

## Problem Set 1

**Due: beginning of class on Tue. Jan. 23, 2001**

### Problem 1

Consider the DFAs  $M_1$  and  $M_2$  shown in figure 1. Build a DFA  $M$  accepting the intersection of the languages  $L(M_1)$  and  $L(M_2)$  using the cartesian product construction described in the second lecture. Give both a state diagram and the formal description of the automaton  $M$ .



Figure 1: Automata  $M_1$  and  $M_2$

### Problem 2

In many programming languages (C, C++, Java, perl, etc.) strings are typically specified as sequences of characters enclosed in double quotes (e.g. "ntnt" stands for the string *ntnt*). In order to specify strings containing the double quote character, the symbol " can be quoted inserting a backslash character right before it. For example, "ttn\"ntt" denotes the 6-character string *ttn"ntt*. Other "escape sequences" with special meaning are `\n` (new line), `\t` (tab), etc. Backslash characters can also be inserted using the escape sequence `\\`. Give a DFA accepting the language of all words over the alphabet  $\Sigma = \{n, t, \backslash, \text{"}\}$  corresponding to valid C-style string constants, as informally described above. (Examples of words that should be accepted by the automaton are "ttn\t\" or "t\t" but not *tnnt* or "\".) Give both the state diagram of the DFA and the table corresponding to its transition function.

### Problem 3

Given a word  $w = w_1w_2 \dots w_n$  (where  $w_i \in \Sigma$  for all  $i$ ), let the even and odd parts of  $w$  be the words  $even(w) = w_2w_4w_6 \dots$  and  $odd(w) = w_1w_3w_5 \dots$  obtained taking

only the characters at even (resp. odd) positions. Notice: the even part of a single character word, and the even or odd part of the empty word  $\epsilon$  are all equal to the empty word  $\epsilon$ . [Informal explanation: you can think of the word  $w$  as a sequence of single character messages sent by two alternating parties (Jee Hea and Alejandro) during a conversation. Then  $even(w)$  is the sequence of messages transmitted by Alejandro, while  $odd(w)$  is the sequence of messages sent by Jee Hea.] These operations can be extended to languages taking the even (resp. odd) part of every word in the language:  $even(L) = \{even(w) \mid w \in L\}$  and  $odd(L) = \{odd(w) \mid w \in L\}$ . For example,  $even(\{askwr, eityqe\}) = \{sw, iye\}$  and  $odd(\{askwr, eityqe\}) = \{akr, etq\}$ .

Prove that if  $L$  is a regular language, then also  $odd(L)$  and  $even(L)$  are regular. (Hint: you can use non-deterministic automata.)

Optional (you don't have to write anything for this part): start thinking about the opposite problem, i.e., assuming that  $even(L)$  and  $odd(L)$  are both regular, can you conclude that  $L$  is also regular?

### Problem 4

Consider again the two deterministic automata from figure 1. Give the state diagram and formal description of an NFA accepting the concatenation of the two languages. Transform the NFA into an equivalent DFA using the transformation described in the book. (Just draw the state diagram of the DFA. You only need to draw the reachable states, but please label the states of the DFA with sets of states of the NFA so that the relation between the two automata is clear.)

### Problem 5

Transform the following regular expression into an equivalent (non-deterministic) finite state automaton using the procedure described in class:

$$(a \cup b)^* ab (bab \cup a^*)^* bb^*$$

### Problem 6

Transform the following automaton into an equivalent regular expression using the procedure studied in class:

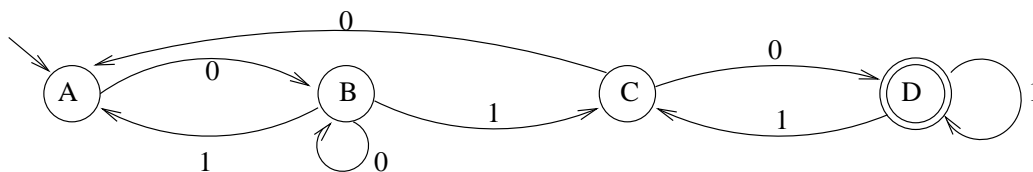


Figure 2: A non-deterministic finite state automaton