ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich
Networked Systems Group (NSG)

HS 2026

Prof. L. Vanbever / J. Mégret
based on Prof. R. Wattenhofer's material

Discrete Event Systems

Exercise Sheet 1

General Instructions

- In this exercise session, you will often have to draw automata. It is easiest (and most instructive for the written exam) to draw them by hand. However, there are several online tools available to draw automata, *e.g.* `https://web.cs.ucdavis.edu/~doty/automata/`or `http://madebyevan.com/fsm/`.
- Our exercises are designed to consolidate (and deepen) your knowledge of the lecture contents, and to prepare you for the written exam. We strictly advise you to take the time for solving them yourself. We may not cover all exercises in the exercise sessions; however, feel free to reach out with questions regarding any of the exercises. You will also have access to sample solutions from all these exercises.

1 Finite Automata

In this exercise you are asked to design your first finite automata. Try to minimize the number of states of your machines.

a) Consider the alphabet {0,1}. Implement an automaton which accepts the following strings:

At first there are zero or more '1's, followed by zero or more '0's. This in turn is followed by an arbitrary (non-zero) number of '1's.

b) Design an automaton which decides whether a number is divisible by three. Assume that the digits of the number are inserted sequentially, that is, the number '135' is inserted as '1', '3' and finally '5'. How many states do you need? [1]

2 Vending Machine Revisited

Consider the vending machine shown at the beginning of the course. Do you see any problems with this machine? How can the machine be made more user-friendly?

3 "Mais im Bundeshuus"

It is Wednesday morning and the seven members of the Swiss Federal Council meet to decide about an important topic: In order to decrease the number of leaks of electronic messages, should only pigeon post be used for official communications?

[1] Hint: A number is divisible by 3 iff the sum of its digits is divisible by 3

a) Assume that the seven members vote one after another. Further, assume that there is no abstention of voting. Design an automaton which accepts the ballot if and only if the majority of the members voted in favor of the proposition.

b) Extend your automaton for the case of abstentions of voting. The automaton should accept if and only if more members voted in favor of the proposition than against it.

4 Build Finite Automata with the Construction Rules

Consider the alphabet {a,b}. For each of the following languages, implement an automaton that accepts it using the constructions seen in the lecture (*e.g.,* the cartesian product construction).

a) $\{w$ — w has exactly two `a`'s and at least two `b`'s$\}$

b) $\{w$ — w does not contain string `baba`$\}$

5 Exam question [2015]

Consider the alphabet $\Sigma = \{0, 1\}$ of binary strings representing numbers (with the most significant bit to the left). Draw a DFA which accepts all the strings that are divisible by 5. Strings are read from left to right. For example, the strings `0`, `101`, `1010` and `01010` would be accepted while `01`, `10` would not. Observe that the empty string, ε, should not be accepted either.

Hint: Consider relying on the modulo operator (mod) which returns the remainder of the division of one number by another. For instance, $1 \bmod 5 = 1$ and $2 \bmod 5 = 2$.

Reminder: A bit shift to the left doubles the value of a binary number.

6 Filter for an Input Stream

We would like to construct an automaton that recognizes substrings from an input stream. The input stream consists of symbols $\{a, b\}$ and the substrings that the automaton should detect are of the form bab^*. In other words, the input of the automaton is a series of a's and b's. The automaton should go into an accepting state whenever the most recently received symbols form a string of the form bab^*. For example, in the input stream $b\ \underline{a}\ \underline{b}\ \underline{b}\ \underline{b}\ \underline{a}\ a\ a\ a\ b\ \underline{a}\ \underline{b}\ \underline{a}\ a$, the automaton should be in an accepting state exactly after the reception of an underlined symbol. Construct a deterministic finite automaton that precisely fulfils the above specification.