

## Homework 2 – Due Thursday, September 10, 2009 before the lecture

Please refer to the general information handout for the full homework policy and options.

**Page limit** You can submit **at most** 1 page per problem, even if the problem has multiple parts. If you submit a longer solution for some problem, only the first page will be graded. This homework contains 3 problems, worth 10 points each. Your solution to each problem should be handed in on a separate sheet of paper.

**Reminder** Collaboration is permitted, but you must write the solutions *by yourself without assistance*, and be ready to explain them orally to the instructor if asked. You must also identify your collaborators. Getting solutions from outside sources such as the Web or students not enrolled in the class is strictly forbidden.

**Exercises** Please practice on exercises and solved problems in Chapters 1. The material they cover may appear on exams.

### Problems

1. (**Conversion procedures**) Use asymptotic (big- $O$ ) notation to answer the following questions.

- Let  $N$  be an NFA that has  $n$  states. If we convert  $N$  to an equivalent DFA  $M$  using the procedure we described, how many states would  $M$  have?
- Let  $R$  be a regular expression that has  $n$  symbols (each constant/operation counts as one symbol). If we convert  $R$  to an equivalent NFA  $N$  using the procedure described in class, how many states would  $N$  have in the worst case?
- Let  $M$  be a DFA that has  $n$  states. If we convert  $M$  to an equivalent regular expression  $R$  using the procedure we described, how many symbols would  $R$  have in the worst case?

In an *extended* regular expression, we may use the complement operation ( $\neg$ ) in addition to the three regular operations ( $\cup, \circ, \star$ ). For example,

$$\neg(\Sigma^*001\Sigma^*) \cup \neg(\Sigma^*100\Sigma^*)$$

is an extended regular expression that describes the collection of all strings that either do not contain the substring 001 or do not contain the substring 100.

- Describe how to modify the conversion procedure from regular expressions to NFAs so that it becomes a conversion procedure from extended regular expressions to NFAs.
  - Let  $R$  be a *extended* regular expression that has  $n$  symbols (each constant/operation counts as one symbol). If we convert  $R$  to an equivalent NFA  $N$  using the procedure you described above, how many states would  $N$  have in the worst case?
2. (**Number of states**) For each  $k \geq 1$  let  $C_k$  be the language over  $\Sigma = \{a, b\}$  consisting of all strings with two  $a$ 's that are  $k - 1$  symbols apart. Using regular expressions,  $C_k = \Sigma^*a\Sigma^{k-1}a\Sigma^*$ .

- (a) Give a state diagram and a formal description of an NFA with  $k + 2$  states that recognizes  $C_k$ .
- (b) Prove that for each  $k$ , no DFA with fewer than  $2^k$  states can recognize  $C_k$ .

*Hint:* If a DFA enters different states after reading two different input strings  $xz$  and  $yz$  with the same suffix  $z$  then the DFA must enter different states after reading input strings  $x$  and  $y$ . (Explain why.) Find  $2^k$  strings on which every DFA recognizing  $C_k$  must enter different states. (Start by finding two such strings.)

3. (**Non-regular languages**) Prove that the following languages are not regular. You may use the pumping lemma and the closure of the class of regular languages under union, intersection and complement.

- (a)  $L_1 = \{0^n 1^m 0^n \mid m, n \geq 0\}$ .
- (b)  $L_2 = \{0^k \mid k \text{ is a prime number}\}$ .
- (c)  $L_3 = \{1^k y \mid y \in \{0, 1\}^* \text{ and } |y| = k\}$ .