Project 1: DFA Simulator

CS 440: Programming Languages and Translators, Fall 2020

Due Mon Oct 19, 11:59 pm

How to Work; How to Submit

As with the homework, you'll be working in groups of 3, either self-formed or randomly-assigned. (We'll discuss details on Piazza.) One person from each group should submit only the program, either as a text file or as a zipped collection of text files. Include the names and A-ids of all three members in a comment at the top of your program files. As with the homework, each group member should turn a work report discussing what each of the group members did (including the submitter).

DFA Simulator

The goal is to write an OCaml or python program that reads a description of a DFA and an input string and runs the DFA to see whether or not it accepts the string. At each DFA step, you print out the state you're in and the terminal symbol you saw. At the end you print out the final state and whether or not it accepted.

DFA Description Format

- One line with the number of states N in the DFA (the states will be 0 through N-1).
- One line with the number of accepting states in the DFA
- One line (for each accepting state) with the state number
- One line with the number of symbols M in the alphabet
- One line with the alphabet (M characters)
- One line with the number of transition cases
  - One line for each transition case: old-state comma symbol comma new-state.
- One line with the input to the DFA.

Example of DFA Input

The DFA below has states 0 and 1 with 1 as the only accepting state, alphabet a, b, c, and space, and seven transition function entries

```
2       — Two states, 0, 1
1       — One accepting state
1       — 1 is an accepting state
4       — 4 characters in the alphabet
abc     — the characters are a, b, c, and space (hard to see, but it's the 4th character)
```



```
7          — Number of transition function entries
1,b,1      — In state 1, on b, go to state 1 [note the entries don't have to be sorted]
1,  a , 1  — In state 1, on a, go to state 1 [notice there exists whitespace]
0, b, 0
0,a, 1
1, c,0
0,,0       — In state 0, on space, go to state 0 [two commas imply space as input symbol]
1 , , 1    — In state 1, on space, go to state 1 [two commas imply space as input symbol]
bba aca    — The input to the DFA
```

The alphabet should include only individual characters (no escape sequences like \t). If a space appears within the M characters of the alphabet, then a space is one of the M symbols. Similarly, if a space appears in the DFA input, treat it as a symbol.

Print out a trace of information about the DFA as you read it. If you notice an error, you're allowed to print an error message/raise an exception and halt. You're also allowed to wait until execution of the DFA to realize there are errors in the transition function (bad state number, bad symbol, multiple entries for the same state/symbol pair).

When you execute the DFA on the input, print a trace of the state, symbol, state, symbol, ..., state it encounters. At the end, say whether or not the input is accepted. If you're in a state and the next symbol has no transition function result defined, then stop execution and say what the input symbol is and that you're stuck [execution cannot proceed]. Since not all the input has been processed, it's automatically rejected.

[Hint: If you cause a runtime error or raise an exception, you might want to flush the output buffer so that you don't lose part of the output.]

A sample trace for the DFA and input in the example above could be something like

```
    0 b 0 b 0 a 1   1 a 1 c 0 a 1    — space as 4th character.
or  0 b 0 b 0 a 1 ' '1 a 1 c 0 a 1   — space as 4th character.
```

Your program should be organized so that at the top level of the read-eval-print loop, you can enter

simulate_dfa "name of text file" or simulate_dfa("name of text file")

as appropriate for OCaml or python.

As with the homework, you should work in groups of 3 (or 2), where groups will be randomly generated for synchronous students and self-organized for asynchronous students. In addition to your program, you should submit individual work reports, as with the homework.

