NFAs and DFAs
Reminder: Regular Operations

Let $A$ and $B$ be languages. We define the regular operations union, concatenation, and star as follows:

- **Union:** $A \cup B = \{ x \mid x \in A \text{ or } x \in B \}$.
- **Concatenation:** $A \circ B = \{ xy \mid x \in A \text{ and } y \in B \}$.
- **Star:** $A^* = \{ x_1 x_2 \ldots x_k \mid k \geq 0 \text{ and each } x_i \in A \}$. 
Proofs that Regular Languages Are Closed Under...

- **Union:** $A \cup B = \{x \mid x \in A \text{ or } x \in B\}$.
  - Proof by Construction (last class)

But...

- **Concatenation:** $A \circ B = \{xy \mid x \in A \text{ and } y \in B\}$.
- **Star:** $A^* = \{x_1x_2\ldots x_k \mid k \geq 0 \text{ and each } x_i \in A\}$.

We Need Better Tools!
Deterministic vs. Nondeterministic Finite Automata

Everything discussed thus far was deterministic, thus a DFA

Now, we will expand DFAs to allow multiple potential paths.
What is Determinism?

Are computers deterministic?

“If the current state is known, and the current inputs are known, then the future state is also known.”

- No Choices
- No Randomness
- No Oracles
- No Errors/Cheating

According to our previous definition, FSM == DFA
What is Nondeterminism?

“If the current state is known, and the current inputs are known, there may be multiple possible future states.”

- Random choice?
- Parallel choice and simultaneous execution?
NFAs: Relaxing The Requirements

- Multiple edges with the same label
- $\varepsilon$ (optional) edges
  - $\varepsilon$ is not in $\Sigma$
  - Does not consume input character
- Only need one path to an accept state

What could go wrong?

How do we know which path to try?
- Try them all
- Always make the right choice
A Nondeterministic Finite Automaton is a 5-tuple

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

where:

- **Q**: Set of states (finite)
- **\Sigma**: Alphabet of symbols (finite)
- **\delta**: The transition function
  \[ \delta: Q \times \Sigma \varepsilon \rightarrow \mathcal{P}(Q) \]
- **q_0**: The starting (initial) state
  \[ q_0 \in Q \]
- **F**: The set of “Accept” states
  \[ F \subseteq Q \]
NFA Formal Definition Example

$$N = (Q, \Sigma, \delta, q_0, F)$$

- **Q**: \(\{q_1, q_2, q_3, q_4\}\)
- **\(\Sigma\)**: \(\{0, 1\}\)
- **\(\delta\)**: Transition function
- **\(q_0\)**: Start state: \(q_1\)
- **F**: Accept states: \(\{q_4\}\)

Transition Table:

<table>
<thead>
<tr>
<th>(q)</th>
<th>0</th>
<th>1</th>
<th>(\varepsilon)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(q_1)</td>
<td>({q_1})</td>
<td>({q_1, q_2})</td>
<td>(\emptyset)</td>
</tr>
<tr>
<td>(q_2)</td>
<td>({q_3})</td>
<td>(\emptyset)</td>
<td>({q_3})</td>
</tr>
<tr>
<td>(q_3)</td>
<td>(\emptyset)</td>
<td>({q_4})</td>
<td>(\emptyset)</td>
</tr>
<tr>
<td>(q_4)</td>
<td>({q_4})</td>
<td>({q_4})</td>
<td>(\emptyset)</td>
</tr>
</tbody>
</table>

Transition Diagram:
NFA Formal Definition Example

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

\[ Q = \{ q_1, q_2, q_3, q_4 \} \]
\[ \Sigma = \{ 0, 1 \} \]
\[ \delta(q_0, 0) = q_1 \]
\[ F = \{ q_4 \} \]

DO YOU REMEMBER WHAT MUST COME NEXT?
NFA Formal Definition of Computation

Let $N = (Q, \Sigma, \delta, q_0, F)$

Let $y_1y_2...y_m$ be a string $w$ where $y_i \in \Sigma^*$

$N$ accepts $w$ if there is a sequence of states $r_0, r_1, r_2...r_m$ in $Q$ such that:

1. $r_0 = q_0$,
2. $r_{i+1} \in \delta(r_i, y_{i+1})$ for $0 \leq i < m$, and
3. $r_m \in F$

$N$ “recognizes” language $A$ if $A = \{ w \mid N \text{ accepts } w \}$

We now have a NONDETERMINISTIC tool that we can use to understand Regular Languages!

Wait... Can we say this?!!?
Proof of Equivalency

NFAs and DFAs

Proof by Construction:
A method to convert
\[ N = (Q, \Sigma, \delta, q_0, F) \]
into
\[ M = (Q', \Sigma, \delta', q'_0, F') \]
What Were We Talking About Last Time???

Infinity War spoiler but without context.
Convert NFA to DFA

Proof by Construction. (ignoring $\varepsilon$ for now)

Let $N = (Q, \Sigma, \delta, q_0, F)$ be a NFA that recognizes some language $A$.

Construct a DFA $M = (Q', \Sigma, \delta', q'_0, F')$ that recognizes the same language $A$.

$Q' = \mathcal{P}(Q)$

ex: $\{[p_0], [p_0, p_1], [p_0, p_2], [p_0, p_1, p_2], \ldots\}$

$q'_0 = \{q_0\}$

$F' = \{R \subseteq Q' \mid R \text{ contains an accept state of } N\}$

or, $F' = \{R \subseteq Q' \mid R \cap F \neq \emptyset\}$

$\delta'(R, a) = \{q \in Q \mid q \in \delta(r, a) \text{ for some } r \in R\}$

where $R \subseteq Q'$ and $a \in \Sigma$.

For $\varepsilon$, define $E(R) = \{q \mid q \text{ can be reached from } R \text{ by traveling along 0 or more } \varepsilon \text{ arrows}\}$ where $R \subseteq Q$ and $q \in Q$.

$q'_0 = E(\{q_0\})$

$\delta'(R, a) = \{q \in Q \mid q \in E(\delta(r, a)) \text{ for some } r \in R\}$
Practice: Convert this NFA to a DFA
Practice: Convert this NFA to a DFA
We defined DFAs.
We defined Computation using DFAs.
We defined Regular Languages using DFAs.

We defined NFAs.
We defined Computation using NFAs.
We showed equivalence of NFAs to DFAs.

Now What?
Regular Operations

They eat their fiber.

Union, Concatenation, and Star:
3 Proofs by Construction
Let $N_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ recognize $A_1$, and $N_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ recognize $A_2$.

Construct $N = (Q, \Sigma, \delta, q_0, F)$ to recognize $A_1 \cup A_2$.

$Q = \{q_0\} \cup Q_1 \cup Q_2$

$F = F_1 \cup F_2$

For $q \in Q$ and any $a \in \Sigma$, 

$$\delta(q, a) = \begin{cases} 
\delta_1(q, a) & q \in Q_1 \\
\delta_2(q, a) & q \in Q_2 \\
\{q_1, q_2\} & q = q_0 \text{ and } a = \varepsilon \\
\emptyset & q = q_0 \text{ and } a \neq \varepsilon 
\end{cases}$$
Regular Operations: Concatenation

Let $N_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ recognize $A_1$, and $N_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ recognize $A_2$. Construct $N = (Q, \Sigma, \delta, q_1, F_2)$ to recognize $A_1 \circ A_2$.

$Q = Q_1 \cup Q_2$

For $q \in Q$ and any $a \in \Sigma^*$,

$$
\delta(q, a) = \begin{cases} 
\delta_1(q, a) & q \in Q_1 \text{ and } q \notin F_1 \\
\delta_1(q, a) & q \in F_1 \text{ and } a \neq \varepsilon \\
\delta_1(q, a) \cup \{q_2\} & q \in F_1 \text{ and } a = \varepsilon \\
\delta_2(q, a) & q \in Q_2 
\end{cases}
$$
Let $N_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ recognize $A_1$.

Construct $N = (Q, \Sigma, \delta, q_0, F)$ to recognize $A_1^*$.

$Q = \{q_0\} \cup Q_1$

$F = \{q_0\} \cup F_1$

For $q \in Q$ and any $a \in \Sigma$, 

$$\delta(q, a) = \begin{cases} 
\delta_1(q, a) & q \in Q_1 \text{ and } q \not\in F_1 \\
\delta_1(q, a) & q \in F_1 \text{ and } a \neq \epsilon \\
\delta_1(q, a) \cup \{q_1\} & q \in F_1 \text{ and } a = \epsilon \\
\{q_1\} & q = q_0 \text{ and } a = \epsilon \\
\emptyset & q = q_0 \text{ and } a \neq \epsilon 
\end{cases}$$