4 5

TDDD65/TDDC95
Introduction to the Theory of Computation
2015-01-09, kl. 14-18, Room TER4

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 will show up after approx one hour and is otherwise available on phone 0705-840889

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. Consider a language L where the strings in L is on either of the following two forms: (6 p)

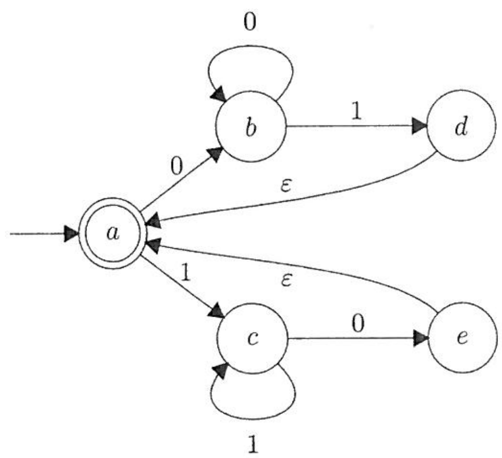
   0^n1, where $n \geq 1$,

   0^n2, where $n \geq 2$.

   (a) Draw the state diagram for a DFA that recognizes L.

   (b) Define L by a regular expression.

2. Convert the following NFA to an equivalent DFA, using the standard method. (6 p)



3. Consider a language L defined by the following CFG. (6 p)

$$S \leftarrow A$$
$$A \leftarrow 0A00 \mid B \mid \varepsilon$$
$$B \leftarrow 1B11 \mid A \mid \varepsilon$$

(a) Prove or disprove that this grammar is ambiguous.

(b) Define L with a regular expression or prove that L is not regular by using the pumping lemma.

4. Consider the usual polynomial-time mapping reduction $\leq_p$. Prove or disprove each of the following claims: (6 p)

(a) $\leq_p$ is reflexive, i.e. for all languages X, $X \leq_p X$.

(b) $\leq_p$ is symmetric, i.e. for all languages X and Y, if $X \leq_p Y$ then $Y \leq_p X$.

(c) $\leq_p$ is transitive, i.e. for all languages X, Y and Z, if $X \leq_p Y$ and $Y \leq_p Z$, then $X \leq_p Z$.

5. Normally, we only distinguish between those Turing machines that are allowed to make non-deterministic moves and those that are only allowed to make deterministic moves. A finer distinction can be made using the concept of *limited non-determinism*, where the number of non-deterministic moves is (6 p)

limited. Then it is possible to define complexity classes of the type $f(n)$-P, for arbitrary function f. Given a function f, the class $f(n)$-P consists of all problems that can be decided in polynomial time on a Turing machine that makes at most $O(f(n))$ non-deterministic moves, where n is the input size.

(a) Prove that $\log n$-P = P.

(b) Prove that $\cup_{i>0} n^i$-P = NP.

Hint, remember that non-determinism can be simulated by search on a deterministic Turing machine. You may also assume that each non-deterministic move consists of only two choices (since this is no restriction).