Finite-State Automata & Lexing

Exercise 1

Consider the alphabet A of letter a, b and c. We consider the following lexer:

< T_1: a(ab)*, T_2 : b*(ac)*, T_3: cba, T_4: cc* >

We assume earlier rules for early tokens have priority over later rules.

Question 1.1

Apply the lexer to the strings below. In each case, show the sequence of tokens output by the lexer:

- c a c c a b a c a c c b a b c

- c c c a a b a b a c c b a b c c b a b a c

Question 1.2

Convert each of the regular expressions of the lexer to a nondeterministic finite-state automaton. Annotate each final state by the token produced by the corresponding rule.

Question 1.3

Build the disjunction of the nondeterministic finite-state automatons from the previous question and convert it to a deterministic finite state automaton. Annotate final states by the token with highest priority amongst the annotated tokens.

Question 1.4

Minimize the automaton you obtained in the previous question.

Question 1.5

Use you automaton to process the sequences from the first question.

Exercise 2

In this exercise, we will have a look at *parallel composition* of deterministic finite state. The parallel composition of n automatons is another finite state automaton such that:

- Its state space is the cartesian product of the state spaces of the n automatons.
- Its starting state is the tuple of the starting states of the n automatons.
- Its transition function applies each of the n transitions pointwise.
- Its set of final states is the set of states where at least one of the points is final in its machine.

Question 2.1

Build a deterministic finite-state automaton for binary numbers multiples of 2. Do the same for binary numbers multiples of 3.

Question 2.2

Build the parallel composition of the two automatons from the previous question.

Question 2.3

Based on your answer from the previous question, show a deterministic finite-state automaton that accepts binary numbers multiples of 2 or 3, but **not both**.

Exercise 3

Let A be the singleton alphabet containing only the symbol 1. Let L be the language of words over A whose size is a prime number.

$$L = \{ w \in A^* \mid |w| \text{ is prime} \}$$

Prove that L is regular by building a regular expression for L, or prove that L is not regular using the pumping lemma. The pumping lemma states that, for any regular language L, there exists a strictly positive constant number p, such that every word w in L whose length is at least p can be written as $w = xyz$ where:

1. $|y| > 0$
2. $|xy| \leq p$
3. For any number i, we have that xy^iz is in L.