San José State University
Department of Computer Science

Ahmad Yazdankhah

ahmad.yazdankhah@sjsu.edu
www.cs.sjsu.edu/~yazdankhah

Nondeterministic Finite Automata

(Part 3)

Lecture 11

Day 12/31

CS 154

Formal Languages and Computability

String 2018

Agenda of Day 12

- Summary of Lecture 10
- Lecture 11: Teaching ...
  - Nondeterministic Finite Automata (Part 3)
- Solution and Feedback of Quiz +

Solution and Feedback of Quiz + (Out of 40)

Metrics	Section 1	Section 2	Section 3
Average	33	32	32
High Score	39	40	38
Low Score	22	24	19

Summary of Lecture 10: We learned ...

NFAs

- NFAs are interesting because ...
  - their transition graphs are simpler.
- Associated language to an NFA is ...
  - ... the set of all strings that it accept.
  - This is a general definition for all type of automata.

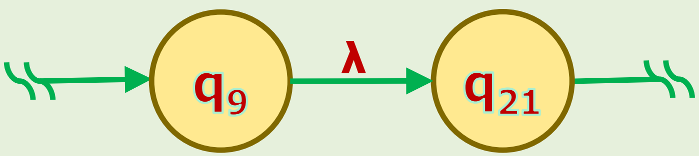
λ-transition

- Short circuit is ...
  - ... an edge with no label (symbol).
- We represent it with symbol λ.
- The transition is called λ-transition.
  - In fact λ means "NO symbol".
- λ-transition is a special transition.
  - It is the 3rd violation of DFAs definition.
- λ-transition in automata theory means ...
  - ... the machine may "unconditionally" transit ...
  - ... in the same timeframe, without consuming input.

Summary of Lecture 10: We learned ...

NFAs

- The sub-rule of the following transition is ...



$$\delta\,(q_9\,,\,\lambda) = \{q_9\,,\,q_{21}\}$$

- To accommodate the λ-transitions in the NFAs' formal definition, we modified the definition of δ as ...

$$\delta: Q \times (\Sigma \cup \{\lambda\}) \rightarrow 2^Q$$

- When an NFA encounters a λ-transition, it has multiple choices for transition.
- It would check all possibilities by parallel processing.

Any question?

Definitions

Formal Definition of NFAs

- An NFA M is defined by the quintuple (5-tuple):

$$M = (Q, \Sigma, \delta, q_0, F)$$

- Where:
  - Q is a finite and nonempty set of states of the transition graph.
  - Σ is a finite and nonempty set of symbols called input alphabet.
  - δ is called transition function and is defined as:

$$\delta : Q \times (\Sigma \cup \{\lambda\}) \to 2^Q$$

δ is total function.

  - $q_0 \in Q$ is the initial state of the transition graph.
  - $F \subseteq Q$ is the set of accepting states of the transition graph.

- Except δ, the rest items are similar to DFAs'.

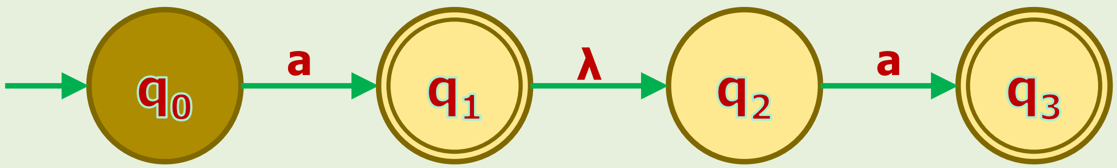
NFAs vs DFAs

	NFAs	DFAs
Transition function	$\delta : Q \times (\Sigma \cup \{\lambda\}) \to 2^Q$	$\delta : Q \times \Sigma \to Q$
Examples	$\delta(q_1, a) = \{q_2, q_5, q_3\}$ $\delta(q_1, \lambda) = \{q_1, q_3\}$ $\delta(q_x, a) = \{\ \}$	$\delta(q_1, a) = q_2$
Type of function	Total	Total
Type of processing	Parallel processing	Single processing

Associated Language to NFAs Examples

Example 16

- What is the associated language to the following NFA?

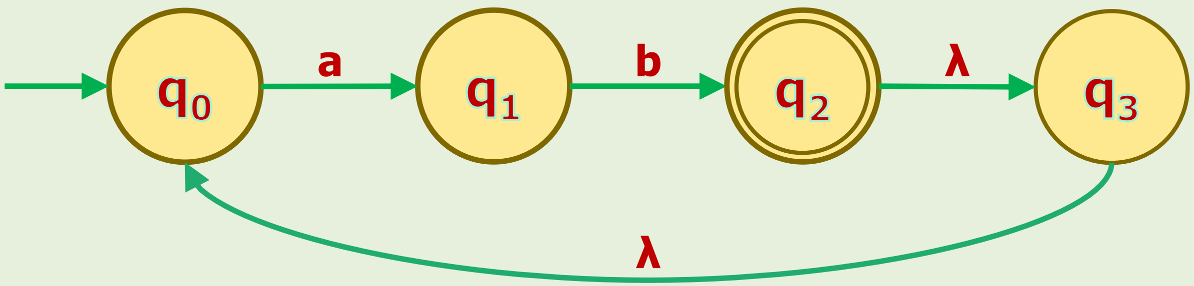


- $L(M) = \{a, aa\}$

Associated Language to NFAs Examples

Example 17

- What is the associated language to the following NFA?



- $L = \{ab, abab, ababab, \dots\}$

  $= \{(ab)^n : n \geq 1\}$

NFA Design Example

Example 18

- Design an NFA and a DFA with 3 states for the following language over $\Sigma = \{a , b\}$.
  - The set of all strings that ends with aa.

Homework

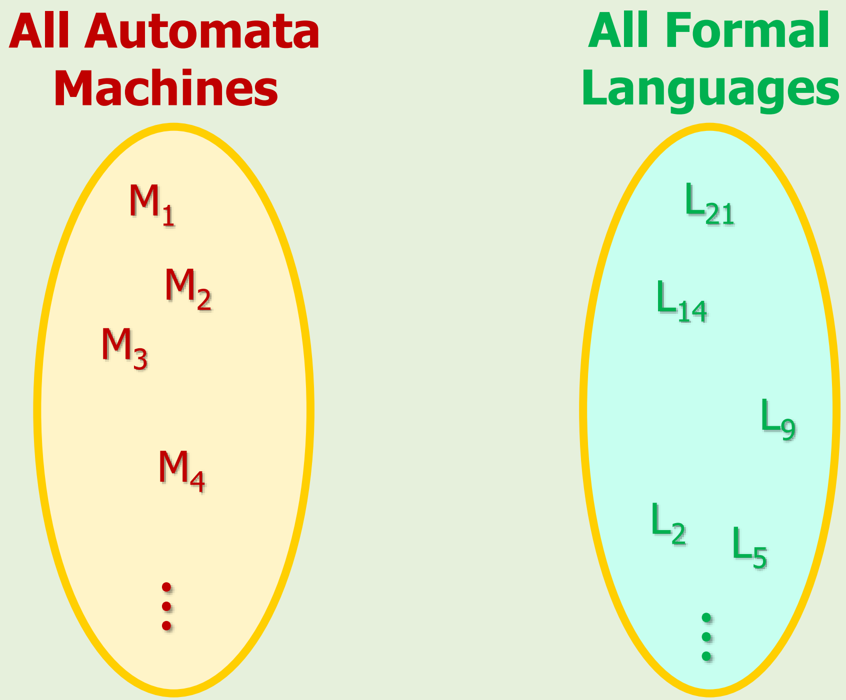
1. Let $L = \{a^n b : n \geq 0\}$, and $L' = L\,(L \cup \{\lambda\})$ over $\Sigma = \{a , b\}$.
   – Design an NFA with 3 states for accepting L'.

2. Design an NFA for each of the following languages.
   a. $L = \{a^n b^m a^k : n, m \geq 0, k \geq 1\}$ with 3 states over $\Sigma = \{a , b\}$
   b. $L = \{(ab)^n (abc)^m : n \geq 0, m \geq 0\}$ over $\Sigma = \{a , b , c\}$

Machines and Languages Association

Machines and Languages Association

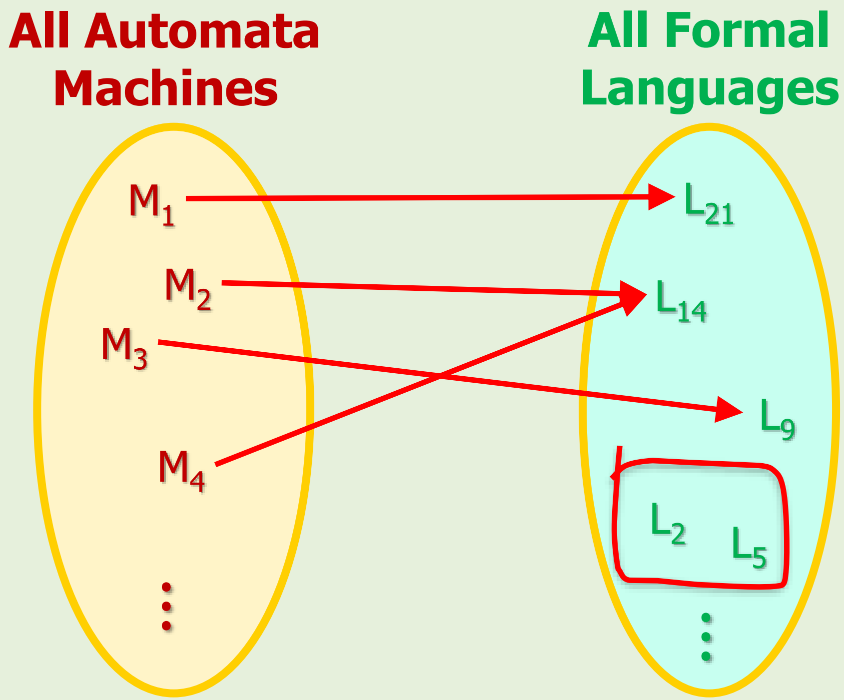
- What is the relationship between the set of all automata machines and the set of all formal languages?



One of the most interesting topics of computer science

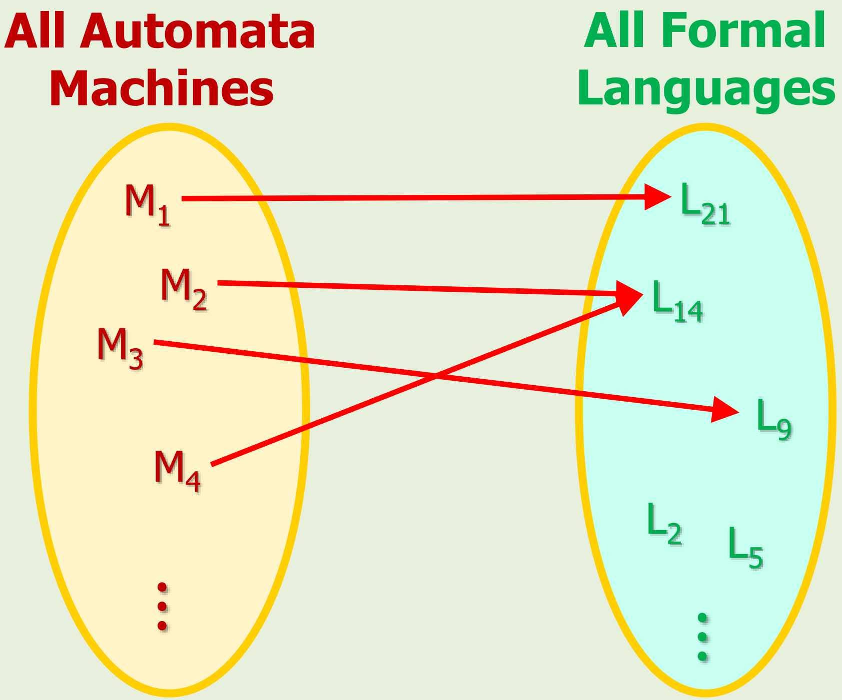
Machines and Languages Association

- So far, we learned that "every machine has an associated language".
- BUT we don't know yet whether or not for every language, we can construct a machine!
  - Our knowledge is not enough yet.



Machines and Languages Association

- Can we consider this relationship as a function?
  - Yes, the definition of the function can be: $L : M \rightarrow L(M)$
- What type of function is this?
  - Total function!



A Side Note: Computer Scientists Mission

- Why should we be interested in the relationship between machines and languages?
- Recall that:

Language ≡ Problem

Accepting (understanding, recognizing) a language
≡ Solving a problem

- So, as computer scientists, our mission is:
- To find a machine for every language ≡ To solve the problems
- This is actually the soul of this course!

DFAs vs NFAs

Objective

- The goal of this section is to compare two classes DFAs and NFAs.
- To compare two classes of automata, we'd need a "metrics".
- We'll use the concept of "power" as the metrics for comparison.
- So, first we need to define "power".

Power of Automata Classes

- Let's assume we have two classes of automata:
  - Class A (e.g. NFAs)
  - Class B (e.g. DFAs)

Question

- What is the best criteria to claim that:

Class A is "more powerful" than class B?

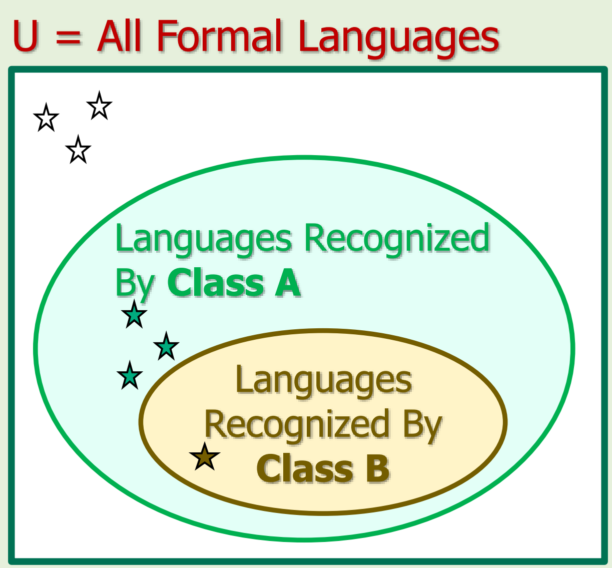
Answer

- If class A can solve more problems, then it is more powerful.

Power of Automata Classes

Definition

- The (automata) class A is "more powerful" than class B iff the set of languages recognized by class B is a proper subset of the set of the languages recognized by class A.

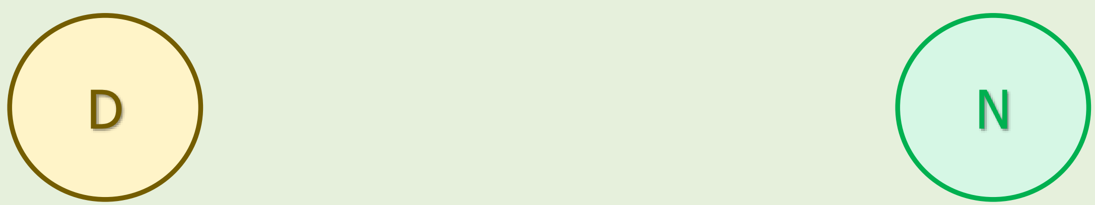


DFAs and NFAs Relationship

- Let's get back to our topic: DFAs vs. NFAs
- If the universal set is the set of all formal languages:
  - What portion of the formal languages can be recognized by DFAs?
  - What portion can be recognized by NFAs?
- Let's use the following definitions:

U = {x : x is a formal language}

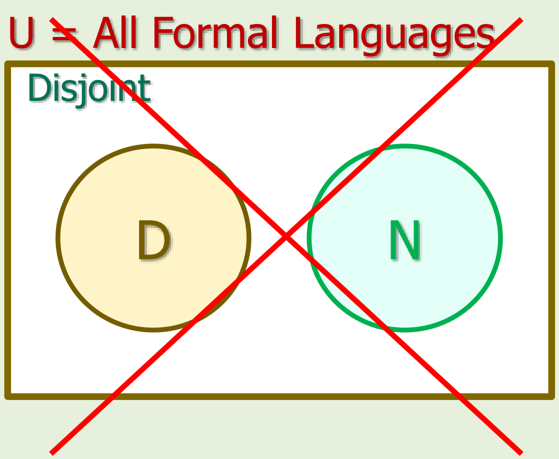
D = {x : x is recognized by a DFA}   N = {x : x is recognized by a NFA}

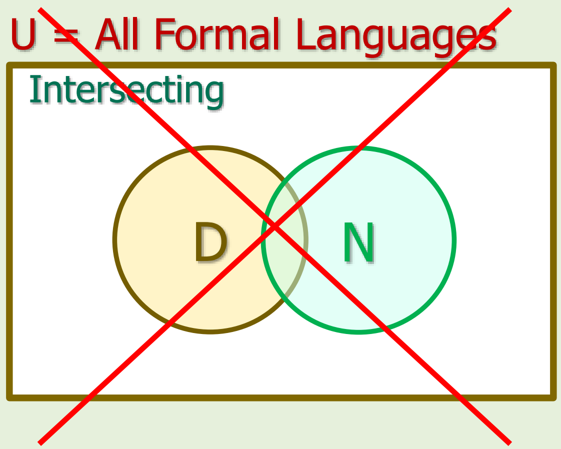


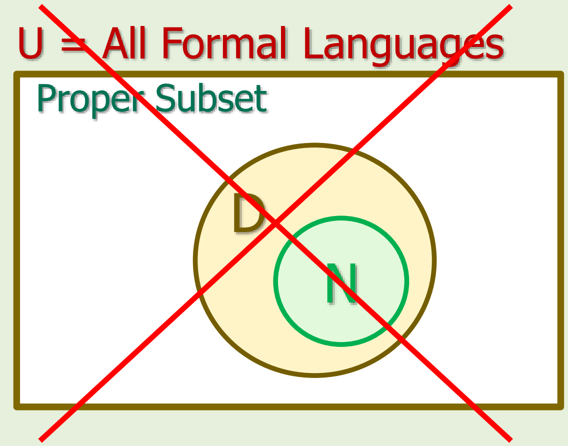
- What is the relationship between the sets D and N?

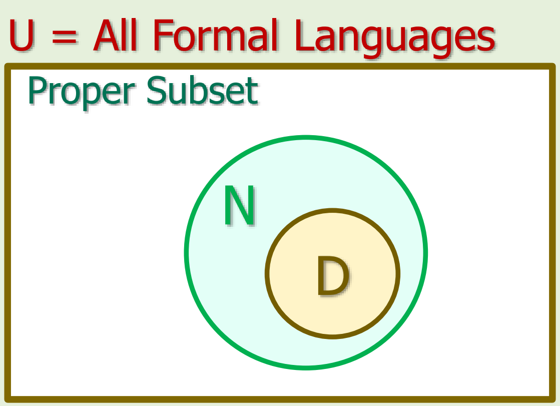
DFAs and NFAs Relationship

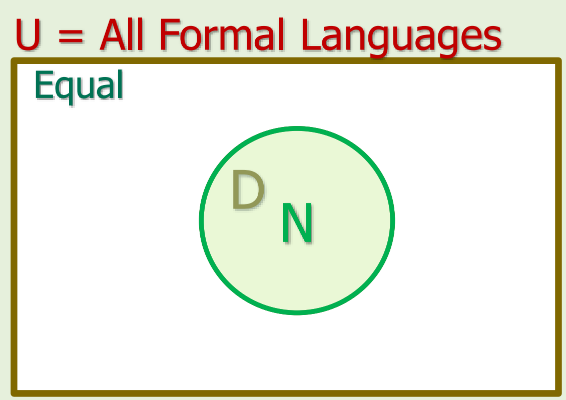
- Which one is reasonable relationship between D and N?











Can NFAs Do Whatever DFAs Can Do?

- Let's assume that we've constructed a DFA for an arbitrary language L.
- Can we always construct an NFA for L?
- Yes, but how?
- Mathematically speaking, the only difference between the definition of NFAs and DFAs is their transition function.
- So, we should prove that we can always convert a DFA's definition to an NFA's definition.
- Let's show this through an example.

Can NFAs Do Whatever DFAs Can Do?

Example 19

- Convert the following DFA's definition to an NFA's.
- q_0 is the initial state, and q_1 is the final state.

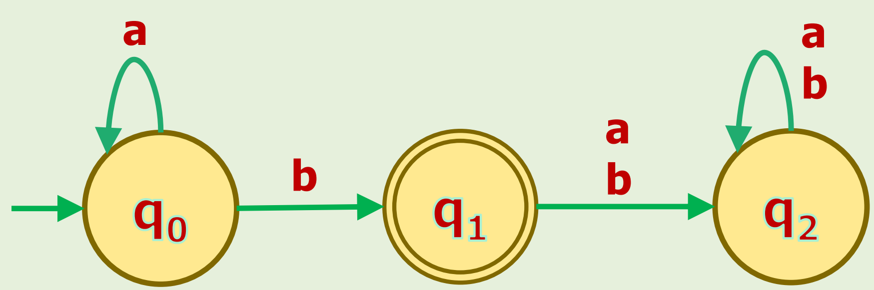
$$\delta: \begin{cases} \delta(q_0, a) = q_0 \\ \delta(q_0, b) = q_1 \\ \delta(q_1, a) = q_2 \\ \delta(q_1, b) = q_2 \\ \delta(q_2, a) = q_2 \\ \delta(q_2, b) = q_2 \end{cases}$$

DFA

$$\delta: \begin{cases} \delta(q_0, a) = \{q_0\} \\ \delta(q_0, b) = \{q_1\} \\ \delta(q_1, a) = \{q_2\} \\ \delta(q_1, b) = \{q_2\} \\ \delta(q_2, a) = \{q_2\} \\ \delta(q_2, b) = \{q_2\} \end{cases}$$

NFA

- Just convert the δ.
- The rest items are the same.



DFAs Can be Converted to NFAs

	DFA	NFA
States	$Q = \{q_0, q_1, q_2\}$	$Q = \{q_0, q_1, q_2\}$
Alphabet	$\Sigma = \{a, b\}$	$\Sigma = \{a, b\}$
Sub-rule	$\delta\ (q_i, a) = q_j$	$\delta\ (q_i, a) = \{q_j\}$
Initial state	q_0	q_0
Final states	$F = \{q_1\}$	$F = \{q_1\}$

Can NFAs Do Whatever DFAs Can Do?

- As the previous example showed, there is a simple algorithm to convert a DFA to an NFA.

Algorithm: Converting DFAs' Formal Definition to NFAs'

- Change all DFAs' sub-rules to NFAs format by enclosing the range element with a pair of curly brackets. i.e.:

$$\delta\ (q_i\ , x) = q_j$$

changes to

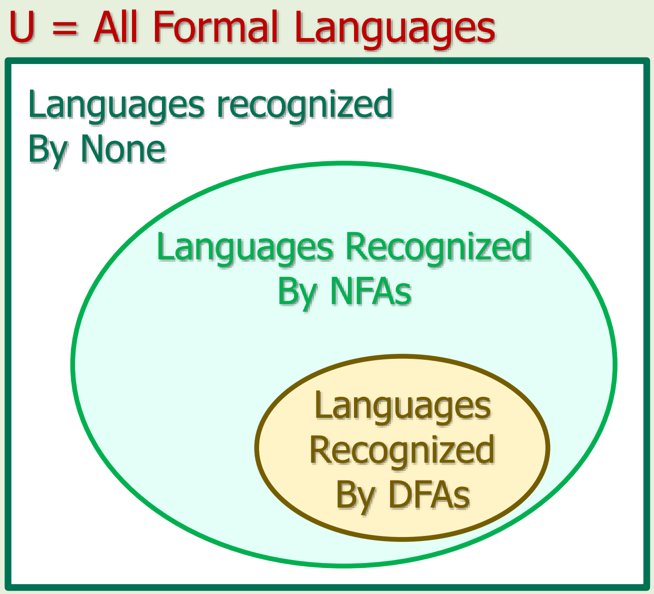
$$\delta\ (q_i\ , x) = \{q_j\}$$

- The rest of the definitions, (i.e. Q, Σ, q_0, F) are the same.

Can NFAs Do Whatever DFAs Can Do?

Conclusion

- Can NFAs do whatever DFAs can do?
- Yes, the set of all languages recognized by DFAs can be recognized by NFAs too.



- Now, let's ask another question ...

Can DFAs Do Whatever NFAs Can Do?

- Let's assume that we've constructed an NFA for an arbitrary language L.
- Can we always construct a DFA for L?
- The answer of this question is not so obvious.
- Let's take an example to make it clear.

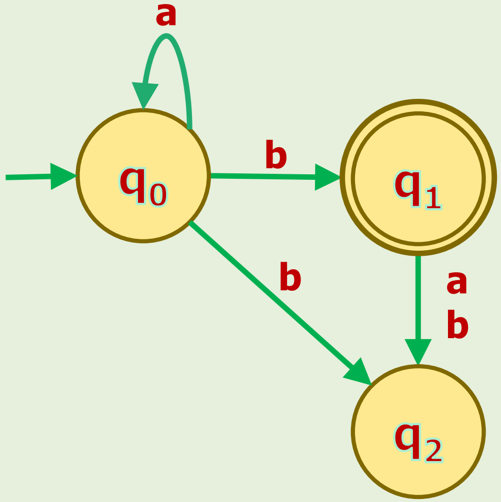
Can DFAs Do Whatever NFAs Can Do?

Example 20

- Can we convert the following NFA to a DFA?
- q_0 is the initial state, and q_1 is the final state.

$$\delta: \begin{cases} \delta(q_0, a) = \{q_0\} \\ \delta(q_0, b) = \{q_1, q_2\} \\ \delta(q_1, a) = \{q_2\} \\ \delta(q_1, b) = \{q_2\} \\ \delta(q_2, a) = \{\,\} \\ \delta(q_2, b) = \{\,\} \end{cases}$$

NFA $\Rightarrow$? DFA



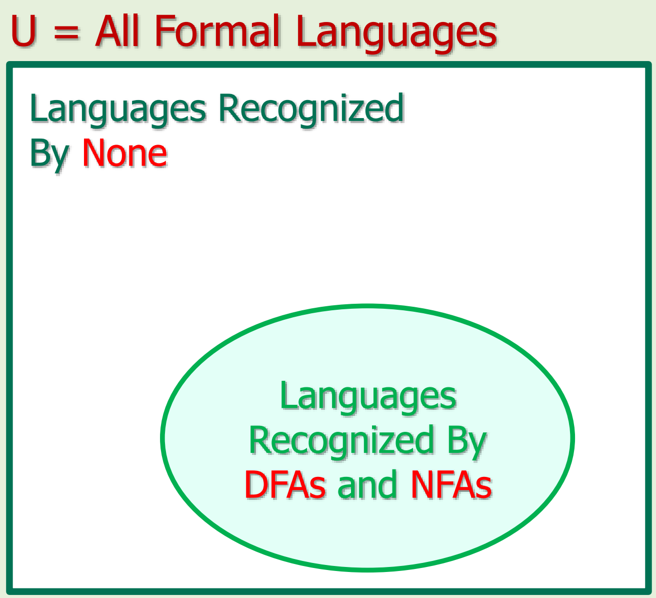
- Yes, but it needs a special technique to convert an NFA to DFA.
- We might cover it later if we have time!

DFAs Class and NFAs Class are Equivalent

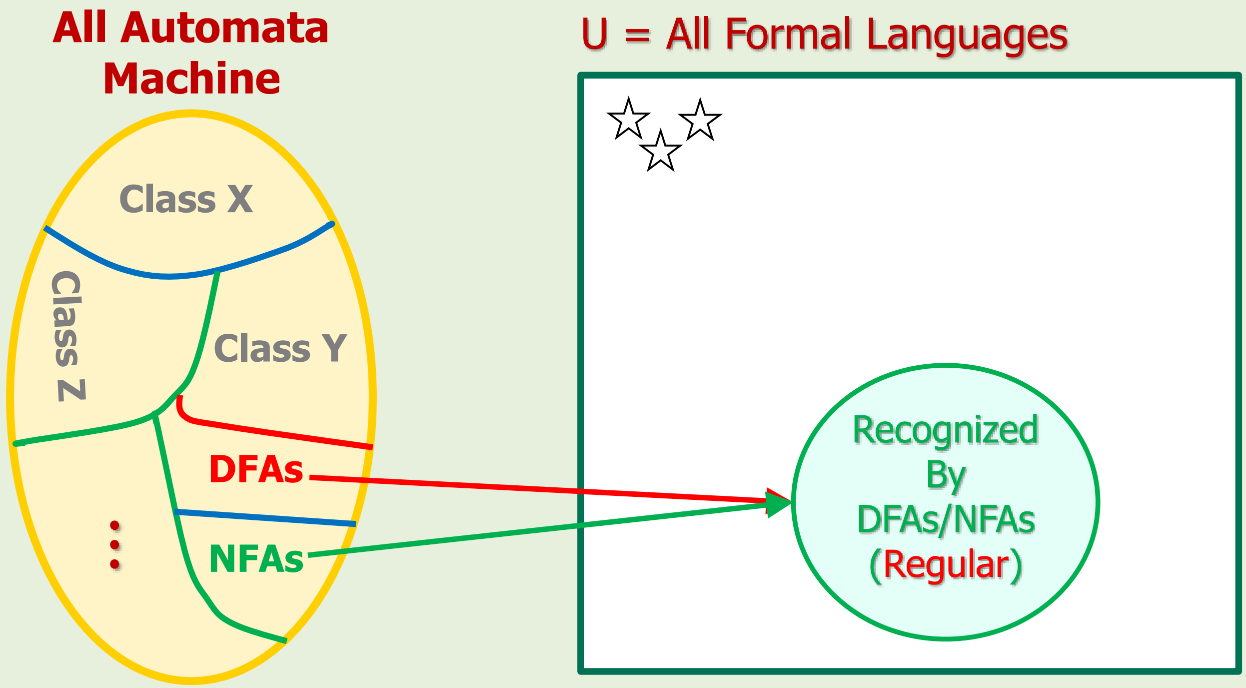
- DFAs and NFAs are equivalent as the following theorem proves.

Theorem

- The set of languages recognized by NFAs are equal to the set of languages recognized by DFAs.



Machines and Languages Association

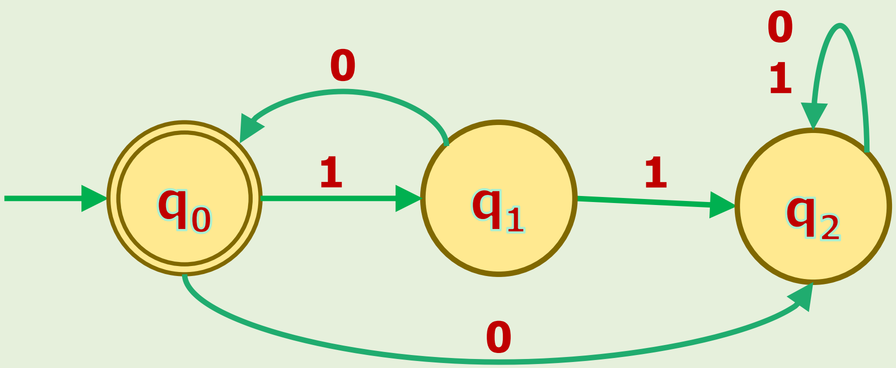


- DFAs and NFAs have the same power because both recognize the same portion of languages.
- Later we'd define other classes of machines (i.e. Class X, Y, Z, etc.) and the languages they are associated with.

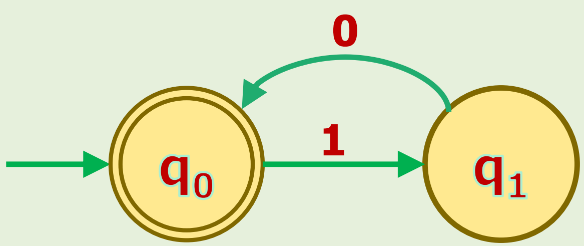
Equivalency of DFAs and NFAs Example

Example 21

- What are the associated languages to the following machines?



DFA



NFA

$L_1 = \{(10)^n : n \geq 0\}$ $L_2 = \{(10)^n : n \geq 0\}$

- They are equivalent because both have the same associated languages.

References

1. Linz, Peter, "An Introduction to Formal Languages and Automata, 5th ed.," Jones & Bartlett Learning, LLC, Canada, 2012
2. Kenneth H. Rosen, "Discrete Mathematics and Its Applications, 7th ed.," McGraw Hill, New York, United States, 2012
3. Michael Sipser, "Introduction to the Theory of Computation, 3rd ed.," CENGAGE Learning, United States, 2013
   ISBN-13: 978-1133187790