Compiler Design

Dr. Sahar kamal

Lexical Analysis

Lecture 4

Compilers *Principles, Techniques, & Tools*

Second Edition

How to convert NFA with epsilon to DFA

In this method, we first convert Non-deterministic finite automata (NFA) with ε to NFA without ε.

Then, NFA without ε- can be converted to its equivalent Deterministic Finite Automata (DFA).

❑Method for conversion

The method for converting the NFA with ε to DFA is explained below −

Step 1 − Consider M={Q, Σ, δ,q0,F) is NFA with ε. We have to convert this NFA with ε to equivalent DFA denoted by

- M0=(Q0,Σ, δ0,q0,F0)
- Then obtain, ε-closure(q0) ={p1,p2,p3,……pn}
- then [p1,p2,p2,….pn] becomes a start state of DFA
- now[p1,p2,p3,….pn] ∈ Q0

Step 2 − We will obtain δ transition on [p1,p2,p3,…pn] for each input.

- δ 0([p1,p2,p3,..pn],a) = ε-closure(δ(p1,a) U δ(p2,a2)U……………… δ(pn,a))
- = U (i=1 to n) ε-closure d(pi,a)
- Where a is input ∈Σ

Step 3 − The state obtained [p1,p2,p3,…pn] ∈ Q0 .

- The states containing final state in pi is a final state in DFA

The steps involved in the conversion of NFA to DFA are,

1. Transform the NFA with Epsilon transitions to NFA without epsilon transitions.

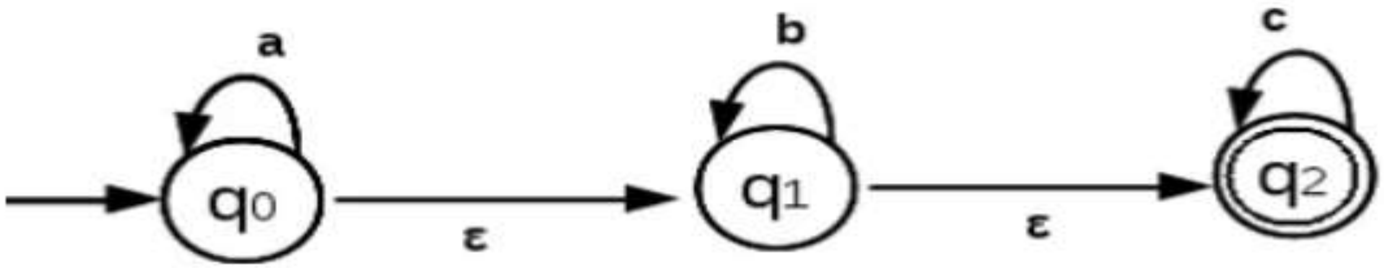
2. convert the resulting NFA to DFA.

These steps are exaplained in detail as follows:

1. Transform the NFA with ε transitions to NFA without ε transitions.

Example 1:

Consider the following NFA with epsilon transitions:



The above NFA has states, q_0, q_1, q_2. The start state is q_0. The final state is q_2.

Step a: Find the Epsion closure of all states.

ECLOSE(q0) = $\{q_0, q_1, q_2\}$

ECLOSE(q_1) = $\{ q_1, q_2\}$

ECLOSE(q_2) = $\{q_2\}$

The states of the new NFA will be $\{q_0, q_1, q_2\}, \{q_1, q_2\}, \{q_2\}$.

The transition table for the new NFA is,

Current state	Input Symbol a	b	c
$\longrightarrow *\{q_0, q_1, q_2\}$	$\{q_0, q_1, q_2\}$	$\{q_1, q_2\}$	$\{q_2\}$
$*\{q_1, q_2\}$	ϕ	$\{q_1, q_2\}$	$\{q_2\}$
$*\{q_2\}$	ϕ	ϕ	$\{q_2\}$

Let us say,

$\{q_0, q_1, q_2\}$ as q_x

$\{q_1, q_2\}$ as q_y

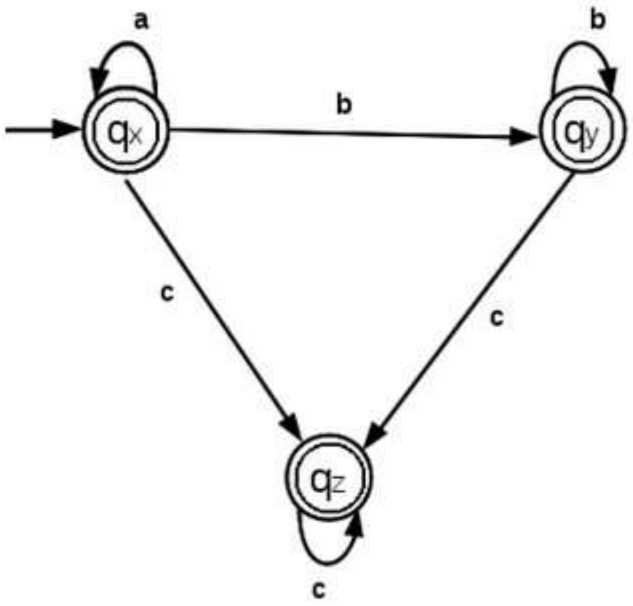
$\{q_2\}$ as q_z

Then the transition table will become,

Current state	Input Symbol a	b	c
$\longrightarrow *q_x$	q_x	q_y	q_z
$*q_y$	ϕ	q_y	q_z
$*q_z$	ϕ	ϕ	q_z

NFA to DFA ConversionNFA to DFA Conversion

The transition diagram for the new NFA is,

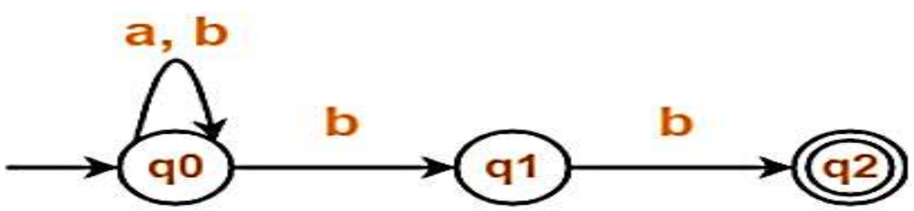


Subset Construction Algorithm

- **Converting NFA to DFA**

Problem-01:

Convert the following Non-Deterministic Finite Automata (NFA) to Deterministic Finite Automata (DFA)-



Transition table for the given Non-Deterministic Finite Automata (NFA) is-

State / Alphabet	a	b
→q0	q0	q0, q1
q1	–	*q2
*q2	–	–

Converting NFA to DFA

Step-02 Add transitions of start state q0 to the transition table T'.

State / Alphabet	a	b
→q0	q0	{q0, q1}

Step-03:

New state present in state Q' is {q0, q1}.
Add transitions for set of states {q0, q1} to the transition table T'

State / Alphabet	a	b
→q0	q0	{q0, q1}
{q0, q1}	q0	{q0, q1, q2}

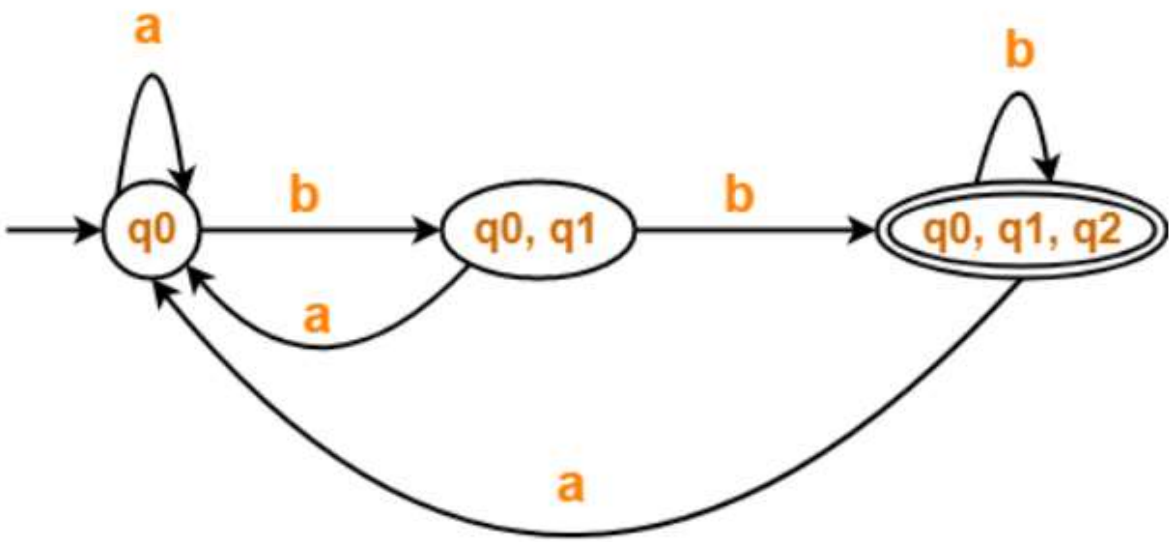
Converting NFA to DFA

State / Alphabet	a	b
→q0	q0	{q0, q1}
{q0, q1}	q0	{q0, q1, q2}
{q0, q1, q2}	q0	{q0, q1, q2

State / Alphabet	a	b
→q0	q0	{q0, q1}
{q0, q1}	q0	*{q0, q1, q2}
*{q0, q1, q2}	q0	*{q0, q1, q2

Converting NFA to DFA

Now, Deterministic Finite Automata (DFA) may be drawn as-

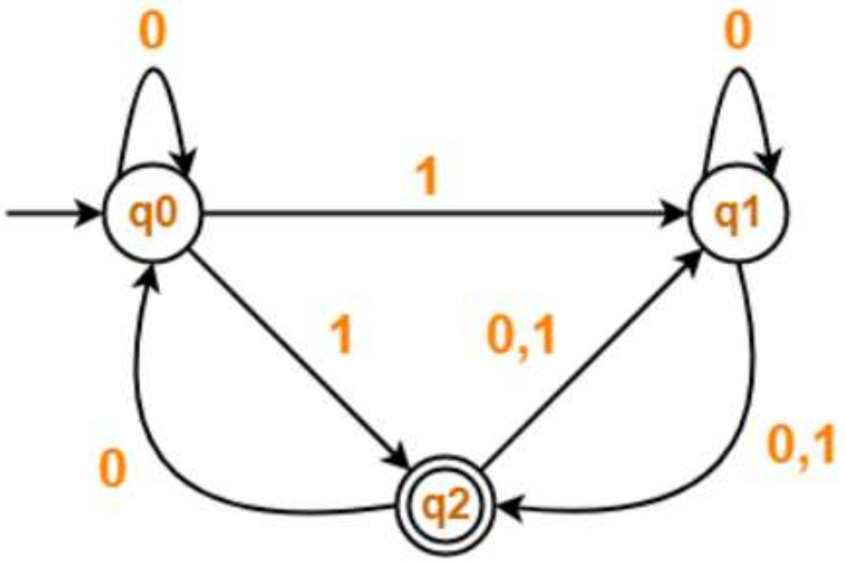


Deterministic Finite Automata (DFA)

Converting NFA to DFA

Problem-02:

Convert the following Non-Deterministic Finite Automata (NFA) to Deterministic Finite Automata (DFA)-



Converting NFA to DFA

- Transition table for the given Non-Deterministic Finite Automata (NFA) is

State / Alphabet	0	1
→q0	q0	q1, *q2
q1	q1, *q2	*q2
*q2	q0, q1	q1

Step-02:

Add transitions of start state q0 to the transition table T’.

State / Alphabet	0	1
→q0	q0	{q1, q2}

Converting NFA to DFA

- **Step-03:**
- New state present in state Q’ is {q1, q2}.
- Add transitions for set of states {q1, q2} to the transition table T’.

State / Alphabet	0	1
→q0	q0	{q1, q2}
{q1, q2}	{q0, q1, q2}	{q1, q2}

Step-04:

New state present in state Q’ is {q0, q1, q2}.

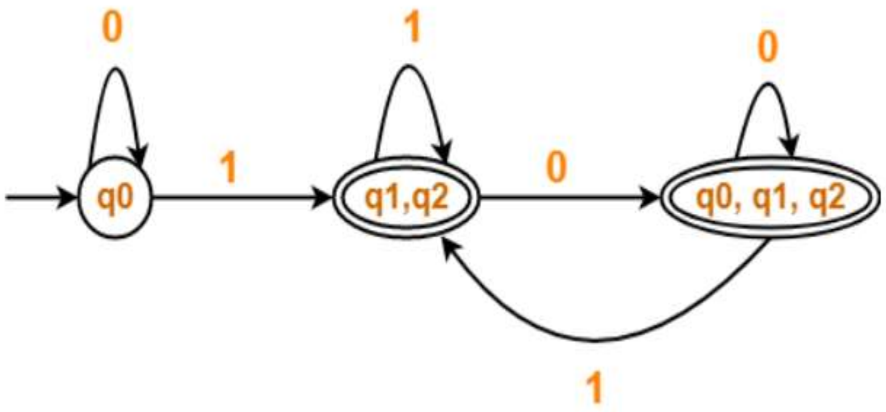
Add transitions for set of states {q0, q1, q2} to the transition table

State / Alphabet	0	1
→q0	q0	{q1, q2}
{q1, q2}	{q0, q1, q2}	{q1, q2}
{q0, q1, q2}	{q0, q1, q2}	{q1, q2}

Converting NFA to DFA

- **Step-05:**
- Since no new states are left to be added in the transition table T’, so we stop.
- States containing q2 as its component are treated as final states of the DFA.
- Finally, Transition table for Deterministic Finite Automata (DFA) is-

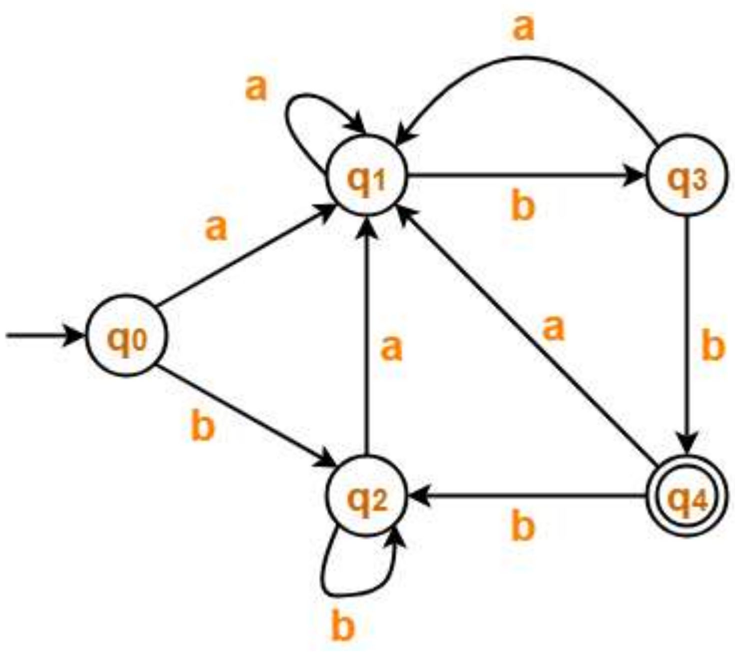
State / Alphabet	0	1
→q0	q0	*{q1, q2}
*{q1, q2}	*{q0, q1, q2}	*{q1, q2}
*{q0, q1, q2}	*{q0, q1, q2}	*{q1, q2}



Deterministic Finite Automata (DFA)

Minimization of DFA

Minimize the given DFA-



Solution-

Step-01:

The given DFA contains no dead states and inaccessible states.

Step-02: Draw a state transition table

	a	b
→q0	q1	q2
q1	q1	q3
q2	q1	q2
q3	q1	*q4
*q4	q1	q2

Step-03:

Now using Equivalence Theorem, we have-

$P_0 = \{ q_0 , q_1 , q_2 , q_3 \} \{ q_4 \}$

$P_1 = \{ q_0 , q_1 , q_2 \} \{ q_3 \} \{ q_4 \}$

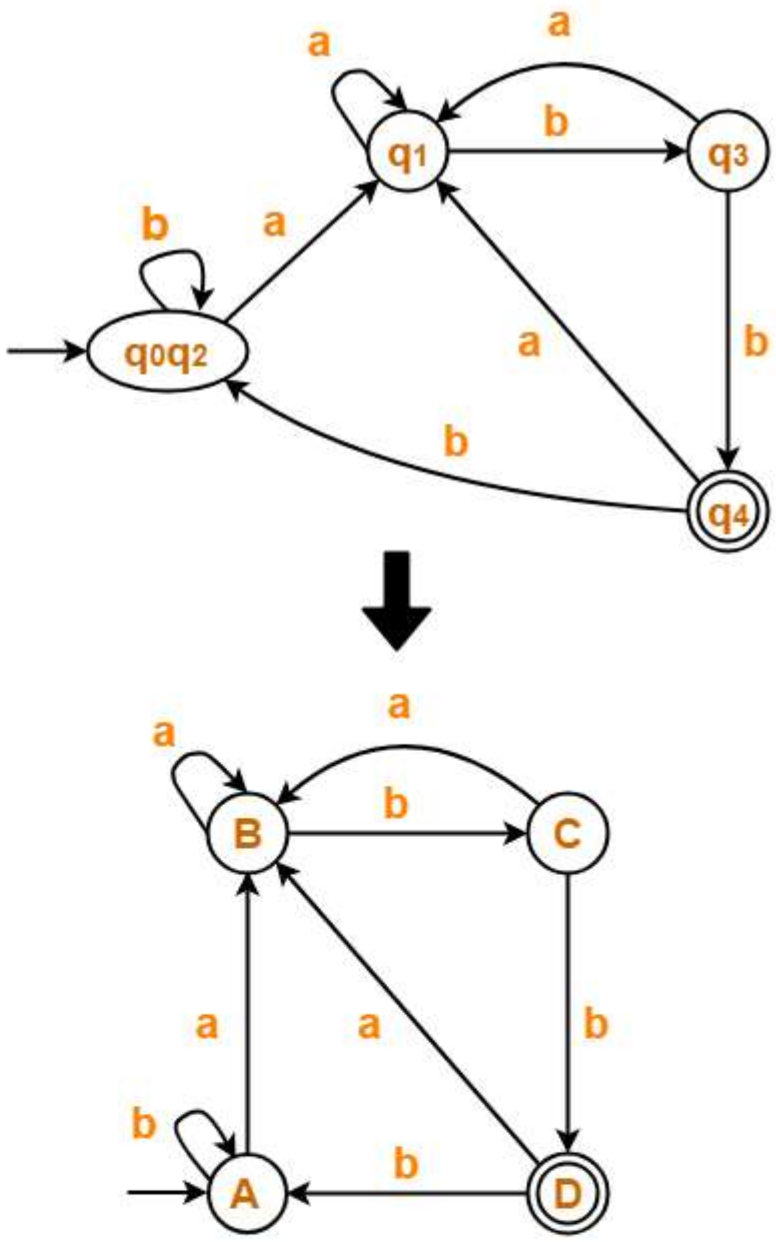
$P_2 = \{ q_0 , q_2 \} \{ q_1 \} \{ q_3 \} \{ q_4 \}$

$P_3 = \{ q_0 , q_2 \} \{ q_1 \} \{ q_3 \} \{ q_4 \}$

Since $P_3 = P_2$, so we stop.

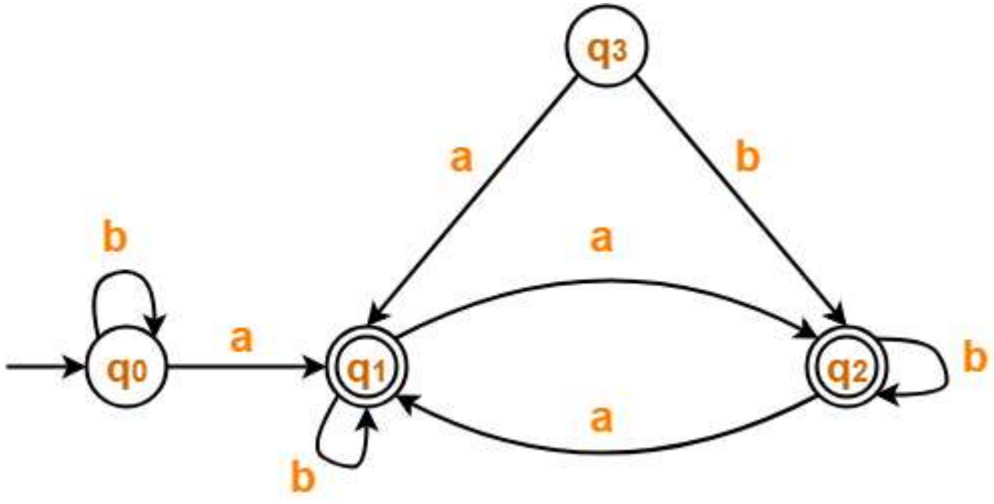
From P_3, we infer that states q_0 and q_2 are equivalent and can be merged together.

So, Our minimal DFA is-



Minimal DFA

EX: 2 Minimize the given DFA-

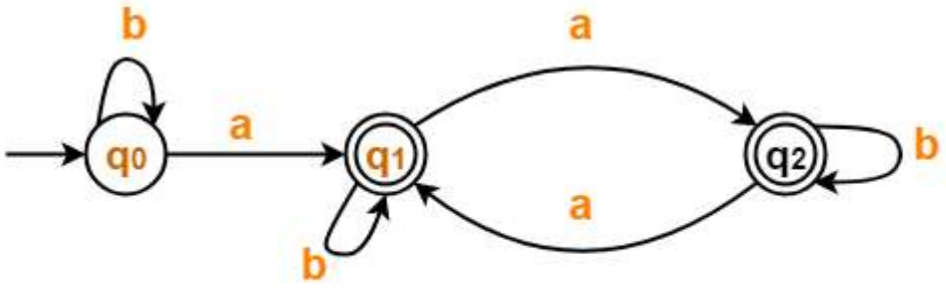


Solution-

Step-01:

- State q_3 is inaccessible from the initial state.
- So, we eliminate it and its associated edges from the DFA.

The resulting DFA is-



Step-02:

Draw a state transition table-

	a	b
→q0	*q1	q0
*q1	*q2	*q1
*q2	*q1	*q2

Step-03:

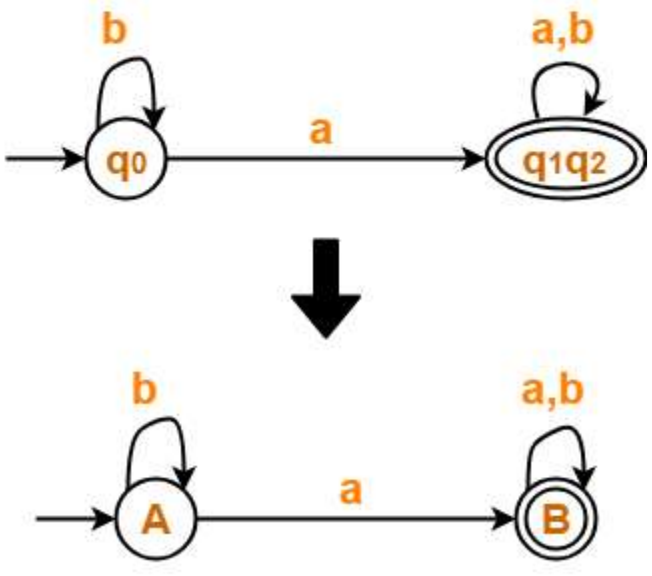
Now using Equivalence Theorem, we have-

$P_0 = \{ q_0 \} \{ q_1 , q_2 \}$

$P_1 = \{ q_0 \} \{ q_1 , q_2 \}$

Since $P_1 = P_0$, so we stop.

From P_1, we infer that states q_1 and q_2 are equivalent and can be merged together.So, Our minimal DFA is-



Minimal DFA