CS 154
Formal Languages and Computability
Midterm #1 Solutions

Department of Computer Science
San Jose State University

Spring 2016
Instructor: Ron Mak
www.cs.sjsu.edu/~mak

Midterm #1 Solutions: Question 1

- $L_G = \{Guten, Tag, Katzen, Gesundheit\}$
  $L_F = \{Bonjour, Au, Revoir, Amour\}$.

a. An example member of $L_G L_F$

- A word from L_G concatenated with a word from L_F
- *KatzenAmour*

b. An example member of $L_F L_G$

- A word from L_F concatenated with a word from L_G
- *BonjourGesundheit*

Midterm #1 Solutions: Question 1, *cont'd*

- $L_G = \{Guten, Tag, Katzen, Gesundheit\}$
  $L_F = \{Bonjour, Au, Revoir, Amour\}$.

c. Three different example members of $L_F^* - L_F$

- λ plus concatenations of any number of words from L_F but not single words.
- λ, *AuRevoir*, *AmourBonjourAmour*

d. Three different example members of $L_F \cup L_G$

- Single words from either L_F or L_G
- *Bonjour, Guten, Katzen*

Midterm #1 Solutions: Question 1

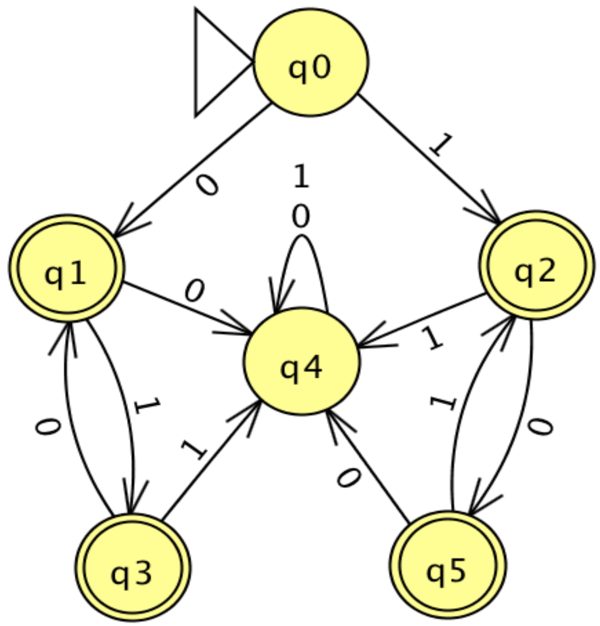
- $L_G = \{Guten, Tag, Katzen, Gesundheit\}$
  $L_F = \{Bonjour, Au, Revoir, Amour\}$.

e. Three different example members of $(L_F \cup L_G)^*$

- λ plus concatenations of any number of words from either L_F or L_G in any order.
- *Bonjour, GutenTag, AuRevoirKatzen*

Midterm #1 Solutions: Question 2

- Consider the DFA

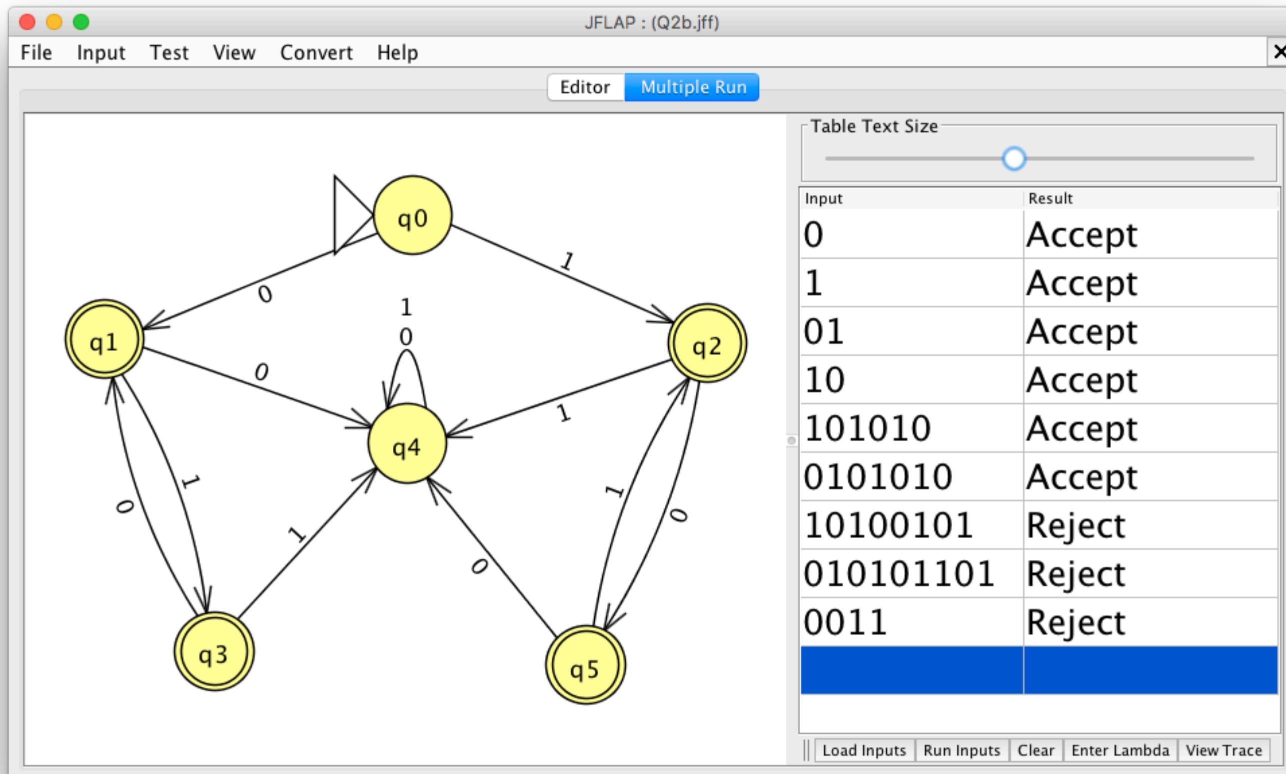


a. What strings will the DFA accept?

   - All strings whose symbols alternate between 0 and 1.

Midterm #1 Solutions: Question 2, *cont'd*

b. Use JFLAP to test your answer with at least four sample strings that are accepted and at least two sample strings that are rejected.



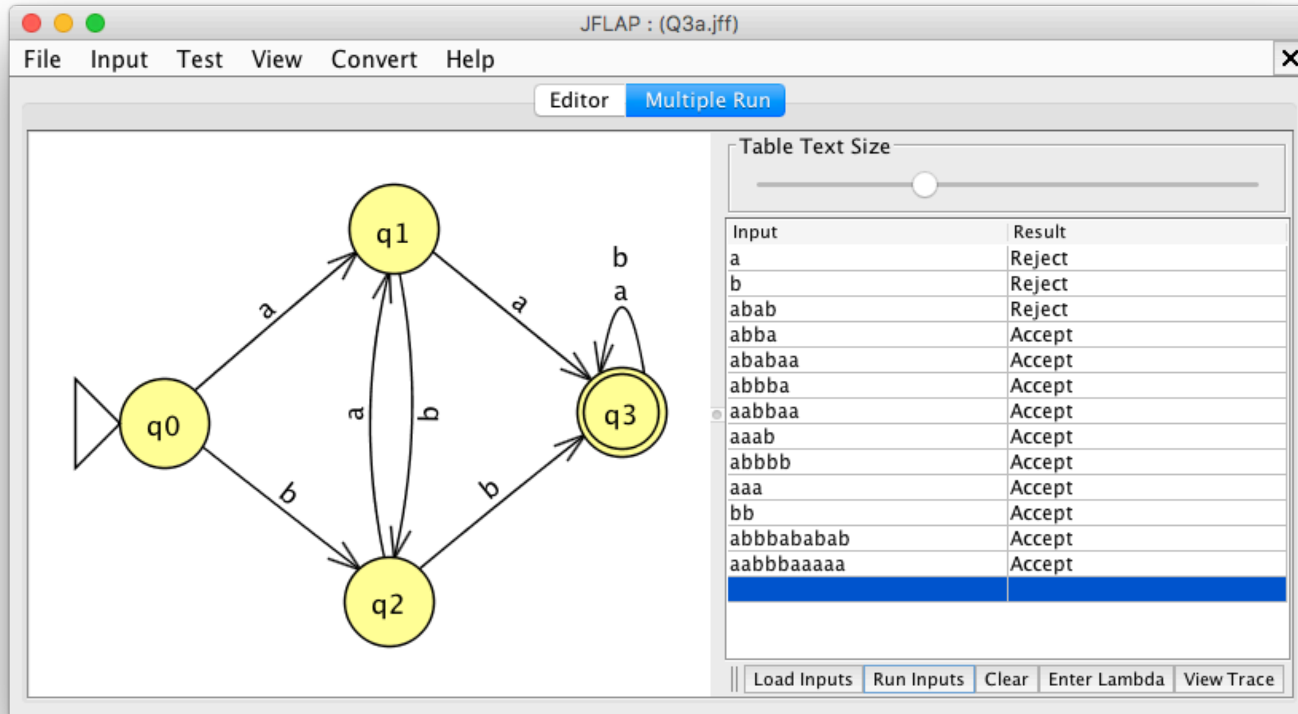
Midterm #1 Solutions: Question 3

- Let $\Sigma = \{a, b\}$.

a. Use JFLAP to construct a DFA that accepts all strings in Σ^* that contain a double letter, and test your DFA with some sample strings.

  - We want at least one double letter *aa* or *bb* to appear anywhere in the string.
  - We don't care if any other *a*'s or *b*'s appear before or after the double letter.

Midterm #1 Solutions: Question 3, *cont'd*



b. Write a regular expression that accepts the same strings.

- $(a + b)^*(aa + bb)(a + b)^*$

Midterm #1 Solutions: Question 4

- Consider this matrix that represents a simple maze:

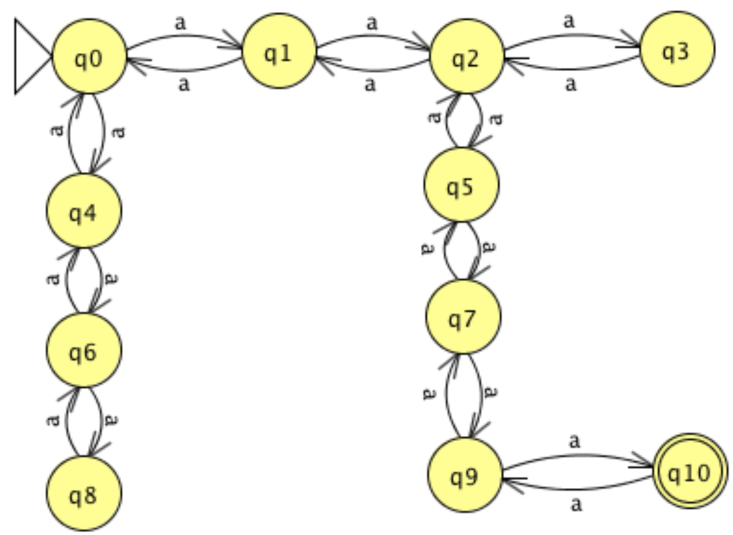
S	1	2	3
4		5	
6		7	
8		9	F

Starting from cell S, you can move horizontally or vertically but not diagonally from one numbered cell to an adjacent cell in order to reach the goal of cell F.

Midterm #1 Solutions: Question 4, *cont'd*

a. Create an NFA using JFLAP as follows: Represent each numbered cell by a state. Draw edges between the states to represent the allowable paths. Label each edge with the symbol a.

- Here, q_0 is state S and q_{10} is state F.



S	1	2	3
4		5	
6		7	
8		9	F

Midterm #1 Solutions: Question 4, *cont'd*

b. How can you use input strings for your NFA to determine the length of the shortest path from S to F?

- Since each edge is labeled with the symbol a, input to the NFA are strings containing only a's.
- Successively feed the NFA strings of a's of length 1, 2, 3, etc.
- The length of the first string that the NFA accepts must be the length of the shortest path from S to F.

Midterm #1 Solutions: Question 4, *cont'd*

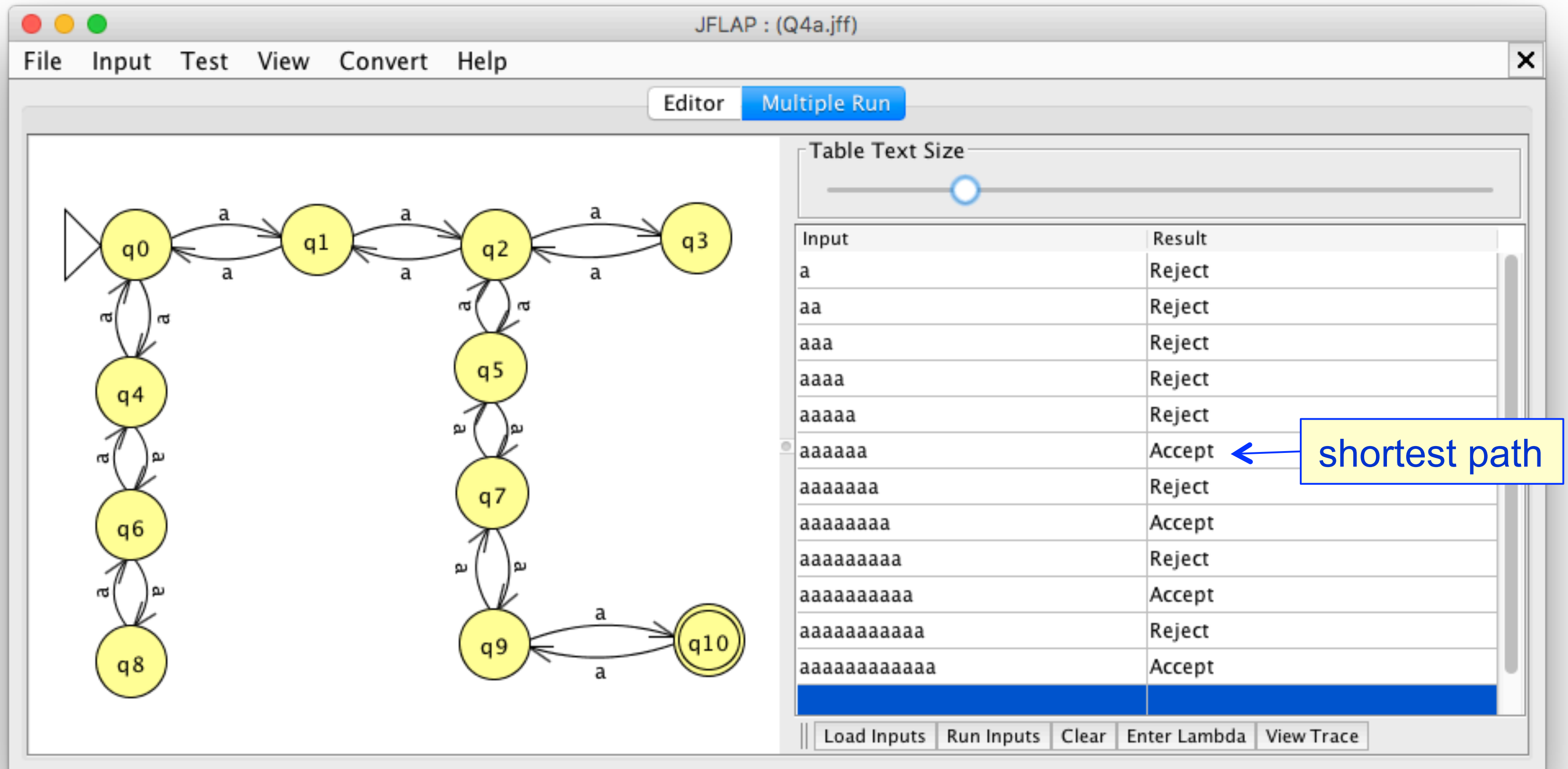
c. What happens to input strings that are longer than the length of the shortest path?

- Each string of length 8, 10, 12, etc. causes the NFA to move back and forth between two states before proceeding to state F, or from state F to state 9 and back to state F.

- Therefore, the NFA accepts all strings whose lengths are 6 or longer and an even number, and it rejects all other strings.

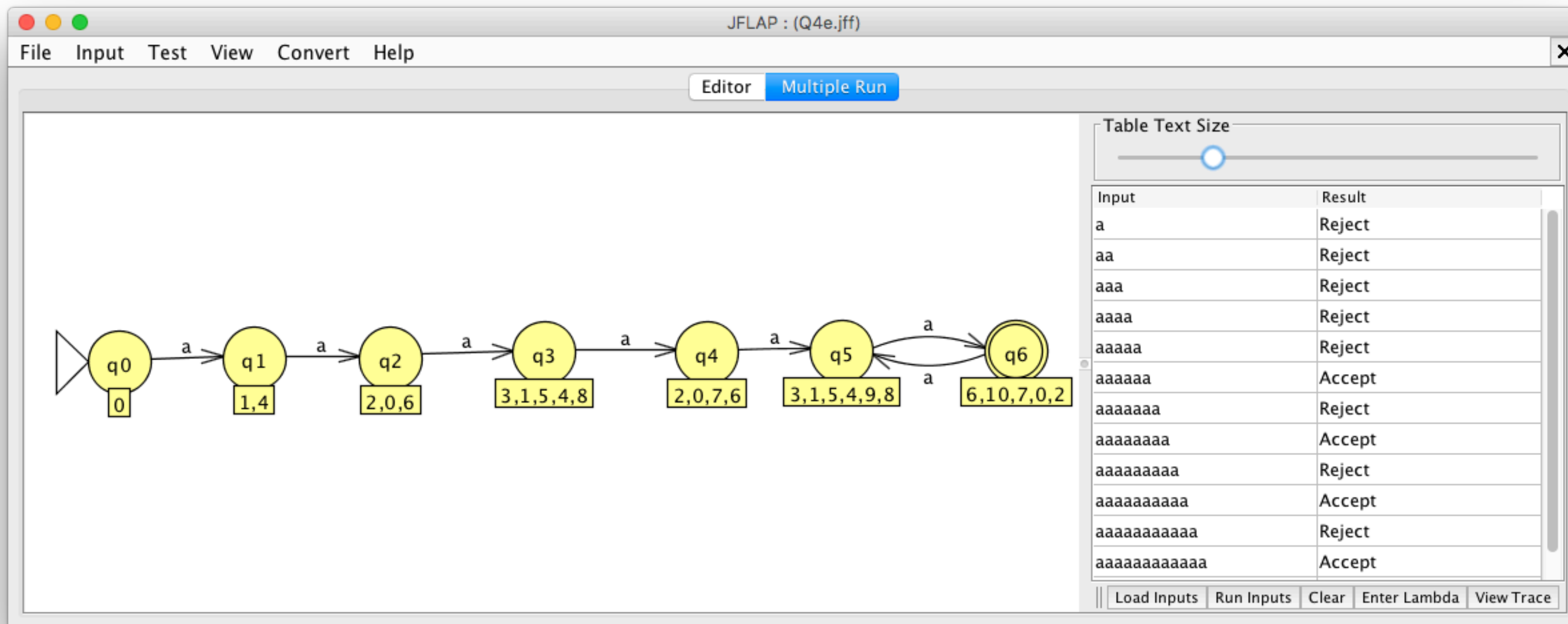
Midterm #1 Solutions: Question 4, *cont'd*

d. Use JFLAP and your NFA to demonstrate your answers to the previous two questions.



Midterm #1 Solutions: Question 4, *cont'd*

e. Use JFLAP to <u>automatically</u> convert your NFA to a minimal DFA. Test your DFA with the same input strings that you used for the NFA.



Midterm #1 Solutions: Question 4, *cont'd*

f. Explain the difference in performance between the NFA and the DFA.

- From a given state of the NFA and an input symbol a, there can be choices of where to move next.
- Therefore, to determine whether or not to accept a string, the NFA may have to backtrack to try other paths in order to find one that accepts the string.
- From a given state of the DFA and an input symbol a, there can be only one possible move.
- Therefore, the DFA does no backtracking to determine whether or not to accept a string.

Midterm #1 Solutions: Question 5

- Construct a grammar that generates the language $L = \{a^n b^n c^i : n > 0, i \geq 0\}$. Test your grammar using JFLAP with sample strings, some that are accepted and others that are rejected.
  - At least one group of consecutive a's and consecutive b's precede the optional group of consecutive c's.
  - The number of a's equals the number of b's.
  - Each string in L has two parts, $a^n b^n$ and c^i.
  - Therefore, generate each part from a separate production rule, say with nonterminal A for the $a^n b^n$ part and nonterminal B for the c^i part.
  - Then we must include the rules $A \rightarrow ab$ and $B \rightarrow \lambda$.

Midterm #1 Solutions: Question 5, *cont'd*