1 First Order Logic

**Exercise 1 (Some Examples).** Provide first-order transductions that represent the following functions:

1. The function that maps a word \( w \) to its reverse.
2. The function that maps a word \( ab^n a \) to \( (ab)^n (ba)^n \), \( ba^n b \) to \( (ba)^n (ab)^n \), and \( w \) to \( w \) otherwise.
3. The function that sorts the letter in a word.

**Exercise 2 (Some Non-Examples).** Prove that the following functions are not representable by first-order transductions:

1. The function that maps \( w \) to \( a \) if \( w \) is of odd length and \( b \) otherwise.
2. The function that maps \( w \) to \( w^2 \) if \( w \) is of even length and \( w^3 \) otherwise.
3. Given a non-trivial group \( (G, \cdot) \), the function that maps a word \( w \) to its image in \( G \).

2 Lambda Terms

**Exercise 3 (Extra Functions).** Prove that the lambda-calculus becomes strictly more expressive when adding the following functions:

1. The trace operator \( \text{trace} : (A \times B \to A \times B) \to (B \to 1 + B) \) that computes the trace of a function.
2. The fold operator \( \text{fold} : (Q \times \Sigma \to Q) \to Q \times \Sigma^* \to Q \).
3 Blind again

Exercise 4 (Pumping lemma for regular functions). Let \( f \) be a regular function. Prove that there exists \( N \geq 0 \) such that for all \( w \in A^* \) with \(|w| \geq N\), there exist \( v_0, v_1 \in A^*, u \in A^+, n \geq 0, \alpha_0, \ldots, \alpha_n \in B^*, \beta_1, \ldots, \beta_n \in B^+ \) such that \( w = v_0uv_1 \) and

\[
f(v_0u^{X+1}v_1) = \alpha_0\beta_1^X\alpha_1\ldots\beta_n^X\alpha_n, \quad \text{for all } X \geq 0.
\]

\( \triangleright \) Hint 1
\( \triangleright \) Solution 1 (Self-contained proof)

Exercise 5 (Prefixes is not blind). Our goal is to prove that the function \( \text{prefixes} \) is not computable by a polyblind function.

1. Let \( f_1, \ldots, f_n \) be regular functions. Is it possible that \( f_1(w)f_2(w)\cdots f_n(w) \) computes a factor of \( \text{prefixes}(w) \) with a number of hashes that tends to \( +\infty \) as \(|w|\) grows?

2. Let \( f \) be a regular function. Is it possible that \( f(w)^{|w|} \) computes a factor of \( \text{prefixes}(w) \) with a number of hashes that tends to \( +\infty \) as \(|w|\) grows?

3. Using an induction on the polyblind depth and leveraging the pumping lemma of regular functions prove that the function \( \text{prefixes} \) is not polyblind.

\( \triangleright \) Hint 2
\( \triangleright \) Hint 3
\( \triangleright \) Hint 4
\( \triangleright \) Solution 2 (Self-contained proof)

4 Cheat-Sheet

Definition 1 (Regular functions). A function \( f : A^* \to B^* \) is called a regular function if there exists a two-way deterministic finite automaton with outputs (2DFT) that computes \( f \). Such an automaton has a finite set of states \( Q \) with a distinguished initial state \( q_0 \), a transition function function over an extended input alphabet \( \Sigma = A \cup \{←, →, ↑\} \) to delimit the endpoints of the input word. The transition function has the following type \( \delta : \Sigma \times Q \to Q \times \{←, ↑, →\} \). That is, it can read a letter, change state, move left \( ← \), right \( → \), stay in place \( ↓ \), or exit the computation \( ↑ \).

The output of the automaton is guided by a production function \( \lambda : Q \times Σ \to B^* \). That is, for every state and current letter, the automaton can produce some word in \( B^* \).

A run of a 2DFT is a sequence of configurations \( (q_i, p_i) \) where \( q_i \) is the ith state of the computation, and \( p_i \) is the ith position of the head over an extended input word \( \vdash w \vdash \). The run starts in the initial state \( q_0 \), and the initial position \( p_0 = 0 \) (so on the left letter \( ← \)). The unique run is defined inductively as one expects using the transition function \( \delta \). Note that a regular function should guarantee that the run does not go out of bounds nor loops forever.

The production of a run \( \rho \) of a 2DFT is the word obtained by concatenating the outputs produced by each transition.

Definition 2 (The prefixes function). The prefixes function is defined inductively as follows. \( \text{prefixes}(w) \) is the list of non-empty prefixes of \( w \) separated by hashes. For instance, \( \text{prefixes}(abc) = a\#ab\#abc \).

Definition 3 (Composition by substitution). Let \( f \) be a function from \( Σ^* \) to \( \{1, \ldots, k\}^* \), and \( g_1, \ldots, g_k \) be functions from \( Σ^* \to Γ^* \). The composition by substitution of \( f \) by \( g_1, \ldots, g_k \) is the function

\[ \text{cbs}(f, g_1, \ldots, g_k)(w) = \text{map}(λx.g_k(w))(f(w)). \]

Definition 4 (Polyblind functions). The class of polyblind functions is defined as the smallest class of functions containing the regular functions and closed under composition by substitution. The polyblind depth of a function is the smallest \( k \) such that the function can be obtained by composition by substitution of nesting depth at most \( k \).
References

A Hints

**Hint 1** (Exercise 4 Idempotent transition monoid). Look at idempotent words in the transition monoid of the function $f$.

**Hint 2** (Exercise 5 For the first). Note that $f_1(w) \ldots f_n(w)$ is of linear output size.

**Hint 3** (Exercise 5 For the second). Notice that if $f(w)$ outputs a word with at least two hashes, then $f(w)^2$ cannot be a factor of prefixes$(w)$. If it has only one hash, then $f(w)^X = (f(w)^2)^{X/2}$ and we conclude similarly for even $X$'s.

**Hint 4** (Exercise 5 For the third). The statement is clear for regular functions. Let us now consider a function obtained by a composition by substitution. Leveraging the pumping lemma for regular functions, conclude that some factor of prefixes$(w)$ should be computed by a function lower polyblind depth.
B Solutions

Solution 1 (Solution to Exercise 4). One version of the full proof is given by [Dou23, Proposition 2.16].

Solution 2 (Solution to Exercise 5). A complete proof of the result can be found in [Dou23, Proposition 3.14].