CS 181 (Introduction to Formal Languages and Automata Theory)

April 5, 2022

1 Deterministic finite automata (DFAs)

1.1 Basic notions

Definition 1.1.1

An **alphabet** is any finite set of symbols.

Example 1.1.2. Binary alphabet: $\{\mathtt{0},\mathtt{1}\}$

Example 1.1.3. English alphabet: $\{\mathtt{a},\mathtt{b},\ldots,\mathtt{c}\}$

Definition 1.1.4

A **string** is any finite sequence of symbols from a given alphabet.

Example 1.1.5. `001010110101`

Example 1.1.6. `abracadabra`

Example 1.1.7. ε (empty string)

Definition 1.1.8

A **language** is a set of strings over a given alphabet.

Example 1.1.9. $\varnothing$ (empty language)

Example 1.1.10. $\{\varepsilon\}$

Example 1.1.11. $\{\mathtt{acclaim},\mathtt{aim},\mathtt{brim},\ldots\}$

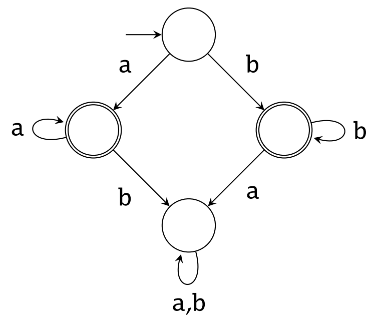
Example 1.1.12. $\{\mathtt{0},\mathtt{1},\mathtt{00},\mathtt{11},\ldots\}$

Definition 1.1.13

A **computational device** is a mechanism that inputs a string and either accepts or rejects it.

1.2 Deterministic finite automata

- Choose an alphabet: $\{\mathtt{a},\mathtt{b}\}$.
- Draw states.
- Choose start state and accept states.
- Draw transitions (out of every state on every symbol).



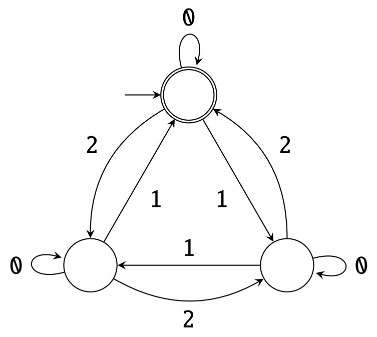
Input	Output
ε	reject
ab	reject
aaa	accept
bb	accept

In words, this machine accepts nonempty strings of all a's or all b's.

Definition 1.2.1

The **language** of a DFA is the set of all strings it accepts.

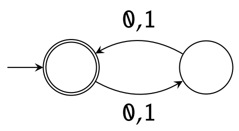
Example 1.2.2.



Input	Output
00...0	accept
12	accept
111	accept
20	reject
1	reject

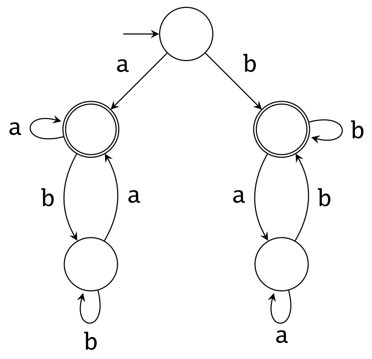
Alphabet: $\{0, 1, 2\}$, language: $\{w : 3 \mid \sum w_i\}$

Example 1.2.3.



Alphabet: $\{0, 1\}$, language: $\{w : 2 \mid |w|\}$

Example 1.2.4.

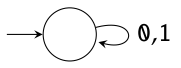


Alphabet: $\{a, b\}$, language: $\left\{w : w \neq \varepsilon \wedge w_1 = w_{|w|}\right\}$

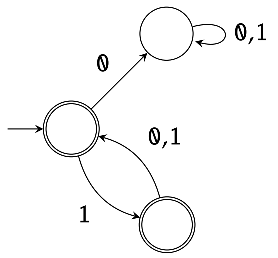
1.3 Designing DFAs

We will be using the binary alphabet $\{\texttt{0}, \texttt{1}\}$.

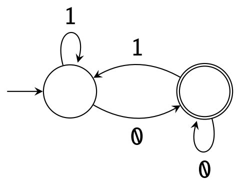
Example 1.3.1. Language: $\varnothing$



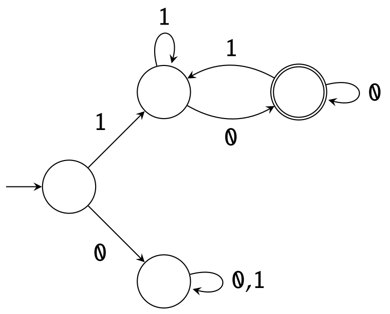
Example 1.3.2. Language: $\{w : \text{every odd position is a } \texttt{1}\}$



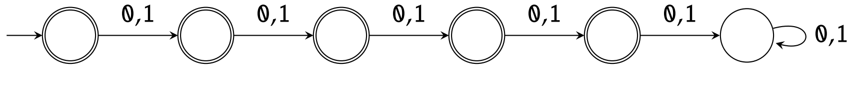
Example 1.3.3. Language: $\{w : w \text{ ends in } \texttt{0}\}$



Example 1.3.4. Language: $\{w : w \text{ begins with } \texttt{1}, \text{ ends with } \texttt{0}\}$



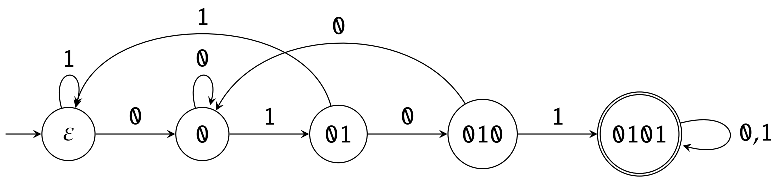
Example 1.3.5. Language: $\{w : |w| \leq 4\}$



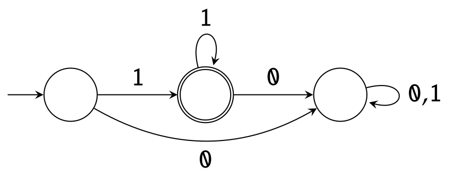
Example 1.3.6. Language: $\{w : 1000 \mid |w|\}$

In words, each state represents a remainder modulo 1000, and only the 0 state is accepting.

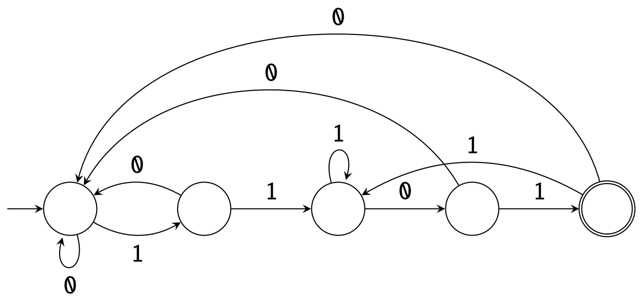
Example 1.3.7. Language: $\{w : w \text{ contains } \texttt{0101} \text{ as a substring}\}$



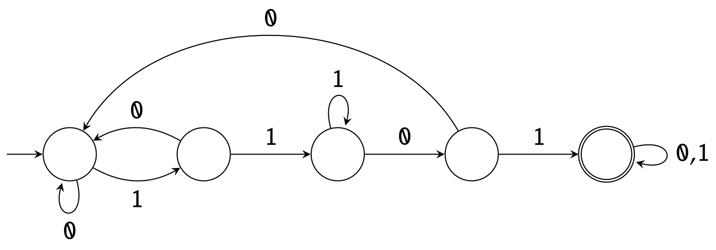
Example 1.3.8 (Week 1 Discussion). $L = \{w : |w| > 0 \wedge w \text{ contains only 1s}\}$



Example 1.3.9 (Week 1 Discussion). $L = \{w : w \text{ ends in } \texttt{1101}\}$



Example 1.3.10 (Week 1 Discussion). $L = \{w : w \text{ contains } \texttt{1101}\}$



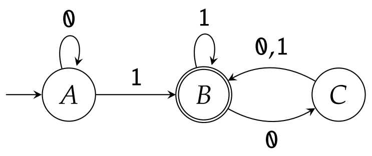
1.4 Formal definitions

Definition 1.4.1

A DFA is a tuple $(Q, \Sigma, \delta, q_0, F)$ where

- Q = set of states,
- Σ = alphabet,
- δ = transition function ($\delta\colon Q \times \Sigma \to Q$),
- q_0 = start state ($q_0 \in Q$), and
- F = set of accept states ("favorable"? states, $F \subseteq Q$).

Example 1.4.2.

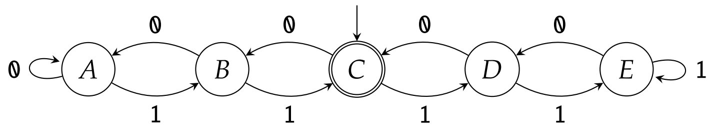


Formal description: $(\{A, B, C\}, \{0, 1\}, \delta, A, \{B\})$ where δ is defined by the table

	0	1
A	A	B
B	C	B
C	B	B.

Example 1.4.3. Formal description: $(\{A, B, C, D, E\}, \{0, 1\}, \delta, C, \{C\})$ where δ is defined by the table

	0	1
A	A	B
B	A	C
C	B	D
D	C	E
E	D	E.



Example 1.4.4. Formal description for Example 1.3.6: $(\{0, 1, 2, \ldots, 999\}, \{0, 1\}, \delta, 0, \{0\})$ where $\delta(q, \sigma) = (q + 1) \mod 1000$.

Definition 1.4.5

DFA $(Q, \Sigma, \delta, q_0, F)$ **accepts** a string $w = w_1 w_2 \dots w_n$ iff

$$\delta(\cdots \delta(\delta(q_0, w_1), w_2) \cdots, w_n) \in F.$$

Definition 1.4.6

DFA D **recognizes** the language $\mathcal{L}$ iff

$$\mathcal{L} = \left\{ w : D \text{ accepts } w \right\}.$$

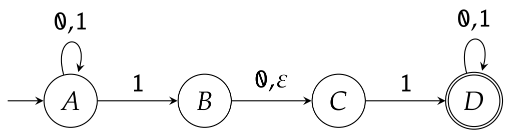
Note.

- Every DFA recognizes exactly 1 language.
- A language has either 0 or ∞ DFAs recognizing it.

2 Nondeterminism

2.1 Basic notions

Example 2.1.1.



- Choose an alphabet: $\{\mathtt{0}, \mathtt{1}\}$.
- Draw states.
- Choose start state and accept states. The steps so far are the same as those of a DFA.
- Draw transitions. A state may have any number of transitions on a given symbol. A state may also have transitions on ε.

Definition 2.1.2

An NFA **accepts** w iff there is *at least* one path to an accept state.

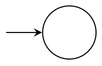
Example 2.1.3. Output table for Example 2.1.1:

Input	Accepting path	Output
ε	-	reject
0	-	reject
1	-	reject
010110	$AABCDDD$	accept
010	-	reject
11	$ABCD$	accept

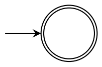
Language: all strings containing 101 or 11

2.2 Using shortcuts

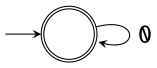
Example 2.2.1. Language: $\varnothing$



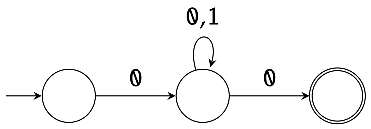
Example 2.2.2. Language: $\{\varepsilon\}$



Example 2.2.3. Language: $\{w : w \text{ doesn't contain } 1\}$

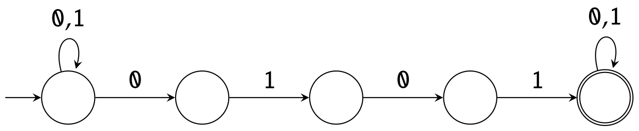


Example 2.2.4. Language: $\{w : |w| \geq 2 \text{ and } w \text{ starts and ends with } 0\}$

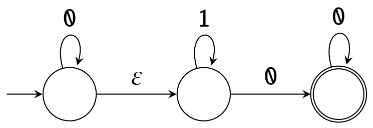


2.3 Pattern matching

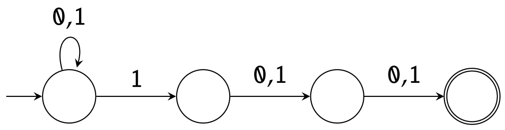
Example 2.3.1. Language: $\{w : \text{conatins } 0101\}$



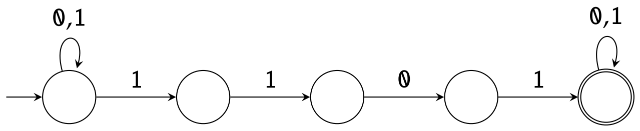
Example 2.3.2. Language: $\{w : w = \underbrace{\texttt{00...0}}_{\geq 0}\underbrace{\texttt{11...1}}_{\geq 0}\underbrace{\texttt{00...0}}_{\geq 1}\}$



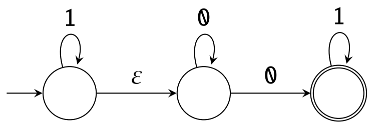
Example 2.3.3. Language: $\{w : w \text{ has a } \texttt{1} \text{ in the 3rd position from the end}\}$



Example 2.3.4 (Week 1 Discussion). $L = \{w : w \text{ contains } \texttt{1101}\}$

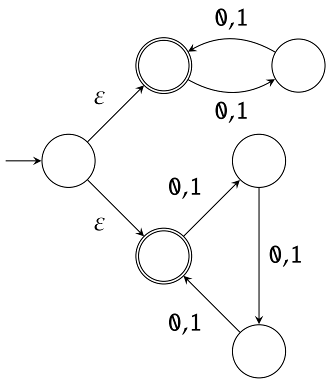


Example 2.3.5 (Week 1 Discussion). $L = \{w : w = \underbrace{\texttt{11...1}}_{\geq 0}\underbrace{\texttt{00...0}}_{\geq 1}\underbrace{\texttt{11...1}}_{\geq 0}\}$

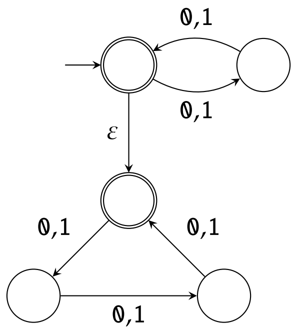


2.4 Alternatives

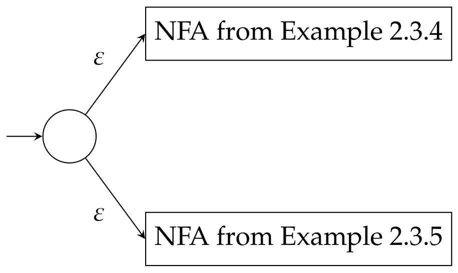
Example 2.4.1. Language: $\{w : 2 \mid |w| \vee 3 \mid |w|\}$



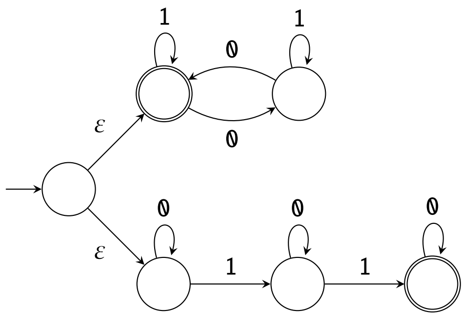
Note that the following is not valid due to side effects:



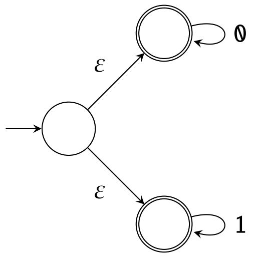
Example 2.4.2 (Week 1 Discussion). $L = \{w : w \text{ contains } \texttt{1101} \lor w = \underbrace{\texttt{11...1}}_{\geq 0}\underbrace{\texttt{00...0}}_{\geq 1}\underbrace{\texttt{11...1}}_{\geq 0}\}$



Example 2.4.3. Language: $\{w : w \text{ contains an even number of } \texttt{0}\text{s, or exactly two 1s}\}$



Example 2.4.4. Language: $\{w : w \text{ does not contain both } \texttt{0} \text{ and } \texttt{1}\}$



2.5 Formal definitions

Definition 2.5.1

An NFA is a tuple $(Q, \Sigma, \delta q_0, F)$ where

- Q = set of states,
- Σ = alphabet,
- δ = transition function ($\delta\colon Q \times (\Sigma \cup \{\varepsilon\}) \to \mathcal{P}(Q)$),
- q_0 = start state ($q_0 \in Q$), and
- F = set of accept states ($F \subseteq Q$).

Example 2.5.2. Formal description for NFA from Example 2.1.1:

$$(\{A, B, C, D\}, \{\mathtt{0}, 1\}, \delta, A, \{D\})$$

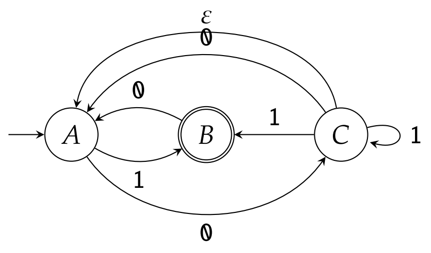
where δ is defined by the table

	$\mathtt{0}$	1	ε
A	$\{A\}$	$\{A, B\}$	$\varnothing$
B	$\{C\}$	$\varnothing$	$\{C\}$
C	$\varnothing$	$\{D\}$	$\varnothing$
D	$\{D\}$	$\{D\}$	$\varnothing$.

Note that as pictures are not precise, we had to make an assumption on the alphabet of the NFA. Also note that although adding transition from A to A on ε does not change the language, it does not match the drawing and therefore represents a different NFA. Make sure to transcribe the given NFA.

Example 2.5.3. $(\{A, B, C\}, \{\mathtt{0}, 1\}, \delta, A, \{B\})$ where δ is defined by the table

	$\mathtt{0}$	1	ε
A	$\{C\}$	$\{B\}$	$\varnothing$
B	$\{A\}$	$\varnothing$	$\varnothing$
C	$\{B\}$	$\{B, C\}$	$\{A\}$.



Definition 2.5.4

NFA $(Q, \Sigma, \delta, q_0, F)$ **accepts** a string w iff

$$\begin{aligned}\exists\, q_1, q_2, \ldots, q_m \in Q\, \exists\, \sigma_0, \sigma_1, \ldots, \sigma_{m-1} \in \Sigma \cup \{\varepsilon\} :&\\ q_1 \in \delta(q_0, \delta_0) \wedge q_2 \in \delta(q_1, \sigma) \wedge \cdots&\\ \wedge\, q_m \in \delta(q_{m-1}, \sigma_{m-1}) \wedge q_m \in F&\\ \wedge\, \sigma_0\sigma_1 \ldots \sigma_{m-1} = w,&\end{aligned}$$

or, in words, an accept state is reachable from q_0 via some path on input w.

Definition 2.5.5

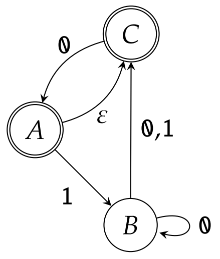
NFA N **recognizes** the language $\mathcal{L}$ iff

$$\mathcal{L} = \left\{w : N \text{ accepts } w\right\}.$$

3 Equivalence of DFAs and NFAs

3.1 Example

Does this NFA accept the string `0110`?



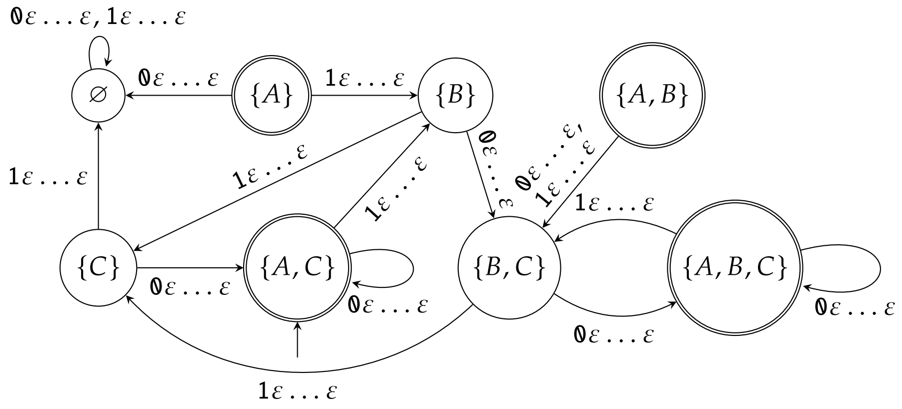
Recall that the NFA accepts `0110` iff an accept state is reachable from the start state via some path $\underbrace{\varepsilon \ldots \varepsilon}_{\geq 0} \mathtt{0} \underbrace{\varepsilon \ldots \varepsilon}_{\geq 0} \mathtt{1} \underbrace{\varepsilon \ldots \varepsilon}_{\geq 0} \mathtt{1} \underbrace{\varepsilon \ldots \varepsilon}_{\geq 0} \mathtt{0} \underbrace{\varepsilon \ldots \varepsilon}_{\geq 0}$.

Path	Possible end states
$\varepsilon \ldots \varepsilon$	A, C
$\varepsilon \ldots \varepsilon \mathtt{0} \varepsilon \ldots \varepsilon$	A, C
$\varepsilon \ldots \varepsilon \mathtt{0} \varepsilon \ldots \varepsilon \mathtt{1} \varepsilon \ldots \varepsilon$	B
$\varepsilon \ldots \varepsilon \mathtt{0} \varepsilon \ldots \varepsilon \mathtt{1} \varepsilon \ldots \varepsilon \mathtt{1} \varepsilon \ldots \varepsilon$	C
$\varepsilon \ldots \varepsilon \mathtt{0} \varepsilon \ldots \varepsilon \mathtt{1} \varepsilon \ldots \varepsilon \mathtt{1} \varepsilon \ldots \varepsilon \mathtt{0} \varepsilon \ldots \varepsilon$	A, C

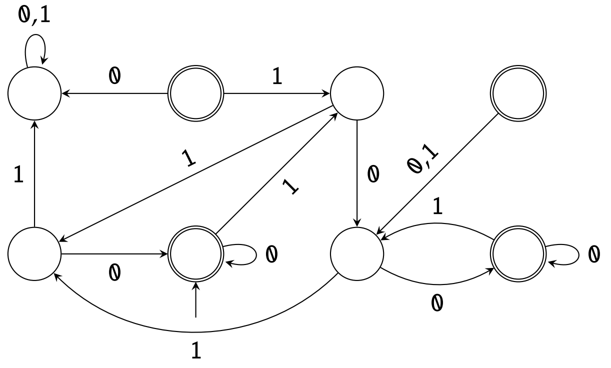
Since C is an accept state, the NFA accepts the string `0110`.

Can we convert this to a DFA? Yes.

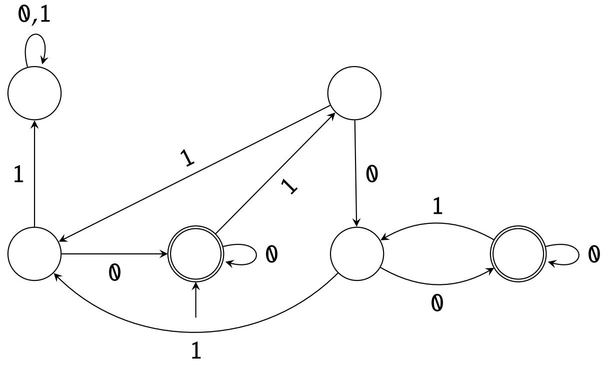
Step 1 Use subsets of the states of the NFA as the states of the DFA. Accept states are subsets that contain accept states of the NFA. The start state is the subset of all reachable states from the start state of the NFA via ε. For subsets containing more than one element, the transition is the union of all transitions for each individual element.



Step 2 Clean up.



Step 3 Optionally drop unreachable states.



3.2 General theorem

Theorem 3.2.1

Every NFA N can be converted to a DFA D that recognizes the same language.

Proof. Given NFA $N = (Q, \Sigma, \delta, q_0, F)$ define DFA $D = (\mathcal{P}(Q), \Sigma, \Delta, S_0, \mathscr{F})$ where

- $S_0 = \{q \in Q : q \text{ is reachable from } q_0 \text{ via a path } \underbrace{\varepsilon \ldots \varepsilon}_{\geq 0}\}$,
- $\Delta(S, \sigma) = \{q \in Q : q \text{ is reachable from a state in } S \text{ via a path } \delta\, \underbrace{\varepsilon \ldots \varepsilon}_{\geq 0}\}$, and
- $\mathscr{F} = \{S \subseteq Q : S \text{ contains a state in } F\}$.

Then N accepts a string $w = w_1 w_2 \ldots w_n \Leftrightarrow$ a state in F is reachable via a path $\underbrace{\varepsilon \ldots \varepsilon}_{\geq 0} w_1 \underbrace{\varepsilon \ldots \varepsilon}_{\geq 0} w_2 \ldots w_n \underbrace{\varepsilon \ldots \varepsilon}_{\geq 0}$
$\Leftrightarrow$ a state in $\mathscr{F}$ is reachable from S_0 via the path $w_1 w_2 \ldots w_n \Leftrightarrow D$ accepts w. $\square$