RE & DFA

Young-Rae Cho, Ph.D.
Professor
Department of Software / Department of Digital Healthcare
Yonsei University – Mirae Campus

Regular Expression (RE)

- **Definition**
  - A sequence of characters matching (or describing) a pattern in a string

- **Syntax**
  - RE is composed of characters and metacharacters
  - Metacharacters: Characters having special meanings
  - Examples of metacharacters
    - Grouping: **( )**
    - Quantifier: ***** (occurring zero or more times)
    - Alternative: **|**

Regular Expression (RE)

- **Practice**
  - Binary strings of length-2
  - Binary strings of length-3
  - Binary strings starting with ‘0’
  - Strings of ‘a’ and ‘b’ ending with ‘a’
  - Strings of ‘a’ and ‘b’ that include “aa” at least once
  - Strings of ‘a’ and ‘b’ with an odd number of ‘a’
  - Strings of ‘a’ and ‘b’ with an even number of ‘a’
  - Strings of ‘a’ and ‘b’ that do not have “aa”

Regular Expression (RE)

- **More Practice**
  - DNA sequences starting with “TA” and ending with “AA”
  - DNA sequences that include only one ‘T’
  - DNA sequences that include at least one ‘G’
  - DNA sequences that do not have “AA”

Regular Expression in Python

- **Metacharacters**
  - . (any character), \n, \t, \s (whitespace),
  - \w (any alphabetic or numeric character), \W (not alphabetic nor numeric character)
  - \d (decimal digit), \D (no decimal digit)
  - Grouping and back-reference:

    e.g., '(.)(.)aa\1\2'
  - Quantifier: *, +, ?, { }

    e.g., 'ct .*g', 'ct .+g', 'ct .?g', 'ct{2}g', 'ct{2,5}g'
  - Alternative: |

    e.g., '(ct|ca)'
  - Character set: [ ]

    e.g., '[acgt]', '[a-zA-Z]'
  - Anchors: ^ (the start of the string), $ (the end of the string)

    e.g., '^tata' , 'aa$'

Deterministic Finite Automaton (DFA)

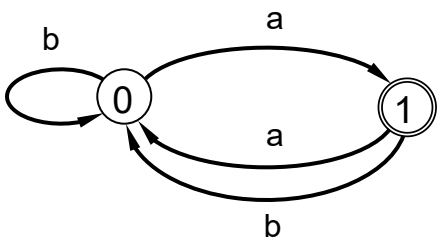
- **Definition**
  - A finite-state machine matching (or describing) a pattern in an input string
    → DFA accepts the input string if it contains the pattern
    → DFA rejects the input string if it does not contain the pattern

- **Syntax**
  - DFA is composed of (Q, q_0, A, Σ, δ)
    - Q : a finite set of states
    - q_0 : a start state
    - A : a set of accepting states
    - Σ : a finite set of input characters in a domain
    - δ : transition functions from (Q x Σ) to Q

Deterministic Finite Automaton (DFA)

- **Application**
  - Read each character of the input string and move on DFA, repeatedly
    → DFA accepts the input string if it ends up in an accepting state
    → DFA rejects the input string if it doesn't end up in an accepting state

- **Example**
  - Verifying inputs
  - Q ?
  - q_0 ?
  - A ?
  - Σ ?
  - δ ?



"aba" ?

"babb" ?

- **Example**
  - Constructing automaton

[a|b]* a b

Deterministic Finite Automaton (DFA)

- **Practice**
  - Binary strings of length-2
  - Binary strings of length-3
  - Binary strings starting with ‘0’
  - Strings of ‘a’ and ‘b’ ending with ‘a’
  - Strings of ‘a’ and ‘b’ that include “aa” at least once
  - Strings of ‘a’ and ‘b’ with an odd number of ‘a’
  - Strings of ‘a’ and ‘b’ with an even number of ‘a’
  - Strings of ‘a’ and ‘b’ that do not have “aa”

Deterministic Finite Automaton (DFA)

- **More Practice**
  - DNA sequences starting with “TA” and ending with “AA”
  - DNA sequences that include only one ‘T’
  - DNA sequences that include at least one ‘G’
  - DNA sequences that do not have “AA”

Questions?

- Lecture Slides on the Course Website, “https://ads.yonsei.ac.kr/faculty/biocomputing”