CS 8803-O08: Homework #1

Santosh Pande
santosh.pande@cc.gatech.edu

Georgia Institute of Technology — Due: Sep 12, 2022 11:59 PM AOE

1 Overview

Homework questions cover the topics discussed in the lecture videos and assigned textbook readings. If you have questions please ask them on Ed Discussion using the appropriate tags and visibility. Students are expected to solve the problems themselves and abide by the Georgia Tech Honor Code at all times. Answers should be complete, concise, and understandable.

2 Regular Expressions

Warning: When writing regular expressions, ONLY the following syntax is allowed:

1. concatenation (abc)
2. alternation (a|b|c)
3. grouping (a(b|c)d)
4. . (represents any character in the alphabet except line break)
5. + (one or more)
6. * (zero or more)
7. ? (once or none)
8. [](character groups like [a-z] or [0-9])

Question 1 *(18 points)*

Write regular expressions to match the following:

(a) strings of length three or longer that start with the letter *b* and end with either the letter *d* or *m*. Alphabet is [a-z]

(b) unsigned integers in which digits occur in a non-increasing order. Empty string is allowed. Leading zeros are not allowed,e.g., 00, 01, 000, 001, etc. Alphabet is [0-9]

(c) non-empty strings which do not contain 00 as a substring. Alphabet is [0-1]

Question 2 *(24 points)*

For each of the following languages, determine if they are regular. If they are regular, implement them using regular expressions. If they are not regular, informally say why not (what does the machine need to do to recognize the expression and why is it incapable).

(a) Given a fixed positive integer k, the language $L(k) = a^{*k}$ where ^{*k} is defined as a modification of $*$ meaning zero to k rather than zero or more. Alphabet is [a]

(b) $L = a\,\Omega\,b$ where Ω represents a substring made up of four or fewer a's followed by two or fewer b's. Ω may contain zero a's and/or zero b's. Alphabet is [a-b]

(c) non-empty strings that contain a positive multiple of six b's and zero or more a's. Alphabet is [a-b]

(d) non-empty strings whose length is a power of two. Alphabet is [a]

3 DFAs and NFAs

Formally, a finite automaton is described as a five-tuple $(S, \Sigma, \delta, s_0, S_A)$, where

- S is the finite set of states.
- Σ is the finite alphabet.
- $\delta(s, c)$ is the transition function which maps each state $s \in S$ and each character $c \in \Sigma$ to next state.
- $s_0 \in S$ is the start state.
- S_A is the set of accepting states, $S_A \subseteq S$.

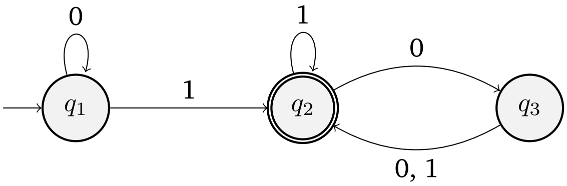

Figure 1: A simple DFA.

The DFA in Figure 1 has the five-tuple representation

- $S = \{q_1, q_2, q_3\}$
- $\Sigma = \{0, 1\}$
- δ is the set of transitions
  - $(q_1, 0) \rightarrow q_1$
  - $(q_1, 1) \rightarrow q_2$
  - $(q_2, 0) \rightarrow q_3$
  - $(q_2, 1) \rightarrow q_2$
  - $(q_3, 0) \rightarrow q_2$
  - $(q_3, 1) \rightarrow q_2$
- $s_0 = q_1$
- $S_A = \{q_2\}$

Question 3 *(24 points)*

Provide the DFAs for the following languages using **BOTH** the five-tuple format described above and a graphical representation similar to figure 1. Both answer formats must be included for full credit.

(a) strings that **EITHER** start with 1 and have an even length **EXCLUSIVE OR** start with 0 and have a length which is a multiple of 3. Alphabet is [0-1]

(b) strings that **EITHER** contain an odd number of 1's **INCLUSIVE OR** contain exactly two 0's and end on a 1. Alphabet is [0-1]

(c) strings that do **not** match the regular expression $(ab)^*$. Alphabet is [a-b]

Question 4 *(34 points)*

1. Using primitive NFAs (start with NFA for each character in the regular expression), apply Thompson's construction to assemble a full NFA for the regular expression: $(ab)^+(a|b)^*$
2. Next, use Brzozowski's algorithm to convert this NFA to a minimized DFA. Specify all NFA/DFA using a graphical representation similar to figure 1. Show each of the following steps:

   (a) NFA for $(ab)^+(a|b)^*$

   (b) Reverse NFA from (a) (Show reversed NFA)

   (c) Subset the NFA in (b) (Show complete subset construction table and resulting DFA)

   (d) Reverse the DFA in (c) (Show reversed DFA)

   (e) Subset the NFA in (d) (Show complete subset construction table and resulting minimized DFA)