

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.
 - $\lambda, AuRevoir, AmourBonjourAmour$

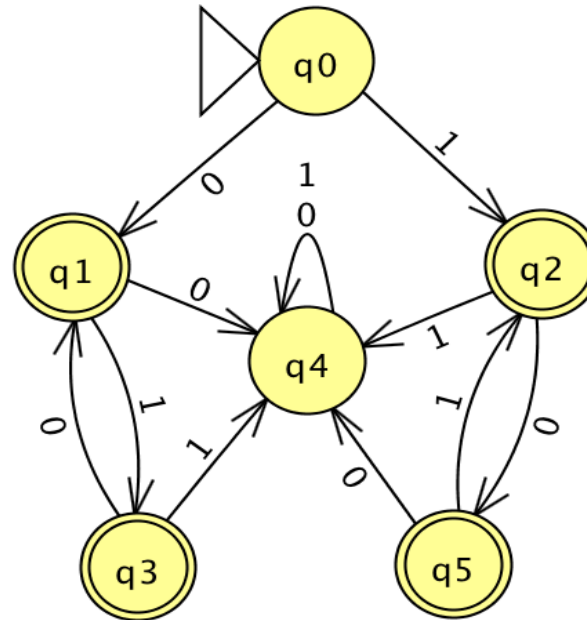
- d. Three different example members of $L_F \cup L_G$
 - Single words from either L_F or L_G
 - $Bonjour, Guten, Katzen$

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.

The screenshot shows the JFLAP interface for a DFA. The DFA has six states: q0 (start state), q1, q2, q3, q4, and q5. The transitions are as follows:

- q0 to q1 on 0
- q0 to q2 on 1
- q1 to q3 on 0
- q1 to q4 on 1
- q2 to q4 on 1
- q2 to q5 on 0
- q3 to q4 on 1
- q3 to q1 on 0
- q4 to q0 on 1
- q4 to q3 on 0
- q4 to q5 on 0
- q5 to q4 on 1
- q5 to q2 on 0

The test results table is as follows:

Input	Result
0	Accept
1	Accept
01	Accept
10	Accept
101010	Accept
0101010	Accept
10100101	Reject
010101101	Reject
0011	Reject

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*

The screenshot shows the JFLAP software interface with a finite automaton diagram and a table of test results. The automaton has four states: q0 (start), q1, q2, and q3 (final). Transitions are: q0 to q1 on 'a', q0 to q2 on 'b', q1 to q2 on 'a', q2 to q1 on 'b', q1 to q3 on 'a', and q2 to q3 on 'b'. There is a self-loop on q3 for both 'a' and 'b'.

Input	Result
a	Reject
b	Reject
abab	Reject
abba	Accept
ababaa	Accept
abbba	Accept
aabbaa	Accept
aaab	Accept
abbbb	Accept
aaa	Accept
bb	Accept
abbbababab	Accept
aabbbaaaaa	Accept

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:

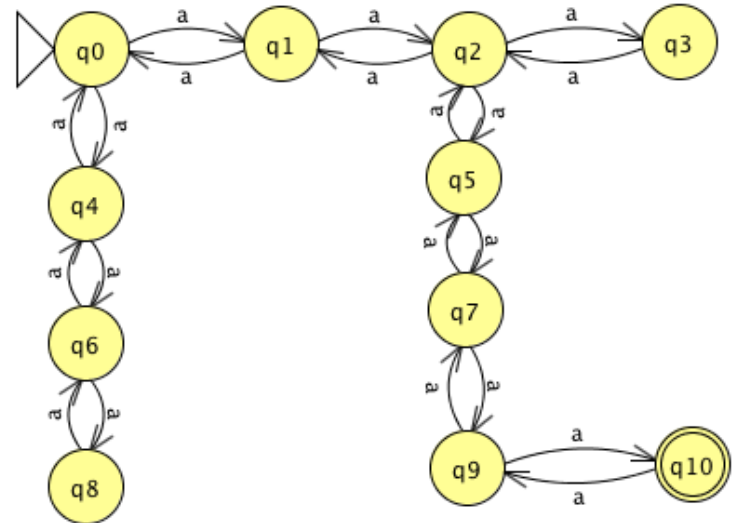
<i>S</i>	1	2	3
4		5	
6		7	
8		9	<i>F</i>

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.

The screenshot shows the JFLAP interface for a Non-deterministic Finite Automaton (NFA). The NFA has 11 states (q0 to q10). q0 is the start state, and q10 is the final state. Transitions are as follows: q0 to q1 (a), q1 to q0 (a), q1 to q2 (a), q2 to q1 (a), q2 to q3 (a), q3 to q2 (a), q0 to q4 (a), q4 to q0 (a), q4 to q5 (a), q5 to q4 (a), q5 to q6 (a), q6 to q5 (a), q6 to q7 (a), q7 to q6 (a), q7 to q8 (a), q8 to q7 (a), q7 to q9 (a), q9 to q7 (a), q9 to q10 (a), and q10 to q9 (a).

The table on the right shows the results of running various input strings:

Input	Result
a	Reject
aa	Reject
aaa	Reject
aaaa	Reject
aaaaa	Reject
aaaaaa	Accept
aaaaaaa	Reject
aaaaaaaa	Accept
aaaaaaaaa	Reject
aaaaaaaaaa	Accept
aaaaaaaaaaa	Reject
aaaaaaaaaaaa	Accept

A blue box labeled "shortest path" points to the "Accept" result for the input "aaaaaa".

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

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

The screenshot shows the JFLAP interface with the following components:

- Window Title:** JFLAP : (Q4e.jff)
- Menu Bar:** File, Input, Test, View, Convert, Help
- Buttons:** Editor, Multiple Run
- NFA Diagram:** A sequence of states q0 to q6. q0 is the start state (triangle) and q6 is the final state (double circle). Transitions are labeled 'a'. Each state has a box below it containing a set of integers: q0 (0), q1 (1,4), q2 (2,0,6), q3 (3,1,5,4,8), q4 (2,0,7,6), q5 (3,1,5,4,9,8), q6 (6,10,7,0,2).
- Table Text Size:** A slider control.
- Table:** A table with two columns: Input and Result. It lists 12 test cases.
- Buttons at the bottom:** Load Inputs, Run Inputs, Clear, Enter Lambda, View Trace

Input	Result
a	Reject
aa	Reject
aaa	Reject
aaaa	Reject
aaaaa	Reject
aaaaaa	Accept
aaaaaaa	Reject
aaaaaaaa	Accept
aaaaaaaaa	Reject
aaaaaaaaaa	Accept
aaaaaaaaaaa	Reject
aaaaaaaaaaaa	Accept

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*

The screenshot shows the JFLAP software interface for a grammar. The window title is "JFLAP : (Q5.jff)". The menu bar includes "File", "Input", "Test", "Convert", and "Help". The "Multiple Run" button is active. The interface is divided into several sections:

- Table Text Size:** A slider control.
- Start Step:** A dropdown menu set to "Noninverted Tree".
- Input:** A text field for entering a string.
- Grammar Rules:** A table with LHS and RHS columns.
- Execution Results:** A table with Input and Result columns.
- Buttons:** "Load Inputs", "Run Inputs", "Clear", and "Enter Lambda".

The grammar rules are as follows:

LHS	RHS
S	→ AB
A	→ aAb
A	→ ab
B	→ cB
B	→ λ

The execution results are as follows:

Input	Result
abc	Accept
ab	Accept
c	Reject
aaabbbcc	Accept
aaaabbbc	Reject
aaaaabbbbb	Accept
ccc	Reject
abababc	Reject
ccaaabbb	Reject