CS154

Name: _____ Date: _____

Note: The purpose of the following questions is:

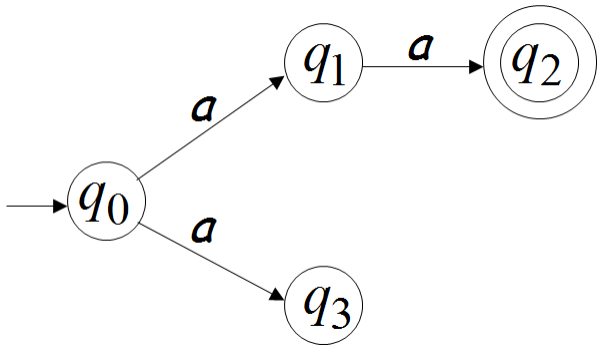
• *Enhance learning*	• *Summarized points*	• *Analyze abstract ideas*

Class 3: None Deterministic Automata

Finite accepters are more complicated if we allow them to act nondeterministically. Nondeterminism is a powerful but, at first sight, unusual idea. We normally think of computers as completely deterministic, and the element of choice seems out of place. Nevertheless, nondeterminism is a useful notion, as we shall see as we proceed.

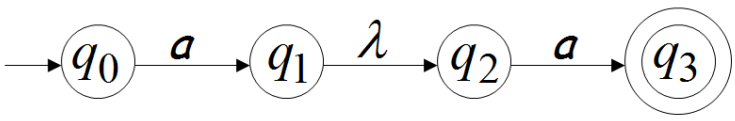
1. Define Non deterministic Finite Accepters (nfa)?
   Nondeterminism means a choice of moves for an automaton. Rather than prescribing a unique move in each situation, we allow a set of possible moves. Formally, we achieve this by defining the transition function so that its range is a set of possible states.

2. Why the automaton in the following figure is nondeterministic?



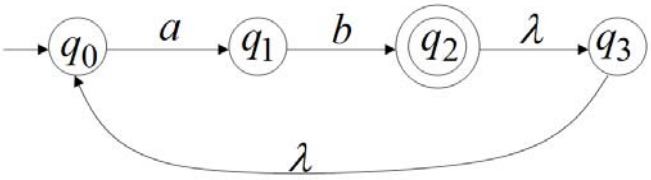
   This automaton accepts Alphabet = {a}, check the acceptance for the following strings, show the different choices for every case.
   i. aa [Slide 5-12]
   ii. a [Slide 15-20]
   iii. aaa [Slide 23-31]
   What is the Language accepted by the automata [Slide 32]

3. For the following NFA, check the acceptance for the following strings



   i. aa [Slide 34-38]
   ii. aaa [Slide 39-43]
   What is the Language accepted by the automata? [Slide 44]

4. For the following NFA, check the acceptance for the following strings

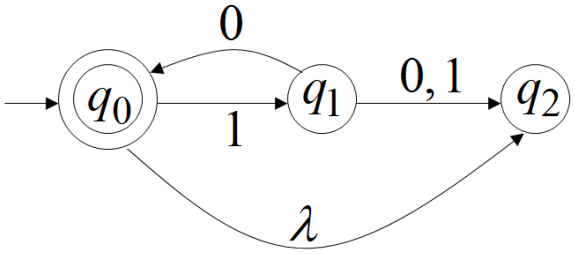


i. ab [Slide 46-49]

ii. abab [Slide 50-57]

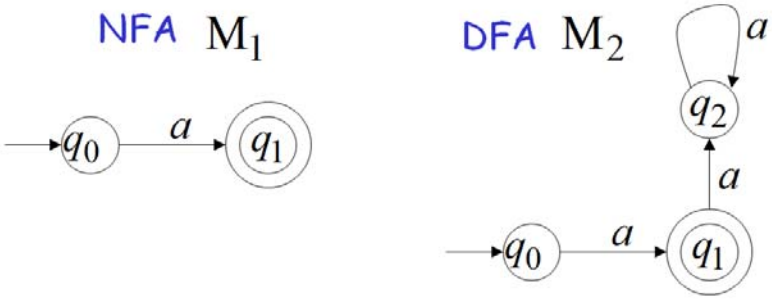
What is the Language accepted by the automata? [Slide 58]

5. [Slide 60] What is the Language accepted by the following NFA?



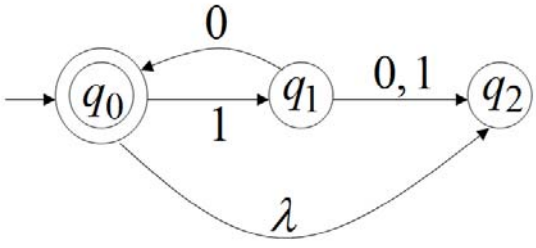
6. [Slide 61] What is the Language accepted by the following Machines?

7. [Slide 62] What is the Language accepted by the following Machines?



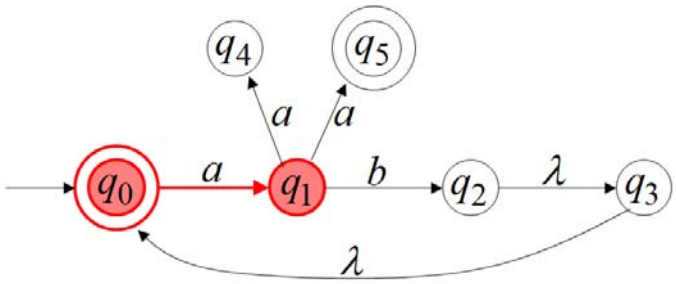
8. [Slide 63] What is the Formal Definition of NFAs?

9. [Slide 64-67] For the following NFA:



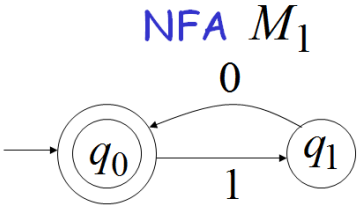
i. $\delta(q_0, 1) =$ [Slide 64]
ii. $\delta(q_0, 0) =$ [Slide 65]
iii. $\delta(q_0, \lambda) =$ [Slide 66]
iv. $\delta(q_2, 1) =$ [Slide 67]

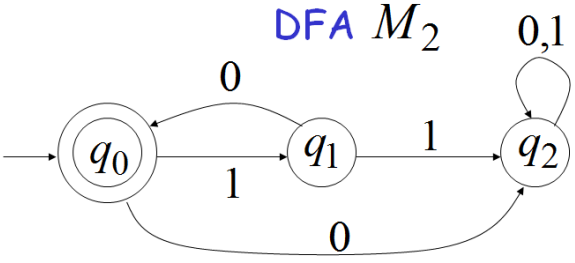
10. [Slide 68-76] For the following NFA:



i. $\delta^*(q_0, a) =$ [Slide 68]
ii. $\delta^*(q_0, aa) =$ [Slide 69]
iii. $\delta^*(q_0, ab) =$ [Slide 70]
iv. $\delta^*(q_0, ab) =$ [Slide 73]
v. $\delta^*(q_0, abaa) =$ [Slide 74]
vi. $\delta^*(q_0, aba) =$ [Slide 75]
What is the Language accepted by the automata? [Slide 76]

11. [Slide 81] What is the Language accepted by the following Machines? And what is your observations

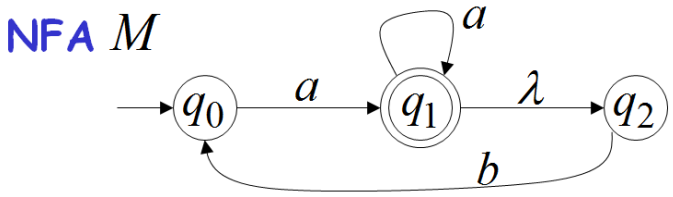




12. Why Nondeterminism?
    - Digital computers are completely deterministic; their state at any time is uniquely predictable from the input and the initial state.
    - Why we study nondeterministic machines at all. We are trying to model real systems, so why include such nonmechanical features as choice?
    - A typical example is a game-playing program. Frequently, the best move is not known, but can be found using an exhaustive search with back- tracking. When several alternatives are possible, we choose one and follow it until it becomes clear whether or not it was best. If not, we retreat to the last decision point and explore the other choices.
    - A nondeterministic algorithm that can make the best choice would be able to solve the problem without backtracking, but a deterministic one can simulate nondeterminism with some extra work. For this reason, nondeterministic machines can serve as models of search-and-backtrack algorithms.
    - Nondeterminism is sometimes helpful in solving problems easily.
    - [Slide 76] The language is the union of two quite different sets, and the nondeterminism lets us decide at the outset which case we want. The deterministic solution is not as obviously related to the definition, and so is a little harder to find.

13. Since a **dfa** is in essence a restricted kind of **nfa**, it is clear that any language that is accepted by some **dfa** is also accepted by some **nfa** for which in principle, we cannot find a **dfa**. But it turns out that this is not so. The classes of dfa's and nfa's are equally powerful: For every language accepted by some nfa there is a dfa that accepts the same language. Prove the last statement. [Slide 82-84]

14. [Slide 85-101] Define step-by-step procedure to convert NFA to equivalent DFA. Use the procedure to convert the following NFA to equivalent DFA



15. [Slide 102-113] Prove the following theorem:

Let L be the language accepted by a nondeterministic finite accepter $M_N = (Q_N, \Sigma, \delta_N, q_0, F_N)$. Then there exists a deterministic finite accepter $M_D = (Q_D, \Sigma, \delta_D, \{q_0\}, F_D)$ such that

$$L = L(M_D).$$

Reading:

An Introduction to Formal Language and Automata, Peter Linz, 5th edition, Sec 2.2, 2.3.

Selected Exercises:

Section 2.2: 3, 5, 8, 9, 16, 18, 21.

Section 2.3: 2, 7, 8, 11, 14, 15.