Comments and hints for 2009 example exams

Harald Hammarström

May 26, 2009

This is an updated version (Monday 25 May) since there were a few errors in the previous version. Chapter references are to the course book. Apologies for typos and errors. (When you write your exam you should be more explicit in your solutions than these notes are.)

20070827

1. - What is the definition of a Determinstic Finite Automaton?

     The answer is simply the five-tuple with states, input symbols, transition function, start state and final states with the correct requirements, i.e., the set of states has to be finite, the transition function has to be a total function taking a state and an alphabet symbol and outputs a state etc. (see 2.2.1 p. 46 in the book for the rest.)
   - Explain what is the language determined by such a finite automaton (1p)

     It's the set of strings $w \in \Sigma^*$ such that $\delta(q_0, w)$ is a final state. Details see 2.2.5.
   - Explain why such a language is a context-free language (1p).

     Every regular language is (also) context-free as it is accepted by, for example, a right-linear CFG. (A formal proof of this is exercise 5.1.4.)
2. Minimize the following automaton (2p)

   Apply the state distinguistishability algorithm described in 4.4.

0, 1, 2, 4 are distinguishable from 3, 5 (via ϵ). 0 and 2 as well as 1 and 2 are distinguishable via b. 4 is distinct from 0, 1, 2 via a. So we merge [0,1] and [3,5].

3. Build a NFA with 3 states that accepts the language $\{ab, abc\}^*$ (2p)

   Note that an NFA (not a DFA) is being asked for. Let X1, X2, X3 be the three states with X1 initial and X1, X3 final. Let there be an a-transition from X1 to X2, b-transition from X2 to X3, and c-transition from X3 to X1, and an ϵ-transition from X3 to X1.

4. Build a DFA corresponding to the regular expression $(ab)^* + a^*$. (3p)

   Perhaps the easiest is to first build a NFA and then convert it to a DFA.

   Another way is to look at the derivatives (see the slides around the Myhill-Nerode theorem for algorithm and proof of correctness, not so much in the book). We then get:
   $\epsilon/L = (ab)^* + a^*$
   $a/L = b(ab)^* + a^*$
   $b/L = \emptyset$
   $aa/L = a^*$
   $ab/L = (ab)^*$
   $aba/L = b(ab)^*$

   Though when you do your solutions in the exam you should also argue that these are the only derivatives. The derivatives give the following DFA (where F is the name for the $\emptyset$ state, and the others named in the order of above):

	a	b
A	B	F
B	C	D
C	C	F
D	E	F
E	F	D
F	F	F

   A initial and A, B, C, D final.

5. Give a regular expression E such that $L(E) = \Sigma^* - L(10(0+1)^*)$ (2p)

The language asked for is the language of strings that don't begin with 10 so $(11 + 01 + 00)(0 + 1)^* + 0 + 1 + \epsilon$ should do the trick.

6. Build a DFA that recognizes exactly the word in $\{0, 1\}^*$ ending with the string 1110. (2p)

	0	1
A	A	B
B	A	C
C	A	D
D	E	C
E	A	B

With A initial and E final.

7. Is the following grammar ambiguous? Why? (2p)

aab has two parse trees. One in which it has height 2 and comes from $S \rightarrow aaB$ and $B \rightarrow b$. Another in which it has height 3 and comes from $S \rightarrow AB$, $A \rightarrow aA$, $A \rightarrow a$ and $B \rightarrow b$.

We have to be a bit clever and see that the language generated is a^*b so an equivalent non-ambiguous grammar simply drops the $S \rightarrow aaB$ rule.

8. Show that the string *aabbabba* is not in the language generated by this grammar (3p).

The best way to show this is to transform the grammar into a CNF and then parse the string and see that S is not in the top-left corner. Another way is as follows. If $S =>^* aabbabba$ then clearly $B =>^* bbabba$ and, in turn $A =>^* bbabb$ and in turn $B =>^* bab$. But there is no way that $B =>^* bab$ because $B \rightarrow Aa$, being the only rule for B, must produce a stringe ending with a.

9. Find context-free grammars for the languages

For b, see exercise 5.1.1b which shows the trick and has an online solution. For a, contact me if you find a CFG for the language.

10. Let L,M,N be languages on an alphabet Σ^*. Explain why we have $L(M \cap N) \subseteq LM \cap LN$ (2p).

Let w be an arbitrary string in the LHS set. Then we can break $w = xy$ into parts $x \in L$ and $y \in M \cap N$. Since $y \in M \cap N$ we have $y \in M$ and $y \in N$. Since $y \in M$, $y \cap N$ and $x \in L$ we have $xy \in LM$ and $xy \cap LN$. Since $xy \in LM$ and $xy \cap LN$ we have $w \in LM \cap LN$.

Give an example showing that we do not have $LM \cap LN \subseteq L(M \cap N)$ in general (2p).

With $L = \{\epsilon, a\}$, $M = \{a\}$, $N = \{aa\}$ we have $LM \cap LN = \{aa\}$ but RHS is empty.

11. Yes L_2 is regular. If $w \in L_1 \cup L_2$ and $w \in \overline{L_1}$ then because $L_1 \cap L_2 = \emptyset$, w must be in L_2. So an automaton for $(L_1 \cup L_2) \cap \overline{L_1}$ is an automaton for L_2. There is such an automaton since regular languages are closed under union, intersection and complementization.

20070531

1. What is, mathematically, a context-free Language (1p)? Give, with motivation, an example of a language which is context-free, but not regular (1p) and an example of a language which is not context-free (1p)

   A context-free language is a set $\{w \in T^* | S =>^* w\}$ where S is the start category of a context-free grammar. See 5.1.2 and 5.1.5 for details. Examples of the kind requested can be found in chapter 5 (and many other places in the literature for the course).

2. Prove this, e.g., by induction on x.

3. Very similar to 5 on the previous exam.

4. Very similar to 2 on the previous exam.

5. Very similar to 6 on the previous exam.

6. Very similar to 4 on the previous exam.

7. Hint: $A - B$ (set difference) is the same as $A \cap \overline{B}$

8. Yes. Because abab has two parse trees. One via $S \to SS, S \to ab, S \to ab$ and one via $S \to aSb, S \to ba$.

9. a. Let $L_1 = \Sigma^*$ and $L_2 = \{a^n b^n | n \geq 1\}$.

   b. Let $L_1 = \{ab\}$ and $L_2 = \{a^n b^n | n \geq 1\}$.

10. First one:

    We first write a (non-CNF) CFG grammar for the language:
    $S \rightarrow AC$
    $A \rightarrow abb | aAbb$
    $C \rightarrow c | cC$

    which is easily transformed into CNF by introducing new symbols
    $S \rightarrow AC$
    $B \rightarrow b$
    $D \rightarrow BB$
    $F \rightarrow a$
    $G \rightarrow FA$
    $A \rightarrow FD | GD$
    $C \rightarrow c | CC$

    Second one, we first write a (non-CNF) grammar:
    $S \rightarrow aBa | aSa$
    $B \rightarrow b | bB$

    which is easily transformed into CNF (omitted).

11. (a) Any subset of a regular language is regular (1p)

    No, $\{a^n b^n | n \geq 1\}$ is a subset of $\{a, b\}^*$.

    (b) If L_n is a family of regular language then $\cup_n L_n$ is regular (1p)

    No, let $L_i = \{a^i b^i\}$, then the infinite union will yield a non-regular language.

12. Use pumping lemma, on e.g., the string $b^N c^N$ where N is the constant of the pumping lemma.

13. Explain why if L is regular then so is L/a for any $a \in \Sigma$ (2p).

    If L is regular then L has an automaton. Remove all the transitions with other symbols than a from the start state. Replace all the a-transitions from the start state with epsilon transitions instead. We now have a NFA for L/a so L/a is regular. An automaton for (L/a)a is similarly constructed from that of L/a by adding a new final state (and removing the final-ness from the old final states). Then put a-transitions from all old final states to the new final one.