}, M_{a_2}^{\#_{a_2}(w)}, \ldots, M_{a_k}^{\#_{a_k}(w)}$ have been computed and stored in the state, we need to calculate the final value

$$q_{\text{ini}} \cdot M_{a_1}^{\#_{a_1}(w)} \cdot M_{a_2}^{\#_{a_2}(w)} \cdot \ldots \cdot M_{a_k}^{\#_{a_k}(w)} \cdot q_{\text{fin}}.$$

This is a polynomial involving $d^2 \cdot k$ variables, which are the elements of the matrices stored in the state. Hence, the automaton $\mathcal{B}$ can compute this, making it the only non-linear transition.

Since $\mathcal{A}$ is also a polynomial automaton (as linear functions are polynomials), we can use the algorithm presented in the lecture to determine whether $\mathcal{A}$ and $\mathcal{B}$ are equivalent. If they are, the function f is commutative; otherwise, it is not, which concludes the proof.

Problem 3

This problem was solved in collaboration with Kacper Bal and Mateusz Mroczka.

Since weighted automata are a special case of polynomial automata, we only need to show that if a polynomial automaton $\mathcal{A}$ computes a function $f : \Sigma^* \to \mathbb{F}$, then there exists a weighted automaton $\mathcal{B}$ that computes the same function.

Let d be the dimension of $\mathcal{A}$. We set the dimension of $\mathcal{B}$ to $D = |\mathbb{F}|^d$, which represents the number of possible states $\mathcal{A}$ can be in. The invariant for $\mathcal{B}$ is that it will always have exactly one coordinate equal to 1, with the rest set to 0. This coordinate corresponds to the state of $\mathcal{A}$. In the initial state of $\mathcal{B}$, represented as a vector in $\mathbb{F}^{1\times D}$, the coordinate corresponding to the initial state of $\mathcal{A}$ is set to 1, with all others set to 0.

For any $a \in \Sigma$, let $M_a \in \mathbb{F}[x_1, x_2, \ldots, x_d]^d$ be the transition polynomials for the letter a. Since for every $(x_1, x_2, \ldots, x_d) \in \mathbb{F}^d$, the result of applying the transition M_a to $(x_1, x_2, \ldots, x_d)$ is uniquely determined, we define the transition function of $\mathcal{B}$ as a $D \times D$ matrix. The entry at position (i, j) is 1 if and only if applying M_a to $(x_1, x_2, \ldots, x_d)$ results in $(y_1, y_2, \ldots, y_d)$, where i corresponds to $(x_1, x_2, \ldots, x_d)$ and j corresponds to $(y_1, y_2, \ldots, y_d)$; otherwise, the entry is 0. This construction maintains the invariant of $\mathcal{B}$ and ensures correct transitions.

The final mapping of $\mathcal{B}$ is a vector in $\mathbb{F}^{D\times 1}$. The entry at position i is $(x_1, x_2, \ldots, x_d) \cdot q_{\text{fin}}$, where i corresponds to $(x_1, x_2, \ldots, x_d)$, and q_{fin} is the final mapping of $\mathcal{A}$. With this construction, automata $\mathcal{A}$ and $\mathcal{B}$ compute the same function, thus completing the proof.