To tell whether an NFA $N$ accepts a string $x = x_1 \cdots x_n$, we want to compute, for each $i$, $0 \leq i \leq n$, the set

$$R_i = \{ q \in Q : N \text{ can process } x_1 \cdots x_i \text{ from } s \text{ to } q \}$$

Goal: $N$ accept $x$ $\Leftrightarrow$ $R_n$ includes a state in $F$.

Origin: $R_0 = \{ q \in Q : N \text{ can process } \varepsilon \text{ from } s \text{ to } q \}$

= the “epsilon closure” of $\{s\}$.

Helpful extra notation: not in any text(!) $S(q, c) = \{ r : q \text{ to } r \text{ by first doing } \alpha \text{ or an arc on } c \text{ then an optional trailing } \varepsilon \text{ arcs} \}$

Induction: Thus given $R_{i-1}$, we can calculate $R_i$ by

$$R_i = \bigcup_{q \in R_{i-1}} S(q, x_i)$$

$\diamond$ Theorem: For every NFA $N = (Q, \Sigma, \delta, s, F)$ we can build an equiv DFA $M = (Q, \Sigma, \Delta, s, F)$

$S \subseteq \text{ eps closure of } \{s\} \text{, e.R}_0$

for all $P \subseteq Q, c \in \Sigma, \Delta(P, c) = \bigcup_{p \in P} S(p, c), RNF \neq \emptyset$
Example:

\[ N = \{ b, a, c \} \]

\[ \delta(1, a) = \{ 1, 2, 3 \} \]
\[ \delta(1, b) = \{ 3, 7 \} \]
\[ \delta(2, a) = \{ 3, 7 \} \]
\[ \delta(2, b) = \emptyset \]
\[ \delta(3, a) = \{ 1, 2, 7 \} \]
\[ \delta(3, b) = \{ 2, 7 \} \]

\[ \Delta(p, c) = \bigcup_{p \in p} \delta(p, c) \]

\[ \delta(1, a) = \{ 1, 2, 7 \} \]
\[ \delta(1, b) = \{ 3, 7 \} \]
\[ \delta(2, a) = \{ 3, 7 \} \]
\[ \delta(2, b) = \emptyset \]
\[ \delta(3, a) = \{ 1, 2, 7 \} \]
\[ \delta(3, b) = \{ 2, 7 \} \]

\[ \Delta(p, c) = \bigcup_{p \in p} \delta(p, c) \]

Use Breadth First Search from \( S \):

\[ \Delta(5, a) = \delta(1, a) \cup \delta(2, a) \]
\[ = \{ 1, 2, 7 \} \cup \{ 3, 7 \} = \{ 1, 2, 3, 7 \} \]
\[ \text{new state} \]

\[ \Delta(5, b) = \delta(1, b) \cup \delta(2, b) \]
\[ = \{ 3, 7 \} \cup \emptyset = \{ 3, 7 \} \]
\[ \text{new state} \]

\[ \Delta(1, 2, 3, 7, a) = \{ 1, 2, 7 \} \cup \{ 3, 7 \} = \{ 1, 2, 3, 7 \} \]
\[ \text{no more new states} \]

\[ \Delta(3, 7, a) = \delta(3, a) = \{ 1, 2, 7 \} \]
\[ \Delta(3, 7, b) = \delta(3, b) = \{ 2, 7 \} \]

\[ \Delta(1, 2, 3, 7, b) = \{ 3, 7 \} \cup \emptyset = \{ 3, 7 \} \]

In lecture 2 pointed out:

\[ N \]

\[ \text{can process } a \text{ to any one of its three states: } \{ \text{ b, c, d, e, f, g } \} \]

\[ \text{cannot process } bbb \text{ from } S \text{ at all. In } N \text{ it goes to } \emptyset. \]

\[ \text{Indeed, } N \text{ cannot process } bbb \text{ at all: from } 5, 12, 37 \text{ it goes to } \emptyset. \]

So \( 1, 2, 3, 7 \) is not a final state accept condition, nor path condition.

The DFA cannot have the states \( \{ 1 \} \) or \( \{ 1, 3, 7 \} \) because they have 1 but not 2.
Example of an NFA that does have a conditional accept condition.

\( L = \{ x \in \{a,b\}^* : x \text{ has aab in it} \} \)

**NFA:**

\[
\begin{array}{c}
\text{a} \\
(\text{a,b}) \\
\end{array}
\]

**OFA** has no dead state because no initial word can forbid having aab later. It has a conditional-accept state once an aab occurs.

**Regexp:** \((a+b)^* aab (a+b)^*\).

\[ L' = \{ x \in \{a,b\}^* : x \text{ ends in aab} \} \]

**NFA:**

\[
\begin{array}{c}
\text{a} \\
\text{a,b} \\
\text{a} \\
\text{b} \\
\end{array}
\]

**OFA** has no dead state, as there is a conditional accept state.

\[ L'' = \{ x \in \{a,b\}^* : x \text{ has an aab and has no bbaa} \} \]

**OFA** has dead state. Hard even to build NFA.