Exercise 1: Interpret Turing Machines (4.5 + 1.5 points)

Consider the following Turing machine \( M \) given as state diagram. An edge label \( (A \rightarrow Y, H) \) with \( A \subseteq \Gamma, Y \in \Gamma, H \in \{L, R\} \) means that if a symbol \( X \in A \) is read on the tape, it is replaced with \( Y \) and the read/write head of the Turing machine is moved according to \( H \). In this notation the symbol \( Y \) can be omitted, which means the Turing machine leaves the symbol \( X \in A \) it currently reads unchanged. If \( M \) reads \( \sqcup \in \Gamma \) this means that the current cell on the tape is empty. \( M \) halts in a state if there is no eligible transition from that state. An input string written on \( M \)'s tape before being started is accepted, if and only if \( M \) halts in \( q_a \). Note that we consider a Turing machine with an infinite tape in both directions, which is equally mighty and often easier to work with.

(a) Simulate \( M \) with inputs \( s_1 = abccba \), \( s_2 = aacceaa \), \( s_3 = abccab \) until it halts. Give the sequence of configurations which the simulation of \( M \) on \( s_i \) produces. You may omit configurations where no symbol is replaced. Which of these inputs are in \( L(M) \)?

(b) Give the language \( L(M) \) of strings that are accepted by \( M \).

Solution

(a) See Figure 1.

(b) \( L(M) = \{wuw^R | w \in \{a,b\}^*, u \in \{c\}^*, |u| = |w| \geq 1 \} \)
Figure 1: Configurations of the Turing machine \( M \) with inputs \( s_1 = abccba, s_2 = aacccaa, s_3 = abccab \).
Exercise 2: Construct Turing Machines (4+3 points)

(a) Draw a state diagram of a Turing machine $M$ recognizing the language $\{a^n b^n a^n \mid n \geq 0\}$ over the alphabet $\Sigma = \{a, b\}$.

(b) Simulate your Turing machine with the input strings $s_1 = aabbaa$ and $s_2 = abaaa$. Give the sequence of configurations for these inputs. You may omit configurations where no symbol is replaced.

Solution

(a) See Figure 2.

(b) See Figure 3.

Figure 2: Turing machine that recognizes the language $\{a^n b^n a^n \mid n \geq 0\}$.

Figure 3: Configurations of our Turing machine of exercise 2 (a) with inputs $s_1 = aabbaa$, $s_2 = abaaa$.
Exercise 3: RE and Set Operations (3+3 points)

The concept of Turing machines is very powerful. Indeed there are ‘universal’ Turing machines that can emulate other Turing machines $M$ on an input $s$, when given a codification of $M = (Q, \Sigma, \Gamma, \delta, q_0, q_a, q_r)$ together with the input $s$. Such universal Turing machines can base their own behavior on the behavior of the Turing-machines they simulate.

A Turing machine can even do several tasks ‘simultaneously’ due to the concept of multitape Turing machines which have several tapes and read/write heads that can operate independently on their respective tapes. Such a multitape Turing machine can in turn be emulated by a single tape Turing machine.

Now consider the set of Recursive Enumerable Languages (RE). $L \in \text{RE}$, iff there exists a Turing machine $M$ such that:

- $M$ with input string $s$ halts in the accepting state, if and only if $s \in L$ (in short: $L = L(M)$). Based on the insight above, show the following.

(a) If $L_1, L_2 \in \text{RE}$ then $L_1 \cup L_2 \in \text{RE}$.

(b) If $L_1, L_2 \in \text{RE}$ then $L_1 \cap L_2 \in \text{RE}$.

**Hint:** Argument informally what a Turing machine for the respective language should do. Keep in mind that for any Language $L \in \text{RE}$ the corresponding Turing machine $M$ with $L(M) = L$ does not necessarily need to halt for input $s \notin L$.

**Solution**

(a) Since $L_1, L_2 \in \text{RE}$, there exist Turing machines $M_1, M_2$ with $L(M_1) = L_1$ and $L(M_2) = L_2$. We construct a Turing machine $M_\cup$ that recognizes $L_1 \cup L_2$ as follows. Let $s$ be the input for $M_\cup$. Then $M_\cup$ uses two tapes to simulate $M_1$ on $s$ on tape one and $M_2$ on $s$ on tape two, simultaneously or in an alternating step-by-step fashion (first a step of $M_1$ then one of $M_2$ and so on). If either $M_1$ or $M_2$ halts in the accepting state, $M_\cup$ also enters the accepting state and halts. Thus $M_\cup$ halts in $q_a$ on input $s$, if and only if $s \in L(M_1) = L_1$ or $s \in L(M_2) = L_2$.

(b) Let $M_1, M_2$ be defined as before. We construct $M_\cap$ that recognizes $L_1 \cap L_2$. First $M_\cap$ simulates $M_1$ with input $s$. If $M_1$ accepts, then $M_\cap$ simulates $M_2$ on input $s$. If $M_2$ accepts as well, then $M_\cap$ halts in $q_a$. If either $M_1$ or $M_2$ halt in a non-accepting state, $M_\cap$ does so as well. If $s \in L_1$ and $s \in L_2$ both machines $M_1$ and $M_2$ accept and so does $M_\cap$. If either $M_1$ or $M_2$ reject $s$, so does $M_\cap$ which is correct because $s \notin L_1 \cap L_2$. Due to the way we defined it, $M_\cap$ may run infinitely if either $M_1$ or $M_2$ do so. But in this case $s \notin L_1 \cap L_2$ and therefore allowed.