DFA $\equiv$ NFA I

- DFA $\Rightarrow$ NFA
- Formally, a language recognized by a DFA $\Rightarrow$ recognized by an NFA
- The proof is easy because a DFA is an NFA
- However, formally DFA is not an NFA because DFA uses $\Sigma$ but not $\Sigma_\epsilon$
  
  Can easily handle this by adding $q_i, \epsilon \rightarrow \emptyset$

- The other direction: NFA $\Rightarrow$ DFA
DFA $\equiv$ NFA II

- Need to convert NFA to an equivalent DFA
- That is, they recognize the same language
- We do the proof by an example
Consider the following NFA (we discussed this NFA before)

The resulting DFA diagram
Each state is a subset of \( \{1, 2, 3\} \). So each state is an element of \( P(Q) \).

Let's check details of...

\[ \{1, 2\} \rightarrow \{2, 3\} \]
We see

\[ q_1 \xrightarrow{a} \emptyset \]

\[ q_2 \xrightarrow{a} \{ q_2, q_3 \} \]

Thus

\[ \emptyset \cup \{2, 3\} = \{2, 3\} \]

Details of

\[
\begin{array}{c}
\{3\} \\
a \\
\{1, 3\}
\end{array}
\]
We must take care of $\epsilon$

- **Start state:**
  \[
  \{1, 3\} \text{ but not } \{1\}
  \]
  The reason is that in the beginning, even without any input, we can already reach $q_3$

- **Accept states:** any state including $q_1$ is an accept state
Removing unused states

- Some states can never be reached
- We can remove them to simplify the diagram
- Turns out any state **having 1 but without 3** can never be reached
Removing unused states II

\[ a, b \]

\[ \{1, 3\} \]

\[ \{2\} \]

\[ \{2, 3\} \]

\[ \emptyset \]

\[ \{1, 2, 3\} \]
More explanation of example 1.41

- **Idea:**
  Each node includes all states at the current layer
- **Example:** *baa*

```
       q1     q3
      |       |
 q2 ----> q2 ----> q3
  |         |         |
 a        a        a
  |         |         |
 q2 ----> q3 ----> q1 ----> q3
```

September 7, 2022 10 / 13
More explanation of example 1.41 II

We see

\[ \{1, 3\} \xrightarrow{b} \{2\} \xrightarrow{a} \{2, 3\} \xrightarrow{a} \{1, 2, 3\} \]

- Formal description of the procedure
- Given NFA

\[ (Q, \Sigma, \delta, q_0, F) \]

We would like to convert it to a DFA

\[ (Q', \Sigma, \delta', q'_0, F') \]

- Details of this DFA:
More explanation of example 1.41 III

- $Q' = P(Q)$
- $q'_0 \in P(Q)$ includes
  \[
  \{ q_0 \} \cup \{ \text{states reached by } \epsilon \text{ from } q_0 \}
  \]

We call such a set $E(\{ q_0 \})$

- $F' = \{ R \mid R \in Q', R \cap F \neq \emptyset \}$
- $\delta'$:
  \[
  \delta'(R, a) = \bigcup_{r \in R} E(\delta(r, a))
  \]

Now $r$ is a state of NFA and we use $\delta$ to see where to go by taking the input $a$
Note that we cannot just do

$$\bigcup_{r \in R} \delta(r, a)$$

$$\epsilon$$ must be handled