Unit - I | Lecture- 06

CONVERSION OF NFA TO DFA

Dr. R. Madana Mohana
Professor, Artificial Intelligence & Data Science | I/c-Head, Artificial Intelligence & Machine Learning
CHAITANYA BHARATHI INSTITUTE OF TECHNOLOGY
Hyderabad - 500 075, Telangana, INDIA
www.cbit.ac.in

Nondeterministic Finite Automata(NFA)

- Conversion of NFA to DFA (Equivalence of NFA and DFA)
- An Application: Text Search

Equivalence of NFA and DFA:

Theorem:

For every **NFA** there exists an equivalent **DFA**.

(or)

For every **NFA** which accepts language **L** and for equivalent **DFA** also accepts the same language **L**.

Equivalence of NFA and DFA:

Procedure:

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a NFA and

Let $M' = (Q', \Sigma, \delta', q_0, F')$ be an equivalent DFA

Where $Q' = 2^Q$; $q_0 = [q_0]$

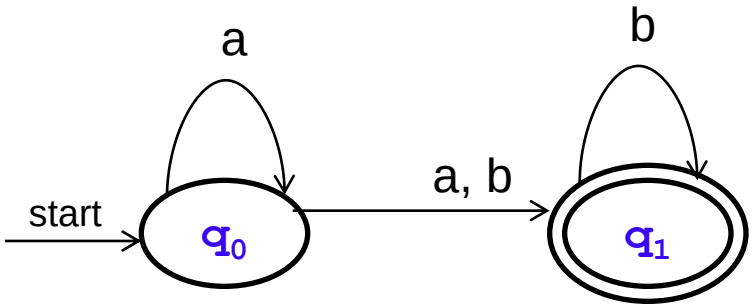
F' is set of states in Q' which contains a final or accepting state of **NFA**

$\delta'([q_0], a) = \delta(q_0, a)$ for every $q_0 \epsilon Q'$

Equivalence of NFA and DFA:

Example-1:

Construct an equivalent DFA for the given NFA.



Equivalence of NFA and DFA:

Example-1: Solution

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

$where$

$$Q' = 2^Q = 2^2 = 4 \ \ states$$

$$Q' = \{\phi, [q_0], [q_1], [q_0, q_1]\}$$

$$\Sigma = \{a, b\}$$

$$q_0 = [q_0]$$

$$F' = \{[q_1], [q_0, q_1]\}$$

Equivalence of NFA and DFA:

Example-1: Solution cont.1

$$1.\delta'([q_0],a) = \delta(\{q_0\},a) = \delta(q_0,a) = \{q_0,q_1\} = [q_0,q_1]$$
$$\delta'([q_0],b) = \delta(\{q_0\},b) = \delta(q_0,b) = \{q_1\} = [q_1]$$

$$2.\delta'([q_1],a) = \delta(\{q_1\},a) = \delta(q_1,a) = \phi$$
$$\delta'([q_1],b) = \delta(\{q_1\},b) = \delta(q_1,b) = \{q_1\} = [q_1]$$

$$3.\delta'([q_0,q_1],a) = \delta(\{q_0,q_1\},a) = \delta(q_0,a) \cup \delta(q_1,a) = \{q_0,q_1\} \cup \phi = \{q_0,q_1\} = [q_0,q_1]$$
$$\delta'([q_0,q_1],b) = \delta(\{q_0,q_1\},b) = \delta(q_0,b) \cup \delta(q_1,b) = \{q_1\} \cup \{q_1\} = \{q_1\} = [q_1]$$

$$4.\delta'(\phi,a) = \phi$$
$$\delta'(\phi,b) = \phi$$

Equivalence of NFA and DFA:

Example-1: Solution cont.2

Q/Σ	Input	
	a	b
$\rightarrow[q_0]$	$[q_0, q_1]$	$[q_1]$
$[q_1]$	ϕ	$[q_1]$
$[q_0, q_1]$	$[q_0, q_1]$	$[q_1]$
ϕ	ϕ	ϕ

Table: Transition Table of equivalent DFA

Equivalence of NFA and DFA:

Example-1: Solution cont.3

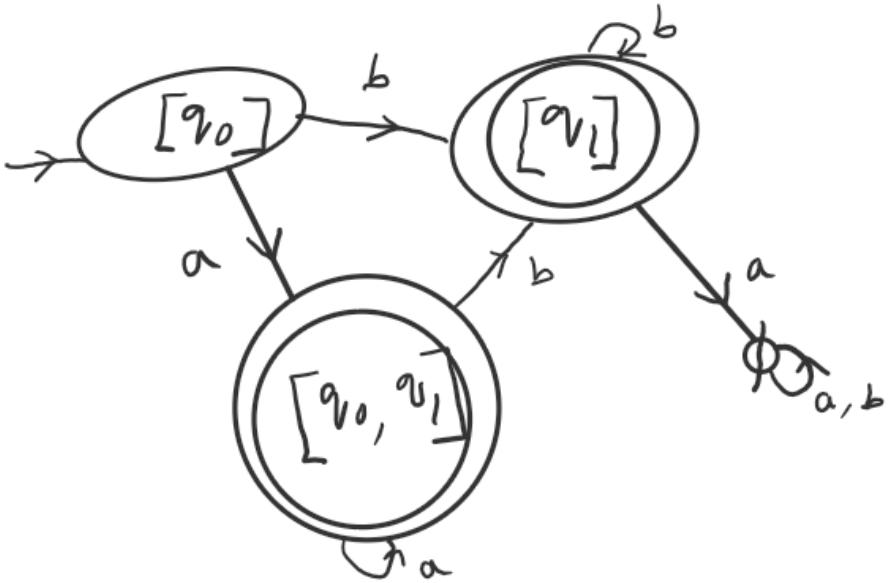
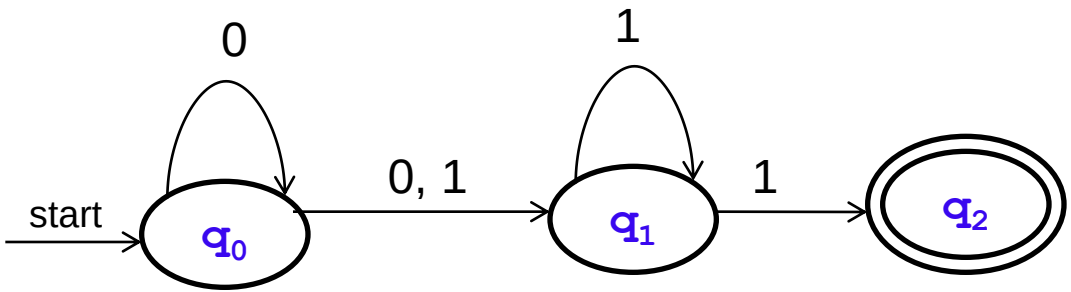


Fig: Transition Diagram of equivalent DFA

Equivalence of NFA and DFA:

Example-2:

Construct an equivalent DFA for the given NFA.



Equivalence of NFA and DFA:

Example-2: Solution (Method-2)

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

where

$$Q' = 2^Q = 2^3 = 8 \ \ states$$

$$Q' = \{\phi, [q_0], [q_1], [q_2], [q_0, q_1], [q_0, q_2], [q_1, q_2], [q_0, q_1, q_2]\}$$

$$\Sigma = \{0, 1\}$$

$$q_0 = [q_0]$$

$$F' = \{[q_2], [q_0, q_2], [q_1, q_2], [q_0, q_1, q_2]\}$$

Equivalence of NFA and DFA:

Example-2: Solution (Method-2) cont.

$$1.\delta'([q_0],0) = \delta(\{q_0\},0) = \delta(q_0,0) = \{q_0,q_1\} = [q_0,q_1](new\_state)$$
$$\delta'([q_0],1) = \delta(\{q_0\},1) = \delta(q_0,1) = \{q_1\} = [q_1](new\_state)$$

$$2.\delta'([q_0,q_1],0) = \delta(\{q_0,q_1\},0) = \delta(q_0,0) \cup \delta(q_1,0) = \{q_0,q_1\} \cup \phi = \{q_0,q_1\} = [q_0,q_1]$$
$$\delta'([q_0,q_1],1) = \delta(\{q_0,q_1\},1) = \delta(q_0,1) \cup \delta(q_1,1) = \{q_1\} \cup \{q_1,q_2\} = \{q_1,q_2\} = [q_1,q_2](new\_state)$$

$$3.\delta'([q_1],0) = \delta(\{q_1\},0) = \delta(q_1,0) = \phi$$
$$\delta'([q_1],1) = \delta(\{q_1\},1) = \delta(q_1,1) = \{q_1,q_2\} = [q_1,q_2]$$

$$4.\delta'([q_1,q_2],0) = \delta(\{q_1,q_2\},0) = \delta(q_1,0) \cup \delta(q_2,0) = \phi \cup \phi = \phi$$
$$\delta'([q_1,q_2],1) = \delta(\{q_1,q_2\},1) = \delta(q_1,1) \cup \delta(q_2,1) = \{q_1,q_2\} \cup \phi = \{q_1,q_2\} = [q_1,q_2]$$

$$5.\delta'(\phi,0) = \phi$$
$$\delta'(\phi,1) = \phi$$

Equivalence of NFA and DFA:

Example-2: Solution (Method-2) cont.

Q/Σ	Input	
	0	1
→[q_0]	[q_0,q_1]	[q_1]
[q_1]	ϕ	[q_1,q_2]
[q_0,q_1]	[q_0,q_1]	[q_1,q_2]
[q_1,q_2]	ϕ	[q_1,q_2]
ϕ	ϕ	ϕ

Table: Transition Table of equivalent DFA

Equivalence of NFA and DFA:

Example-2: Solution (Method-2) cont.

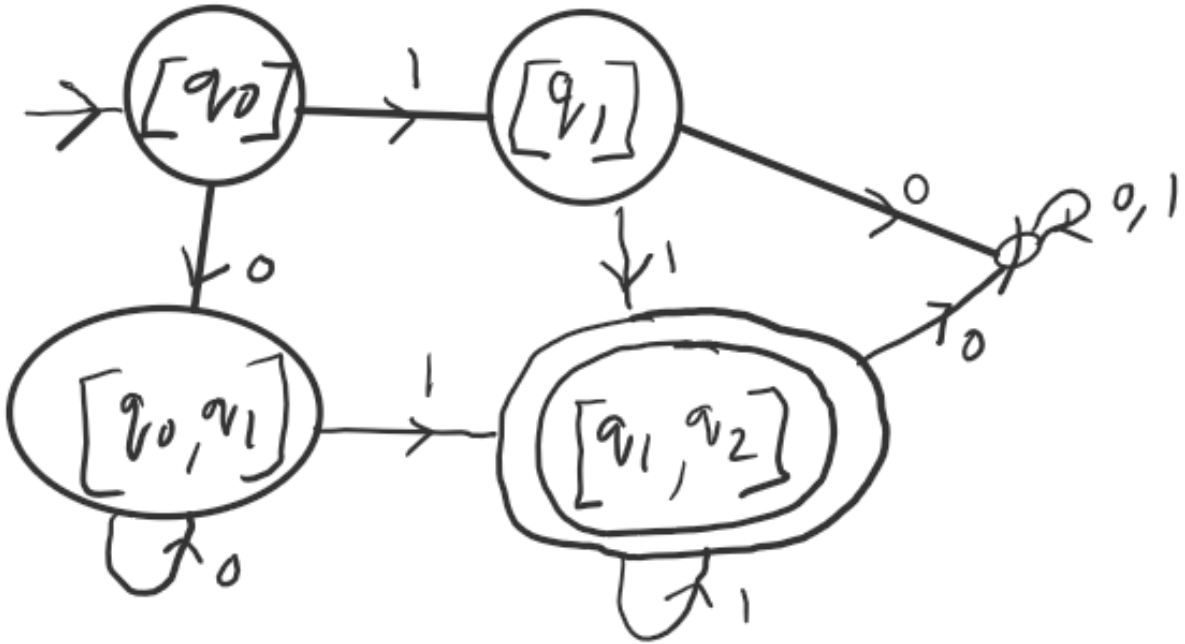
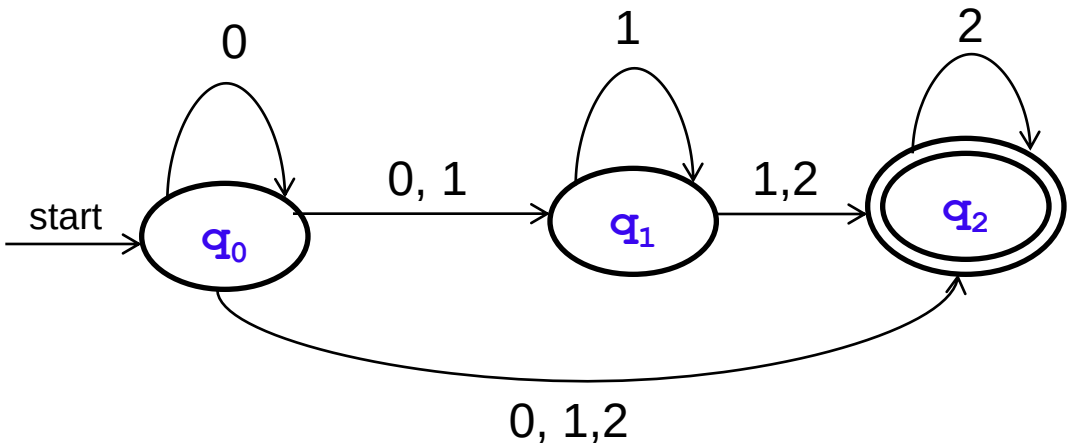


Fig: Transition Diagram of equivalent DFA

Equivalence of NFA and DFA:

Practice problem-1:

Construct an equivalent DFA for the given NFA.



Applications of FA:

- Design of digital circuits
- Compiler construction
- String matching
- String processing
- Text search
- Software design
- Other applications such as artificial intelligence, knowledge engineering, game theory and games, computer graphics, linguistics etc.

Applications of FA:

Design of Digital Circuits:

- The **FA** is used in designing and checking the behaviour of **digital circuits** using **software**.
- The **FA** is useful in **hardware design** such as circuit verification, in design of automatic traffic signals etc.

Applications of FA:

Compiler Construction:

- **FA** used in design of **lexical analyzer** (the first phase of **compiler**) which takes input string and divides into **tokens** such as identifiers, keywords, constants, operators, punctuations etc.

Applications of FA:

String matching:

- In designing of a software, for identifying the words, phrases and other **patterns** in large bodies of text(collection of web pages).

Applications of FA:

String processing:

- To write **software** for processing the natural language (ex., speech processing).
- Large natural vocabularies can be described which includes the applications such as spelling checkers and advisers, multi-language definitions, identifying the documents etc.

Applications of FA:

Software design:

- While developing the software to verify the systems having the finite number of states.

Ex: Communication protocols in computer networks.

An Application: Text Search

1. Finding Strings in Text
2. Nondeterministic Finite Automata for Text Search

An Application: Text Search

1. Finding Strings in Text

A common problem in the age of the Web and other online text repositories is the following:

- Given a set of words, find all documents that contain one (or all) of those words. A search engine is a popular example of this process.
- The search engine uses a particular technology, called inverted indexes, where for each word appearing on the Web (there are 100,000,000 different words), a list of all the places where that word occurs is stored.
- Machines with very large amounts of main memory keep the most common of these lists available, allowing many people to search for documents at once.

An Application: Text Search

1. Finding Strings in Text

- Inverted index techniques do not make use of finite automata, but they also take very large amounts of time for crawlers to copy the Web and set up the indexes.
- There are a number of related applications that are unsuited for inverted indexes, but are good applications for automaton-based techniques.
- The characteristics that make an application suitable for searches that use automata are:
  - The repository on which the search is conducted is rapidly changing
  - The documents to be searched cannot be cataloged

An Application: Text Search

2. Nondeterministic Finite Automata (NFA) for Text Search

- Suppose we are given a set of words, which we shall call the keywords, and we want to find occurrences of any of these words.
- In applications such as these, a useful way to proceed is to design a nondeterministic finite automaton (NFA), which signals, by entering an accepting state, that it has seen one of the keywords.
- The text of a document is fed, one character at a time to this NFA, which then recognizes occurrences of the keywords in this text.

An Application: Text Search

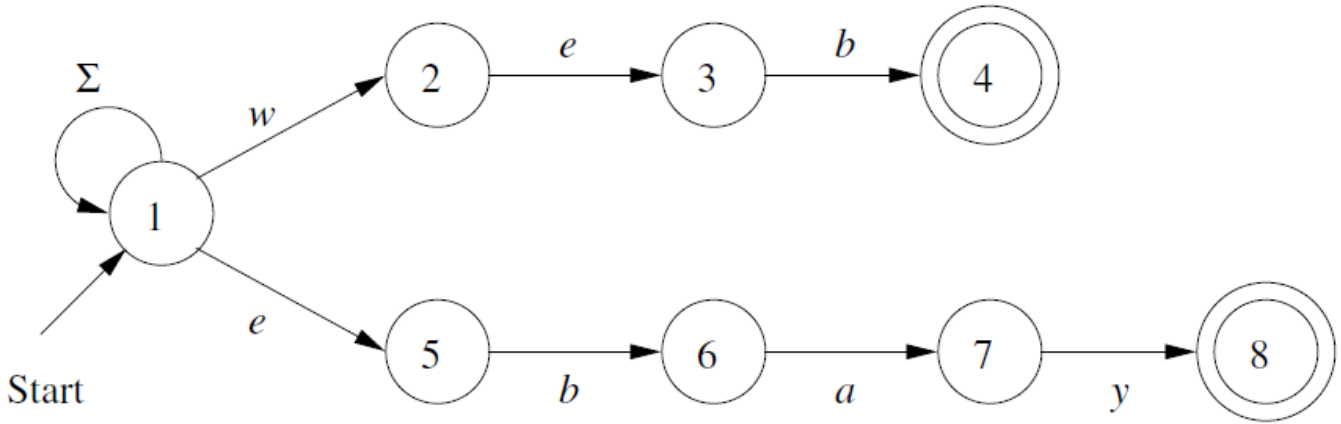
2. Nondeterministic Finite Automata (NFA) for Text Search

- There is a simple form to an NFA that recognizes a set of keywords
  - There is a start state with a transition to itself on every input symbol, e.g., for every printable ASCII character if we are examining text. Intuitively (without conscious reasoning), the start state represents a "guess" that we have not yet begun to see one of the keywords, even if we have seen some letters of one of these words.
  - For each keyword $a_1a_2\ldots a_k$, there are k states, say $q_1, q_2, \ldots, q_k$. There is a transition from the start state to q_1 on symbol a_1, a transition from q_1 to q_2 on symbol a_2, and so on. The state q_k is an accepting state and indicates that the keyword $a_1a_2\ldots a_k$ has been found.

An Application: Text Search

2. Nondeterministic Finite Automata (NFA) for Text Search

- There is a simple form to an NFA that recognizes a set of keywords



Example: An NFA that searches for the words web and ebay

Lecture- 06
CONVERSION OF NFA TO DFA
Summary

- Equivalence of DFA and NFA
- Applications of FA
- An Application: Text Search