Automata Fall 2024, Test 1

Problem 1. Why do we need to study automata? Provide two main reasons.

Problem 2–4. Let us consider the oversimplified version of a computer that has only two states: the starting state f (off) and the final state n (on), and two possible actions: p (pressing the on-off key) and k (pressing any other key). The symbol k brings us back to the same state, while the symbol p brings us from on to off and from off to on.

Problem 2. Trace, step-by-step, how this finite automaton will check that the word *pkpkp* belongs to this language. Use the tracing to find the parts x, y, and z of the word *pkpkp* corresponding to the Pumping Lemma. Check that the "pumped" word $xyyz$ will also be accepted by this automaton.

Problem 3. Write down the tuple $\langle Q, \Sigma, \delta, q_0, F\rangle$ corresponding to this automaton:

- Q is the set of all the states,
- Σ is the alphabet, i.e., the set of all the symbols that this automaton can encounter;
- $\delta : Q \times \Sigma \to Q$ is the function that describes, for each state q and for each symbol s, the state $\delta(q, s)$ to which the automaton that was originally in the state q moves when it sees the symbol s (you do not need to describe all possible transitions this way, just describe two of them);
- q_0 is the staring state, and
- F is the set of all final states.

Problem 4. Use a general algorithm that we had in class to generate a context-free grammar corresponding to this automaton. Show how this grammar will generate the word *pkpkp*.

Problem 5. Let A_1 be the automaton described in Problem 2. Let A_2 be an automaton that accepts only sequences that end with k. This automaton has two states: the starting state s and the final state f. The transitions are as follows: from any state, p leads to s and k leads to f. Use the algorithm that we had in class to describe the following two new automata:

- the automaton that recognizes the union $A_1 \cup A_2$ of the two corresponding languages, and

- the automaton that recognizes the intersection of the languages A_1 and A_2.

Problem 6. Use the general algorithm that we learned in class to design a non-deterministic finite automaton that recognizes the language $(pk^*)^*p$ (that corresponds to all the sequences that start and end with p):

- first, describe automata for recognizing p and k;
- then, use the automaton for k to design an automaton for recognizing k^*;
- then, combine the automata for p and k^* into an automaton for recognizing concatenation pk^*;
- then, use the automaton for pk^* to design an automaton for recognizing the Kleene star $(pk^*)^*$;
- finally, combine the automata for $(pk^*)^*$ and p into an automaton for recognizing concatenation $(pk^*)^*p$.

Problem 7. Use the general algorithm to transform the resulting non-deterministic finite automaton into a deterministic one.

Problem 8–9. Use a general algorithm to transform the finite automaton from Problem 2 into the corresponding regular expression. Start with eliminating the state n.

Problem 10. Prove that the language of all the words that have at least twice more k's than p's is not regular.