PROJECT 2
DFA & PDA

DETERMINISTIC FINITE AUTOMATON

PUSH DOWN AUTOMATON

CSc 304 THEORETICAL OF COMPUTER SCINCE

PROF. LUCCI

BY:

Group 2

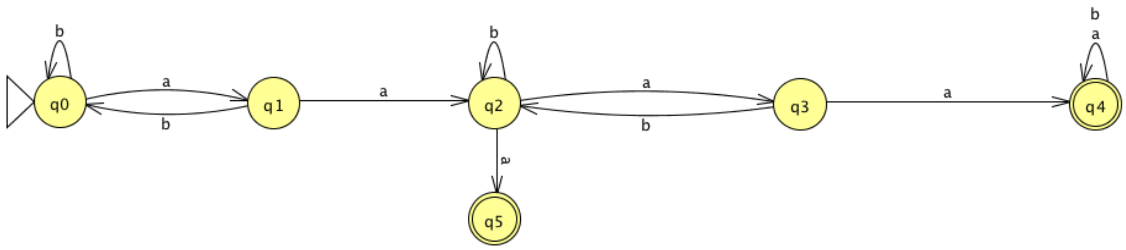
Edwards Ames

Akbar Mirza

Problem 1

Design a dfa for:

L = the set of strings over **{a, b}** in which the substring **aa** occurs at least twice



Testing

The results of Multiple Run testing in *JFLAP* can be seen below.

Input	Result
a	Reject
b	Reject
aaa	Accept
baab	Reject
aaaa	Accept
baaaaab	Accept
baabaab	Accept
aabaa	Accept
ababaabaab	Accept
	Reject

As you can see, our deterministic finite automata accepts any input string that has at least two substrings of *aa*. The final input string is the empty string (epsilon) and it is rejected because there are no substrings of *aa* in it.

Analysis

Running Time

- Given the worst-case scenario: *aabb…bbaa*
    - when
- We would need to read n items to accept or fail to accept our input
- Therefore, we have a running time of **O(n)**
    - i.e. $t(n) = O(n)$

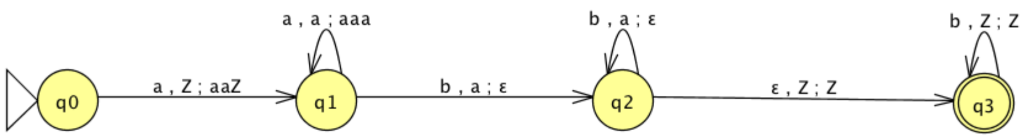
Space Complexity

- Given the size of input, n, our space complexity for our deterministic finite automaton is n
    - We don't need any extra memory
- Therefore, we have a space complexity of **n**

Problem 2

Design a pda for:

$$L = \{a^n b^m, m, n >= 0, m >= 2 * n\}$$



Testing

The results of Multiple Run testing in *JFLAP* can be seen below.

Input	Result
a	Reject
b	Reject
ab	Reject
abb	Accept
abbbb	Accept
aab	Reject
aabb	Reject
aabbbb	Accept
aaabbbbbb	Accept
	Reject

As you can see, our pushdown automaton accepts any input string that has at least twice as many *b*'s as *a*'s. The final input string is the empty string (epsilon) and it is rejected because it is not part of our language by definition.

Analysis

Running Time

- Given the worst-case scenario: *aa…abb…bb*
    - The number of b's (m) is arbitrarily larger than twice the number of a's (n)
- We would need to read n items to accept or fail to accept our input
- Therefore, we have a running time of **O(n)**
    - i.e. $t(n) = O(n)$

Space Complexity

- Let *x* be the length of our input
- Our stack requires at least 2 * n (number of a's) cells to accommodate our machine
- Therefore, we have a space complexity of **x + 2n**