Tutorial 2

Exercise 1 (compulsory)

Can 3-tape Turing machines recognize a larger class of languages than 2-tape Turing machines?

- If your answer is positive, then provide an example of a language recognizable by a 3-tape Turing machine but not recognizable by any 2-tape Turing machine.
- If your answer is negative, give a short but complete argument supporting your claim.

Solution:
The answer is negative. Turing machines with 3 tapes recognize exactly the same class of languages as 2-tape Turing machines. Let L be a language recognized by a given 3-tape Turing machine. Thanks to Theorem 3.13 on page 151 we know that there is a single tape Turing machine recognizing L and any single tape machine can be trivially simulated by 2-tape Turing machine (we simply do not use the second tape). Hence the language L is recognized also by a Turing machine with 2 tapes.

Exercise 2 (compulsory)

One can imagine an extension of the Turing machine model such that the transition function is of the type

$$\delta : Q \times \Gamma \to Q \times \Gamma \times \{\mathrm{L}, \mathrm{R}, \mathrm{V}\}$$

where V denotes that the head is to be immediately (in one step) moved to the leftmost tape-cell. Is this extension more powerful than the original Turing machine model? Give precise arguments for your claim, at least on the implementation level description (see page 159, first paragraph).

Solution:
The answer is that the extended model is not more powerful than the original one. In order to prove this, we will show how the extended Turing machine can be simulated by our standard model of a Turing machine.

The idea is that the standard model can do exactly the same transitions as the extended one, except for the V-move. In order to simulate the V-move (which happens in one step), we need to move the head to the beginning of the tape one by one. Moreover, a Turing machine cannot determine if its head is located at the leftmost tape cell unless we use some trick. We will introduce a new tape symbol $\#$ which will mark the very first cell on the tape.

Let M be the extended Turing machine that we wish to simulate. The simulation proceeds as follows:

On input w:

1. Shift the input string w one cell to the right and write $\#$ in the leftmost cell.
2. Place the head of the machine on the cell immediately to the right of the symbol $\#$.
3. Perform the same steps as M, but replace V-transitions by moving the head to the left (one by one) until the machine reaches the end marker $\#$. Then move the head one cell to the right and continue the simulation of the machine M.

This is just one example of how this problem can be solved. Your solution, though different from the above presented one, can be still correct.

Exercise 3 (compulsory)

Which of the definitions of nondeterministic Turing machines given below is correct? Read them carefully before you answer.

1. A nondeterministic Turing machine M accepts input x if there exists more than one computation of M on x such that M halts in the state q_{accept}.

2. A nondeterministic Turing machine M accepts input x if no computation of M on x halts in the state q_{reject}.
3. A nondeterministic Turing machine M accepts input x if there exists a computation of M on x such that M halts in the state q_{accept}.
4. A nondeterministic Turing machine M accepts input x if there is exactly one accepting computation of M on x.
5. A nondeterministic Turing machine M accepts input x if for every computation of M on x we have that M halts in state q_{accept}.

Complete the definition below and be as precise as possible:

> A nondeterministic Turing machine M is called *decider* if ...

Solution:
Only the third definition is correct. The definition of a nondeterministic decider is as follows:

> A nondeterministic Turing machine M is called *decider* if for any given input $x \in \Sigma^*$ and any computation of M on x the machine halts (either in q_{accept} or q_{reject}).

Exercise 4 (compulsory)

Theorem 3.21 states that a language is recognizable if and only if it can be enumerated by an enumerator. Why cannot we use the somewhat simpler enumeration algorithm below for showing one half of the theorem, namely that if a language L is recognizable, then it can be enumerated? As before $s_1, s_2, s_3, \ldots$ is a list of all strings in Σ^*, and M denotes a Turing machine recognizing the language L.

> E = "Ignore input.
>
> 1. i:=1
> 2. Run M on s_i.
> 3. If M accepted, then print s_i.
> 4. i:=i+1; goto step 2."

Solution:
The machine M is a recognizer for L, and a recognizer need not halt on every input. This means that M may loop on some s_k, and our enumerator then becomes unable to output strings s_i whose index i is larger than k.

Exercise 5 (compulsory)

What is the statement of Church-Turing Thesis?

Solution:
Church-Turing Thesis: "The Turing machine model captures exactly the informal notion of algorithm."

Exercise 6 (optional and challenging)

Problem 3.10 on page 162 and Problem 3.13 on page 163 (both are not very difficult) and Problem 3.11 on page 163 (more challenging).