TDDD65
Introduction to the Theory of Computation
2017-10-17

Materials allowed: A dictionary from English to any other language is allowed. No other books, notes etc. are allowed and no electronic equipment (calculators, computer, mobile phones etc.) are allowed.

Questions: Christer Bäckström is available on phone 0705-840889 during the exam.

Grading: The maximum number of points is 30 and 15 points are required to pass the examination. At least 15 p is required for grade 3, at least 20 p is required for grade 4 and at least 25 p is required for grade 5.

Results: When the exams are graded there will be an opportunity to see the exams and discuss the result with the examiner (this is called a *tentavisning* in swedish). When and where this will happen will be announced on the course homepage as soon as the grading is finished.

Please observe the following:

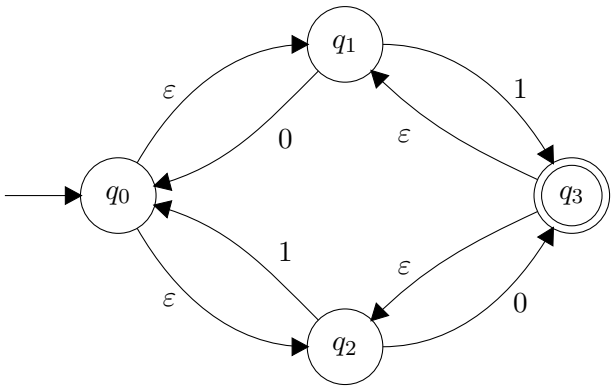
- Use only one side of each paper.
- Each problem must be solved on a separate paper (or several papers, if necessary. Subproblems of a problem (a, b, c etc.) may be solved on the same page.
- Properly justify all your answers. If you give only an answer without justification, you may get zero points even if the answer is correct.
- Make sure your answers are readable.
- Try to leave space for comments on every page.

Good luck!

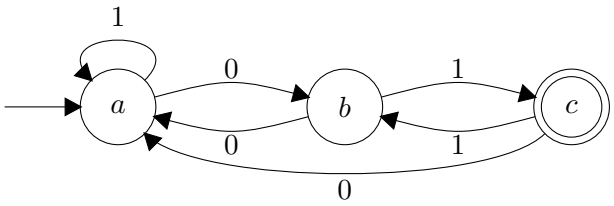
Problems

1. Assume the alphabet $\Sigma = \{0, 1, 2\}$. A string $x_1 x_2 \ldots x_n$ over Σ^* is in numerical order if $x_i \leq x_{i+1}$ for all i where $1 \leq i < n$. For example, the strings 0001222 and 11122 are in numerical order, but the strings 001102 and 1012 are not in numerical order. Draw the state transition diagram for a DFA that accepts exactly those non-empty strings over Σ^* that are *not* in numerical order. (4 p)

2. Construct a DFA that is equivalent to the following NFA using the subset construction method. You must give both the transition table and the state diagram for the resulting DFA. (4 p)



3. Convert the following DFA to a regular expression using the GNFA method. (4 p)



4. Consider the language $L = \{www \mid w \in \{0,1\}^*\}$, i.e. a string in L must consist of three equivalent strings of 0's and 1's. (6 p)

   (a) Show that L is not regular by using the pumping lemma for regular languages.

   (b) Show that L is not context-free by using the pumping lemma for context-free languages.

5. Recall the formal definition of a Turing machine in Sipser. A TM M is a 7-tuple $(Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$, where Q, Σ and Γ are finite sets and (4 p)

   - Q is the set of states,
   - Σ is the input alphabet, not containing the blank symbol $\sqcup$,
   - Γ is the tape alphabet, where $\sqcup \in \Gamma$ and $\Sigma \subseteq \Gamma$,
   - $\delta : Q \times \Gamma \leftarrow Q \times \Gamma \times \{L, R\}$,
   - $q_0 \in Q$ is the start state,
   - $q_{accept} \in Q$ is the accept state and
   - $q_{reject} \in Q$ is the reject state.

   Let $\Sigma = \{0,1\}$ and $\Gamma = \Sigma \cup \{\sqcup\}$ and define the TMs

   $$M_1 = (\{q_0, q_1\}, \Sigma, \Gamma, \delta_1, q_0, q_1, q_0),$$

   and

   $$M_2 = (\{q_0, q_1\}, \Sigma, \Gamma, \delta_1, q_0, q_0, q_1),$$

   where δ_1 is defined as

   $$\begin{aligned} \delta_1(q_0, 0) &= (q_0, \sqcup, R) & \qquad \delta_1(q_1, 0) &= (q_0, 0, L) \\ \delta_1(q_0, 1) &= (q_1, 1, R) & \qquad \delta_1(q_1, 1) &= (q_1, \sqcup, L) \\ \delta_1(q_o, \sqcup) &= (q_1, 0, L) & \qquad \delta_1(q_1, 1) &= (q_0, \sqcup, R) \end{aligned}$$

   (a) Does M_1 halt on all inputs and what is the language $L(M_1)$ that it recognizes?

   (b) Does M_2 halt on all inputs and what is the language $L(M_2)$ that it recognizes?

6. The concept of NP-completeness can be generalised to any complexity class X such that a language L is X-complete if $L \in X$ and $L' \leq_m^p L$ for all languages $L' \in X$ (i.e. NP-completeness is the special case where $X = NP$). (8 p)

   Recall that the complexity class coNP is defined to contain all languages L such that the complement language $\overline{L}$ is in NP. For instance, coNP contains the UNSAT problem, that asks if a boolean formula is not satisfiable, since NP contains the SAT problem. Also this concept can be generalized to any complexity class X such that co$X$ contains all languages L such that $\overline{L}$ is in X.

   (a) We can obviously define the concept of P-completeness for the class P. Is this a useful concept or not?

   (b) We can obviosly define the class coP. Is this a useful class or not?

   (c) It is not currently known if NP=coNP or not. Suppose we could prove that P=NP. Would this have any implications for whether NP=coNP?

   (d) A language is called NP-intermediate if it is in NP, but it is not in P and not NP-complete. The class NPI consists of the NP-intermediate problems. What would happen to this class if we could prove that P=NP?

   It is essential that you explain your answers. Just an answer gives zero points!