CMSC 330: Organization of Programming Languages

Project 2 – Finite Automata Interpreter

Overview

- Write Ruby program to implement finite automata
  - Compose automata representing NFAs
    - Concatenate
    - Union
    - Closure
  - Convert automata representing NFAs to ones representing DFAs
    - Subset construction
  - Minimize automata representing DFAs
    - Hopcroft reduction
    - Or any other minimization algorithm...

Starting Ruby Code – fa.rb

- Class FiniteAutomaton
  - Can already represent DFAs
    ```ruby
class FiniteAutomaton
  def initialize
    @start = nil  # start state
    @state = { }  # all states
    @final = { }  # final states
    @transition = { }  # transitions
    @alphabet = [ ]  # symbols on transitions
  end
```
  - You need to extend it to also represent NFAs

Starting Ruby Code – fa.rb

- Interpreter and stack
  - Reads commands, operates on stack
    ```ruby
def interpreter
dfaStack = [ ]
loop do
  case word
  when /SIZE/  # SIZE command
    f = dfaStack.last  # look at top automata on stack
    puts f.num_states
  when /DFA/  # DFA command
    f = dfaStack.pop  # take top automata on stack
    f2 = f.toDFA  # make it into DFA
    dfaStack.push f2  # push result back on stack
  end
end
```
  - You need to implement functions called by interpreter

Input Format

- Commands to interpreter
  - Consisting of
    - Symbols in alphabet
      - A, b, c
    - Operators
      - , | *
  - Commands to interpreter
    - SIZE, DFA, PRINT, MINIMIZE, DONE
  - Input strings to be tested
  - Example
    - a a a | DFA PRINT "a" "aa" "aaa" DONE

Output of fa.rb Script

- Run as
  ```ruby
  ruby fa.rb < input_file.in
  ```
- Output
  - Results of commands
    - Values (e.g., # of states in finite automata)
  - Accept / reject for string
  - List of strings accepted for GenStr method
    - Lists all strings accepted under some length
  - All output beginning in % ignored by test script
Example Session

- **Input**
  - a a a | . DFA PRINT "a" "aa" "aaa" DONE

- **Output**
  - % Start 8
  - % Final { 10 }
  - % States { 8 9 10 }
  - % Alphabet { a }
  - % Transitions {
    - % (8 a 9)
    - % (9 a 10)
    - % }
  - Reject
  - Reject a
  - Accept aa
  - Reject aaa

Administration

- **Project description & files**
  - Download from class web page

- **Due midnight Thu, Oct 6th**
  - 10% penalty for 1 day late

- **Submit fa.rb to submit server**
  - submit.cs.umd.edu

- **Public test cases**
  - Sample inputs & outputs available