EXERCISE SHEET 1: TURING MACHINES AND FINITE STATE AUTOMATA.

Let us first get more familiar with the definition of a Turing machine.

Exercise 1 (The contrarian machine)**.** *Define a Turing machine with an alphabet $\Sigma = \{0, 1\}$, which outputs the negation of any binary input string $x \in \{0, 1\}^*$. Try to write down the exact set of states and the transition function.*

Exercise 2 (The copying machine)**.** *Define a Turing machine with an alphabet $\Sigma = \{0, 1, \#\}$, where # symbolizes a delimiting symbol, which outputs $x\#x$ for any binary input string $x \in \{0, 1\}^*$. This time, it suffices to provide a higher-level definition, no need to specify the whole transition function exactly.*

We could define Turing machines with varying architectures. The depth of the original definition is that such modifications would still implement the exact same set of computable functions.

Exercise 3 (Turing machine with one-way infinite tape)**.** *Consider an alternative definition of a Turing machine which instead of a bi-infinite tape consists of a one-way infinite tape, indexed by natural numbers $\mathbb{N}$. In this case, the input is always written at the start of the tape (first symbol being at position 0), and the head is initially also at position 0. Show that for any partial computable function φ (implemented by some classical Turing machine) there exists a TM with a one-way infinite tape that implements it.*

Let $\mathcal{T}$ be a Turing machine with alphabet Σ and let $L \subseteq \{0, 1\}^*$ (we sometimes call L a language). We say that $\mathcal{T}$ decides L if $\varphi_{\mathcal{T}}$ is the characteristic function of L. I.e., $\mathcal{T}$ halts for all inputs, it outputs 1 whenever $x \in L$, and it outputs a 0 whenever $x \notin L$.

Exercise 4 (Turing machines have unbounded memory)**.** *Let $L = \{0^n 1^n \mid n \in \mathbb{N}^+\} \subseteq \{0, 1\}^*$. Show that there exists a Turing machine that decides L.*

Not every change of the TM architecture leads to a computational model of the same strength. The aim of the next exercises is to get familiar with finite-state automata, show that they are "computationally weaker" than Turing machines, and lastly to show that we can essentially decompose each Turing machine into a finite-state automaton interacting with an infinite tape.

A *finite-state automaton (FSA)* is a tuple $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ where

- Q is a finite set of states,
- Σ is a finite input alphabet,
- $\delta : Q \times \Sigma \to Q$ is the transition function,
- $q_0 \in Q$ is the initial state,
- $F \subseteq Q$ is the set of accepting states.

The transition function δ extends to a function $\delta^* : Q \times \Sigma^* \to Q$ defined recursively as follows:

- For the empty word ε, define $\delta^*(q, \varepsilon) = q \quad$ for all $q \in Q$.
- For any word $x \in \Sigma^*$ and symbol $a \in \Sigma$, define $\delta^*(q, xa) = \delta\big(\delta^*(q, x), a\big)$.

We say that $\mathcal{A}$ accepts a language $L \subseteq \Sigma^*$, if $\delta^*(q_0, x) \in F$ if and only if $x \in L$.

Exercise 5 (Tracing a finite-state automaton)**.** *Consider the finite-state automaton* $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ *with* $Q = \{q_0, q_1, q_2\}$*,* $\Sigma = \{0, 1\}$ *and* $F = \{q_0\}$ *and whose transition function is given by* $\delta(q_i, 1) = q_{(i+1) \bmod 3}$*,* $\quad \delta(q_i, 0) = q_i$ *for all* $i \in \{0, 1, 2\}$*.*

1. *Compute* $\delta^*(q_0, x)$ *for the inputs*

$$x = 10101, \quad x = 111, \quad x = 001011.$$

2. *Describe explicitly the language accepted by* $\mathcal{A}$*.*

Exercise 6 (Finite-state automata are weaker than Turing machines)**.** *Show that there exists no finite state automaton that accepts the language* $L = \{0^n 1^n \mid n \in \mathbb{N}^+\} \subseteq \{0, 1\}^*$*. Hint*[1]

We can extend a FSA to produce outputs, by simply extending the above definition by an additional function $\lambda : Q \times \Sigma \to \Delta^*$ where Δ is some output alphabet. We call such machines *Mealy machines.*

Exercise 7 (Turing machine can be decomposed into a finite-state automaton and a tape)**.** *Show that each Turing machine* $\mathcal{T} = (Q, \Sigma, \delta)$ *can be essentially decomposed into a Mealy machine* $\mathcal{A}_\mathcal{T}$ *(with a potentially infinite stream of input symbols) and an external tape. What are the states of* $\mathcal{A}$*? And the input and output alphabet? How does the stream of input symbols depend on the outputs of* $\mathcal{A}$*?*

[1] Gur SFN unf bayl n svavgr ahzore bs fgngrf. Jung unccraf gb gur genwrpgebel bs fgngrf gung ner ivfvgrq sbe n ybat rabhtu vachg fgernz 0^n? [encoded in ROT13]