MISC

Homework 2
WHAT REGULAR EXPRESSION DESCRIBES

\[ L = \{ w \mid w \text{ has a 0 in its 4th position from the end} \} \]
\( \Sigma \ast 0 \Sigma^3 \)
What DFA describes this regular expression?

\[ \Sigma^* 0 \Sigma^3 \]
\[ \sum^* \theta \sum^3 = \]
Deterministic Finite Automata
import java.util.*;
import java.io.*;

public class NFA1 extends Thread {
    public static boolean accept = false;
    private String s;
    private int pos;

    NFA1(String s, int pos) {
        this.s = s; this.pos = pos;
    }

    public void run() {
        if (s.length() - pos == 3) {
            accept = true;
        }
    }
}
import java.util.*;
import java.io.*;

public class NFA1 extends Thread {
    public static boolean accept = false;

    private String s;
    private int pos;

    NFA1(String s, int pos) {
        this.s = s; this.pos = pos;
    }

    public void run() {
        if (s.length() - pos == 3) {
            accept = true;
        }
    }

    public static void main(String[] args) throws Exception {
        BufferedReader inp = new BufferedReader(new InputStreamReader(System.in));
        String line = inp.readLine();
        Vector<Thread> threads = new Vector<Thread>();

        for (int i = 0; i < line.length(); i++) {
            if (line.charAt(i) == '0') {
                NFA1 n = new NFA1(line, i+1);
                n.start();
                threads.add(n);
            }
        }

        for (Thread t : threads) {
            t.join();
        }

        System.out.println(accept ? "Accept" : "Reject");
    }
}
DFAs $Q$ set of states
DFAs

\( Q \) set of states

\( \Sigma \) the alphabet
DFAs

\( Q \)  set of states

\( \Sigma \)  the alphabet

\( \delta : Q \times \Sigma \rightarrow Q \)  transition function
DFAs

\[ Q \]

set of states

\[ \Sigma \]

the alphabet

\[ \delta : Q \times \Sigma \rightarrow Q \]

transition function

\[ q_0 \in Q \]

start state
DFAs

- $Q$: set of states
- $\Sigma$: the alphabet
- $\delta: Q \times \Sigma \rightarrow Q$: transition function
- $q_0 \in Q$: start state
- $F \subseteq Q$: set of final states
NFAs $Q$ set of states
NFAs

$Q$ set of states

$\Sigma$ the alphabet
NFAs

\[ Q \]

set of states

\[ \Sigma \]

the alphabet

\[ \delta : Q \times \Sigma_e \rightarrow P(Q) \]

transition function
NFAs

\[ Q \quad \text{set of states} \]

\[ \sum \quad \text{the alphabet} \]

\[ \delta: Q \times \sum \rightarrow P(Q) \quad \text{transition function} \]

\[ q_0 \in Q \quad \text{start state} \]
NFAs

- $Q$: set of states
- $\Sigma$: the alphabet
- $\delta: Q \times \Sigma \rightarrow P(Q)$: transition function
- $q_0 \in Q$: start state
- $F \subseteq Q$: set of final states
\[ \delta : Q \times \Sigma_e \rightarrow P(Q) \] transition function

\[ \Sigma_e = \Sigma \cup \{ \varepsilon \} \]

\[ P(Q) \] power set
Every DFA is an NFA.
Claim: Any NFA has an equivalent DFA.

(we’ll prove this soon)
Transition function?
NFA Examples
\[ \Sigma = \{0, 1\} \]

\[ 0^*1^*0^+ \]
Example 1

\[ \Sigma = \{0, 1\} \]

\[ 0^* 1^* 0^+ \]

Diagram:

- States: A, B, C
- Edges: 0 → A, 1 → B, ε → B, 0 → B, 1 → B, 0 → C

Example 1
$$\Sigma = \{0, 1\}$$

$$L = \{w \mid w \text{ contains 000 or 01100}\}$$
$\Sigma = \{0, 1\}$

$L = \{w \mid w \text{ contains 000 or 01100}\}$
Claim:
The class of regular languages is closed under union.
(Already shown with DFAs)
IDEA

Connect to both NFAs’ start states from a new start state.
Claim:
The class of regular languages is closed under concatenation.
IDEA

Connect first NFA’s accept states to second NFA’s start state.
Claim:
The class of regular languages is closed under Kleene star.
Loop NFA’s accept states to its start state.
Claim: Any NFA has an equivalent DFA.
IDEA

Build a **DFA** to track all possible states the **NFA** could be in.
Example 1

\[ \Sigma = \{0, 1\} \]

0*1*0^+
Example 1

\[ \Sigma = \{0, 1\} \]

\[ 0^*1^*0^+ \]

Example 1
DFAs

\[ Q \]

set of states

\[ \Sigma \]

the alphabet

\[ \delta : Q \times \Sigma \rightarrow Q \]

transition function

\[ q_0 \in Q \]

start state

\[ F \subseteq Q \]

set of final states
Example 1 as a DFA

\[ \Sigma = \{0, 1\} \]
Example 1 as a DFA

\[ \Sigma = \{0, 1\} \]

\[ Q = \{\emptyset, \{A\}, \{B\}, \{C\}, \{A, B\}, \{A, C\}, \{B, C\}, \{A, B, C\}\} \]
\[ \Sigma = \{0, 1\} \]

\[ Q = \{\emptyset, \{A\}, \{B\}, \{C\}, \{A, B\}, \{A, C\}, \{B, C\}, \{A, B, C\}\} \]

\[ q_0 = \{A, B\} \]
\[ \Sigma = \{0, 1\} \]

\[ Q = \{\emptyset, \{A\}, \{B\}, \{C\}, \{A, B\}, \{A, C\}, \{B, C\}, \{A, B, C\}\} \]

\[ q_0 = \{A, B\} \]

\[ F = \{\{C\}, \{A, C\}, \{B, C\}, \{A, B, C\}\} \]
Example 1 as a DFA

\[
\begin{align*}
\{A, B\} &\rightarrow 0, 1 \\
\emptyset &\rightarrow 0, 1 \\
\{B\} &\rightarrow 0 \\
\{C\} &\rightarrow 0 \\
\{A, B, C\} &\rightarrow 0
\end{align*}
\]
Claim:
A language is regular if and only if a regular expression describes it.
Reading: Sipser 1.2, 1.3