Finite Automata
Part One
Recap from Last Time
DFAs

- A **DFA** is a
  - **D**eterministic
  - **F**inite
  - **A**utomaton
- DFAs are the simplest type of automaton that we will see in this course.
DFAs

• A DFA is defined relative to some alphabet $\Sigma$.

• For each state in the DFA, there must be exactly one transition defined for each symbol in $\Sigma$.
  • This is the “deterministic” part of DFA.

• There is a unique start state.

• There are zero or more accepting states.
A language $L$ is called a regular language if there exists a DFA $D$ such that $\mathcal{L}(D) = L$. 
NFAs

- An **NFA** is a
  - **N**ondeterministic
  - **F**inite
  - **A**utomaton
- Can have missing transitions or multiple transitions defined on the same input symbol.
- Accepts if *any possible series of choices leads to an accepting state.*
Hello, NFA!
Hello, NFA!

\[
\begin{array}{c}
\text{start} \quad q_0 \quad \text{h} \quad q_1 \quad \text{i} \quad q_2 \\
\end{array}
\]
Hello, NFA!
Hello, NFA!
Hello, NFA!
Hello, NFA!
Tragedy in Paradise
Tragedy in Paradise

\[ q_0 \xrightarrow{h} q_1 \xrightarrow{i} q_2 \]

\[ \text{start} \quad q_0 \quad h \quad q_1 \quad i \quad q_2 \]

[hip]
Tragedy in Paradise
Tragedy in Paradise

\[
\begin{array}{c}
\text{start} \\
q_0 \\
\quad \quad \quad \quad h \\
q_1 \\
\quad \quad \quad \quad i \\
q_2
\end{array}
\]
Tragedy in Paradise

[Diagram with states q₀, q₁, q₂ and transitions labeled h, i, starting from state q₀]
Tragedy in Paradise
Tragedy in Paradise

\[ q_0 \xrightarrow{h} q_1 \xrightarrow{i} \text{sad smiley face} \]

Start

\[ h \quad i \quad p \]
Tragedy in Paradise
The language of an NFA is
\[ \mathcal{L}(N) = \{ w \in \Sigma^* \mid N \text{ accepts } w \} . \]

What is the language of this NFA?
(Assume \( \Sigma = \{h, i\} \).)
The language of an NFA is
\[ \mathcal{L}(N) = \{ w \in \Sigma^* \mid N \text{ accepts } w \} \].

\[ \Sigma = \{\emptyset, 1\} \]
**ε-Transitions**

- NFAs have a special type of transition called the **ε-transition**.
- An NFA may follow any number of ε-transitions at any time without consuming any input.
ε-Transitions

- NFAs have a special type of transition called the **ε-transition**.
- An NFA may follow any number of ε-transitions at any time without consuming any input.
ε-Transitions

- NFAs have a special type of transition called the **ε-transition**.
- An NFA may follow any number of ε-transitions at any time without consuming any input.
\(\varepsilon\)-Transitions

- NFAs have a special type of transition called the \(\varepsilon\)-transition.
- An NFA may follow any number of \(\varepsilon\)-transitions at any time without consuming any input.
**ε-Transitions**

- NFAs have a special type of transition called the **ε-transition**.
- An NFA may follow any number of ε-transitions at any time without consuming any input.
ε-Transitions

- NFAs have a special type of transition called the **ε-transition**.
- An NFA may follow any number of ε-transitions at any time without consuming any input.
ε-Transitions

- NFAs have a special type of transition called the \textbf{ε-transition}.
- An NFA may follow any number of ε-transitions at any time without consuming any input.

```
start
q_0 \rightarrow a \rightarrow q_1 \rightarrow a \rightarrow q_2
q_3 \rightarrow b, \epsilon \rightarrow q_4 \rightarrow b \rightarrow q_5
```

Input sequence: \texttt{bababb}
ε-Transitions

- NFAs have a special type of transition called the ε-transition.
- An NFA may follow any number of ε-transitions at any time without consuming any input.
ε-Transitions

- NFAs have a special type of transition called the **ε-transition**.
- An NFA may follow any number of ε-transitions at any time without consuming any input.
**ε-Transitions**

- NFAs have a special type of transition called the **ε-transition**.
- An NFA may follow any number of ε-transitions at any time without consuming any input.
**ε-Transitions**

- NFAs have a special type of transition called the **ε-transition**.
- An NFA may follow any number of ε-transitions at any time without consuming any input.
ε-Transitions

• NFAs have a special type of transition called the **ε-transition**.

• An NFA may follow any number of ε-transitions at any time without consuming any input.
ε-Transitions

- NFAs have a special type of transition called the **ε-transition**.
- An NFA may follow any number of ε-transitions at any time without consuming any input.
ε-Transitions

- NFAs have a special type of transition called the **ε-transition**.
- An NFA may follow any number of ε-transitions at any time without consuming any input.
ε-Transitions

- NFAs have a special type of transition called the **ε-transition**.
- An NFA may follow any number of ε-transitions at any time without consuming any input.
ε-Transitions

• NFAs have a special type of transition called the **ε-transition**.

• An NFA may follow any number of ε-transitions at any time without consuming any input.
ε-Transitions

• NFAs have a special type of transition called the $\varepsilon$-transition.

• An NFA may follow any number of $\varepsilon$-transitions at any time without consuming any input.

• NFAs are not required to follow $\varepsilon$-transitions. It's simply another option at the machine's disposal.
Intuiting Nondeterminism

• Nondeterministic machines are a serious departure from physical computers. How can we build up an intuition for them?

• There are two particularly useful frameworks for interpreting nondeterminism:
  • *Perfect positive guessing*
  • *Massive parallelism*
Perfect Positive Guessing

\[ q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_3 \]

Start
Perfect Positive Guessing

\[
\begin{array}{cccccc}
q_0 & \xrightarrow{a} & q_1 & \xrightarrow{b} & q_2 & \xrightarrow{a} q_3 \\
\text{start} & \xrightarrow{\Sigma} & & & & \\
\end{array}
\]
Perfect Positive Guessing

\[
\begin{align*}
q_0 & \xrightarrow{a} q_1 \\
q_1 & \xrightarrow{b} q_2 \\
q_2 & \xrightarrow{a} q_3
\end{align*}
\]

\[\Sigma\]

\[\begin{array}{cccccc}
a & b & a & b & a & a \\
\end{array}\]
Perfect Positive Guessing

\[ a \quad b \quad a \quad b \quad a \quad a \]
Perfect Positive Guessing

\[ q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_3 \]

\[ \Sigma \]

Start

[abaabab]
Perfect Positive Guessing

q₀ q₁ q₂ q₃

Σ

start

a b a b a a

a b a b a a

q₀ → q₁ (a, b, a) → q₂ (a, b, a) → q₃ (a, b, a)
Perfect Positive Guessing

\[ q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_3 \]

\[ \Sigma \]

Input sequence: \[ a \ b \ a \ b \ a \ b \ a \]
Perfect Positive Guessing

\[ q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_3 \]

Input: ababaaba
Perfect Positive Guessing

\[ q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_3 \]

\[ \Sigma \]

a b a b a b a
Perfect Positive Guessing
Perfect Positive Guessing

\[ q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_3 \]

\[ \Sigma \]

Start: \( q_0 \)

States: \( q_0, q_1, q_2, q_3 \)

Transitions:
- \( q_0 \xrightarrow{a} q_1 \)
- \( q_1 \xrightarrow{b} q_2 \)
- \( q_2 \xrightarrow{a} q_3 \)

Input Symbols: \( \Sigma \)

Sequence: \( a, b, a, b, a, b, a, a \)

Image: Seal of Approval
Perfect Positive Guessing

• We can view nondeterministic machines as having *Magic Superpowers* that enable them to guess choices that lead to an accepting state.
  • If there is at least one choice that leads to an accepting state, the machine will guess it.
  • If there are no choices, the machine guesses any one of the wrong guesses.

• There is no known way to physically model this intuition of nondeterminism – this is quite a departure from reality!
Massive Parallelism
Massive Parallelism

\[
\begin{align*}
q_0 & \xrightarrow{a} q_1 \\
q_1 & \xrightarrow{b} q_2 \\
q_2 & \xrightarrow{a} q_3
\end{align*}
\]

\[a \quad b \quad a \quad b \quad a \quad b \quad a\]
Massive Parallelism

\[ q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_3 \]

Input sequence: \[ a b a b a a \]
Massive Parallelism

\[
\begin{align*}
q_0 & \rightarrow q_1 \\
q_1 & \rightarrow q_2 \\
q_2 & \rightarrow q_3
\end{align*}
\]

Transition:

- From \(q_0\) to \(q_1\) on input \(a\)
- From \(q_1\) to \(q_2\) on input \(b\)
- From \(q_2\) to \(q_3\) on input \(a\)

Input Sequence:

\[a \ b \ a \ b \ a \ b \ a\]
Massive Parallelism

\[
\begin{align*}
q_0 & \rightarrow a \rightarrow q_1 \rightarrow b \rightarrow q_2 \rightarrow a \\
& \xrightarrow{\Sigma} \quad \text{start} \quad \xrightarrow{\Sigma} \quad q_0
\end{align*}
\]
Massive Parallelism

\[ \Sigma \]

\[ a \rightarrow q_0 \rightarrow a \\
q_1 \rightarrow b \\
q_2 \rightarrow a \\
q_3 \]

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>a</th>
<th>b</th>
<th>a</th>
<th>b</th>
<th>a</th>
</tr>
</thead>
</table>

\[ \uparrow \]
Massive Parallelism

\[ \Sigma \]

\[ q_0 \rightarrow q_1 \rightarrow q_2 \rightarrow q_3 \]

\[ \text{start} \]

\[ q_0, q_1, q_2, q_3 \]

\[ a, b, a, b, a \]
Massive Parallelism

\[ a b a b a b a \]
Massive Parallelism

\[ q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_3 \]

Input sequence:
\[ a \ b \ a \ b \ b \ a \ a \]
Massive Parallelism

\[ q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_3 \]

Input sequence:

\[ a \ b \ a \ b \ a \ b \ a \]
Massive Parallelism

a b a b a b a
Massive Parallelism

\[ \begin{align*}
q_0 & \xrightarrow{a} q_1 \\
q_1 & \xrightarrow{b} q_2 \\
q_2 & \xrightarrow{a} q_3
\end{align*} \]
Massive Parallelism

\[ q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_3 \]

Input sequence: \[ a \ b \ a \ b \ a \ a \]

Transition diagram with states labeled and transitions marked by symbols.
Massive Parallelism

\[ q_0 \rightarrow q_1 \rightarrow q_2 \rightarrow q_3 \]

\[ \sum \]

\[ a \, b \, a \, b \, a \, b \, a \]
Massive Parallelism

\[ \sum \]

```
\begin{array}{cccc}
a & b & a & b & a \\
\end{array}
```
Massive Parallelism

\[ q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_3 \]

Input: \[ a \ b \ a \ b \ a \ b \ a \]
Massive Parallelism

\[ q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_3 \]

Input: \[ ababaab \]
Massive Parallelism
Massive Parallelism
Massive Parallelism

\[ \sum \]

\[ q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_3 \]

\[ \text{a b a b a b a} \]
Massive Parallelism

start $\rightarrow q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_3$

$\Sigma$

a b a b a a
Massive Parallelism

- $q_0$
- $q_1$
- $q_2$
- $q_3$

Transition:
- $a \rightarrow q_1$
- $b \rightarrow q_2$
- $a \rightarrow q_3$

Input: $a b a b a a$

Start state: $q_0$

Accepting state: $q_3$
Massive Parallelism

\[ a \ b \ a \ b \ a \ b \ a \]
Massive Parallelism

\[ q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_3 \]

\[ \Sigma \]

Input sequence: a b a b a b a
Massive Parallelism

\[ q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_3 \]

Input: \[ a b a b a b a \]
Massive Parallelism

q₃
q₂
q₁
q₀

Σ

a

b

a

b

a


a b a b a b a
Massive Parallelism

start

$q_0$  $a$  $q_1$  $b$  $q_2$  $a$  $q_3$

$\Sigma$

a b a b a a
Massive Parallelism

\[ \sum \]

\[
\text{start} \quad q_0 \quad a \quad q_1 \quad b \quad q_2 \quad a \quad q_3
\]

\[
\text{a b a b a b a}
\]
Massive Parallelism

\[ \sum \]

Diagram:

Start state: \( q_0 \)
- \( q_0 \) transitions to \( q_1 \) on input 'a'.
- \( q_1 \) transitions to \( q_2 \) on input 'b'.
- \( q_2 \) transitions to \( q_3 \) on input 'a'.

Input sequence: \( abaaba \)
Massive Parallelism

\[
\begin{align*}
\Sigma & \rightarrow q_0 \\
& \rightarrow q_1 \quad a \quad b \\
& \rightarrow q_2 \quad a \\
& \rightarrow q_3 \\
\end{align*}
\]
Massive Parallelism

We're in at least one accepting state, so there's some path that gets us to an accepting state.

\[ a \ b \ a \ b \ a \ a \]
Massive Parallelism

start $\rightarrow q_0 \rightarrow a \rightarrow q_1 \rightarrow b \rightarrow q_2 \rightarrow a \rightarrow q_3$

$\Sigma$

A b a b a b
Massive Parallelism

\[
\begin{align*}
q_0 &\xrightarrow{a} q_1 \\
q_1 &\xrightarrow{b} q_2 \\
q_2 &\xrightarrow{a} q_3
\end{align*}
\]

Input alphabet: \(\Sigma = \{a, b\}\)
Massive Parallelism

\[ \sum \]

\[ q_0 \rightarrow q_1 \rightarrow q_2 \rightarrow q_3 \]

\[ \begin{array}{c}
  a \\
  b \\
  a \\
  b
\end{array} \]
Massive Parallelism

\[
\Sigma
\]

q₀ → a → q₁ → b → q₂ → a → q₃

a b a b b
Massive Parallelism

\[ \Sigma \]

\[
\begin{align*}
q_0 & \xrightarrow{a} q_1 \\
q_1 & \xrightarrow{b} q_2 \\
q_2 & \xrightarrow{a} q_3
\end{align*}
\]

\[
\begin{array}{cccc}
a & b & a & b \\
\end{array}
\]

\[
\uparrow
\]
Massive Parallelism

start

$q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_3$

$a b a b a b$

$\Sigma$
Massive Parallelism

\[ q_0 \rightarrow q_1 \rightarrow q_2 \rightarrow q_3 \]

Input symbols: \( \Sigma \)
Massive Parallelism

![Diagram of a finite state machine](attachment:image.png)
Massive Parallelism

\[ q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_3 \]

\[ \Sigma \]

\[ a \quad b \quad a \quad b \quad a \quad b \quad b \]
Massive Parallelism

\[ \Sigma \]

\[ \text{start} \quad q_0 \quad a \quad q_1 \quad b \quad q_2 \quad a \quad q_3 \]

\[ \text{a b a b b} \]
Massive Parallelism

\[ q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_3 \]

\[ \sum \]

\[
\begin{array}{cccc}
\text{a} & \text{b} & \text{a} & \text{b} \\
\end{array}
\]
Massive Parallelism

\[ q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_3 \]

Symbols:
- \( \Sigma \) to represent the input alphabet
- \( a, b \) as input symbols

Sequence of inputs:
- \( a b a b b \)
Massive Parallelism

\[ q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_3 \]

Symbols:
- \( \Sigma \)
- \( q_0, q_1, q_2, q_3 \)
- Start state: \( q_0 \)
- Transition labels: \( a, b \)
Massive Parallelism

\[ ∑ \]

\[ q_0 \rightarrow q_1 \rightarrow q_2 \rightarrow q_3 \]

\[ a \text{ b a a b b} \]
Massive Parallelism
Massive Parallelism

\[ q_0 \rightarrow_{a} q_1 \rightarrow_{b} q_2 \rightarrow_{a} q_3 \]

\[ \sum \]

- Start state: \( q_0 \)
- \( q_0 \rightarrow_{a} q_1 \)
- \( q_1 \rightarrow_{b} q_2 \)
- \( q_2 \rightarrow_{a} q_3 \)

Input sequence: \( a \ b \ a \ b \ a \ b \)
Massive Parallelism

$q_0$ \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_3

\begin{array}{cccc}
a & b & a & b \\
\end{array}
Massive Parallelism
Massive Parallelism

\[ \Sigma \]

\[
\begin{align*}
q_0 & \xrightarrow{a} q_1 \\
q_1 & \xrightarrow{b} q_2 \\
q_2 & \xrightarrow{a} q_3
\end{align*}
\]

\[ \begin{array}{cccc}
a & b & a & b \\
\end{array} \]
Massive Parallelism

\[ q_0 \xrightarrow{\sum} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_3 \]
Massive Parallelism

\[ q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_3 \]

\[ \sum \]

\[
\begin{array}{cccc}
a & b & a & b \\
\end{array}
\]
Massive Parallelism

\[ q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_3 \]

Input: \[ \Sigma \]

Transition:
- \[ q_0 \xrightarrow{a} q_1 \]
- \[ q_1 \xrightarrow{b} q_2 \]
- \[ q_2 \xrightarrow{a} q_3 \]

States:
- \[ q_0 \]
- \[ q_1 \]
- \[ q_2 \]
- \[ q_3 \]
Massive Parallelism

\[ q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_3 \]

\( \Sigma \xrightarrow{\text{start}} q_0 \)

Input sequence: \( a \ b \ a \ b \ b \)
Massive Parallelism

\[
\begin{align*}
q_0 &\xrightarrow{a} q_1 & q_1 &\xrightarrow{b} q_2 & q_2 &\xrightarrow{a} q_3
\end{align*}
\]

\[\Sigma\]
Massive Parallelism

We're not in any accepting state, so no possible path accepts.
Massive Parallelism

• An NFA can be thought of as a DFA that can be in many states at once.

• At each point in time, when the NFA needs to follow a transition, it tries all the options at the same time.

• (Here's a rigorous explanation about how this works; read this on your own time).

  • Start off in the set of all states formed by taking the start state and including each state that can be reached by zero or more ε-transitions.

  • When you read a symbol a in a set of states S:
    - Form the set S' of states that can be reached by following a single a transition from some state in S.
    - Your new set of states is the set of states in S', plus the states reachable from S' by following zero or more ε-transitions.
So What?

- Each intuition of nondeterminism is useful in a different setting:
  - Perfect guessing is a great way to think about how to design a machine.
  - Massive parallelism is a great way to test machines – and has nice theoretical implications.
- Nondeterministic machines may not be feasible, but they give a great basis for interesting questions:
  - Can any problem that can be solved by a nondeterministic machine be solved by a deterministic machine?
  - Can any problem that can be solved by a nondeterministic machine be solved efficiently by a deterministic machine?
- The answers vary from automaton to automaton.
Designing NFAs
Designing NFAs

- *Embrace the nondeterminism!*

- Good model: **Guess-and-check**:  
  - Is there some information that you'd really like to have? Have the machine *nondeterministically guess* that information.
  - Then, have the machine *deterministically check* that the choice was correct.

- The *guess* phase corresponds to trying lots of different options.

- The *check* phase corresponds to filtering out bad guesses or wrong options.
Guess-and-Check

\[ L = \{ \ w \in \{0, 1\}^* \mid \text{w ends in 010 or 101} \} \]
Guess-and-Check

\[ L = \{ w \in \{0, 1\}^* \mid w \text{ ends in } 010 \text{ or } 101 \} \]
Guess-and-Check

\[ L = \{ w \in \{0, 1\}^* \mid w \text{ ends in } 010 \text{ or } 101 \} \]

Nondeterministically guess when the end of the string is coming up.

Deterministically check whether you were correct.
Guess-and-Check

$L = \{ w \in \{0, 1\}^* \mid w \text{ ends in } 010 \text{ or } 101 \} \}$
Guess-and-Check

\[ L = \{ w \in \{0, 1\}^* \mid w \text{ ends in 010 or 101} \} \]
Guess-and-Check

\[ L = \{ w \in \{0, 1\}^* \mid w \text{ ends in } 010 \text{ or } 101 \} \]
Guess-and-Check

\[ L = \{ \; w \in \{0, 1\}^* \; | \; \text{w ends in 010 or 101} \; \} \]
$$L = \{ w \in \{0, 1\}^* \mid w \text{ ends in } 010 \text{ or } 101 \}$$
Guess-and-Check

\[ L = \{ \ w \in \{0, 1\}^* \mid w \text{ ends in } 010 \text{ or } 101 \ \} \]
Guess-and-Check

\[ L = \{ w \in \{0, 1\}^* \mid w \text{ ends in } 010 \text{ or } 101 \} \]
Guess-and-Check

\[ L = \{ \, w \in \{0, 1\}^* \mid w \text{ ends in } 010 \text{ or } 101 \, \} \]
Guess-and-Check

\[ L = \{ w \in \{0, 1\}^* \mid w \text{ ends in } 010 \text{ or } 101 \} \]
Guess-and-Check

\[ L = \{ w \in \{0, 1\}^* \mid w \text{ ends in } 010 \text{ or } 101 \} \]
Guess-and-Check

\[ L = \{ w \in \{a, b, c\}^* \mid \text{at least one of } a, b, \text{ or } c \text{ is not in } w \} \]
Guess-and-Check

$L = \{ w \in \{a, b, c\}^* \mid \text{at least one of } a, b, \text{ or } c \text{ is not in } w \}$
Guess-and-Check

\[ L = \{ w \in \{a, b, c\}^* \mid \text{at least one of } a, b, \text{ or } c \text{ is not in } w \} \]

Nondeterministically guess which character is missing.

Deterministically check whether that character is indeed missing.
Guess-and-Check

$L = \{ w \in \{a, b, c\}^* \mid \text{at least one of } a, b, \text{ or } c \text{ is not in } w \}$
Guess-and-Check

$L = \{ w \in \{a, b, c\}^* \mid \text{at least one of } a, b, \text{ or } c \text{ is not in } w \}$
Guess-and-Check

$L = \{ w \in \{a, b, c\}^* \mid \text{at least one of } a, b, \text{ or } c \text{ is not in } w \}$
Guess-and-Check

$L = \{ \ w \in \{a, b, c\}^* \mid \text{at least one of } a, b, \text{ or } c \text{ is not in } w \ \}$
Guess-and-Check

\[ L = \{ w \in \{a, b, c\}^* \mid \text{at least one of } a, b, \text{ or } c \text{ is not in } w \} \]
Guess-and-Check

$L = \{ w \in \{a, b, c\}^* \mid \text{at least one of } a, b, \text{ or } c \text{ is not in } w \}$
Guess-and-Check

$L = \{ \ w \in \{a, b, c\}^* \mid \text{at least one of } a, b, \text{ or } c \text{ is not in } w \ \}$
Guess-and-Check

\[ L = \{ w \in \{a, b, c\}^* \mid \text{at least one of } a, b, \text{ or } c \text{ is not in } w \} \]
Just how powerful are NFAs?
NFAs and DFAs

- Any language that can be accepted by a DFA can be accepted by an NFA.
- Why?
  - Every DFA essentially already is an NFA!
- **Question**: Can any language accepted by an NFA also be accepted by a DFA?
- Surprisingly, the answer is **yes**!
Tabular DFAs

\[ q_0 \rightarrow 1 \rightarrow q_1 \rightarrow 1 \rightarrow q_2 \rightarrow 0 \rightarrow q_3 \]

\[ \begin{array}{c|cc}
\text{state} & 0 & 1 \\
\hline
q_0 & & \\
q_1 & & \\
q_2 & & \\
q_3 & & \\
\end{array} \]
Tabular DFAs

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q_0$</td>
<td>$q_1$</td>
<td>$q_0$</td>
</tr>
<tr>
<td>$q_1$</td>
<td>$q_3$</td>
<td>$q_2$</td>
</tr>
<tr>
<td>$q_2$</td>
<td>$q_3$</td>
<td>$q_0$</td>
</tr>
<tr>
<td>$q_3$</td>
<td>$q_3$</td>
<td>$q_3$</td>
</tr>
</tbody>
</table>
Tabular DFAs

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q_0$</td>
<td>$q_1$</td>
<td>$q_0$</td>
</tr>
<tr>
<td>$q_1$</td>
<td>$q_3$</td>
<td>$q_2$</td>
</tr>
<tr>
<td>$q_2$</td>
<td>$q_3$</td>
<td>$q_0$</td>
</tr>
<tr>
<td>$q_3$</td>
<td>$q_3$</td>
<td>$q_3$</td>
</tr>
</tbody>
</table>
Tabular DFAs

These stars indicate accepting states.
Tabular DFAs

Since this is the first row, it's the start state.
Tabular DFAs

(start 0)

$q_0$ 1

$q_0$ 0

$q_1$ 1

$q_2$ 0

$q_3$ 0

$\Sigma$

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q_0$</td>
<td>$q_1$</td>
<td>$q_0$</td>
</tr>
<tr>
<td>$q_1$</td>
<td>$q_3$</td>
<td>$q_2$</td>
</tr>
<tr>
<td>$q_2$</td>
<td>$q_3$</td>
<td>$q_0$</td>
</tr>
<tr>
<td>$q_3$</td>
<td>$q_3$</td>
<td>$q_3$</td>
</tr>
</tbody>
</table>

Question to ponder: Why isn’t there a column here for $\Sigma$?
int kTransitionTable[kNumStates][kNumSymbols] = {
    {0, 0, 1, 3, 7, 1, …},
    …
};
bool kAcceptTable[kNumStates] = {
    false,
    true,
    true,
    …
};
bool SimulateDFA(string input) {
    int state = 0;
    for (char ch: input) {
        state = kTransitionTable[state][ch];
    }
    return kAcceptTable[state];
}
Thought Experiment:
How would you simulate an NFA in software?
\[ \sum \]

start

\[ q_0 \] - a - \[ q_1 \] - b - \[ q_2 \] - a - \[ q_3 \]

\[ q_0 \rightarrow q_1 \rightarrow q_2 \rightarrow q_3 \]

Start state: \( q_0 \)

Transitions:
- \( q_0 \rightarrow q_1 \) on \( \Sigma \)
- \( q_1 \rightarrow q_2 \) on \( b \)
- \( q_2 \rightarrow q_3 \) on \( a \)

Input tape:
- \( \ldots \) \( ? \) \( ? \) \( ? \) \( ? \) \( ? \) \( a \) \( ? \) \( ? \) \( ? \) \( ? \) \( ? \) \( ? \) \( \ldots \)
\[ q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_3 \]

Transition table:

<table>
<thead>
<tr>
<th>State</th>
<th>Transition</th>
<th>Next State</th>
</tr>
</thead>
<tbody>
<tr>
<td>{q_0}</td>
<td>a</td>
<td>{q_0, q_1}</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Graphical representation:

- Start state: \( q_0 \)
- Transitions:
  - \( a \) from \( q_0 \) to \( q_1 \)
  - \( b \) from \( q_1 \) to \( q_2 \)
  - \( a \) from \( q_2 \) to \( q_3 \)
- Final state: \( q_3 \)
<table>
<thead>
<tr>
<th>State</th>
<th>Symbol</th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>{q_0}</td>
<td>{q_0, q_1}</td>
<td></td>
<td>{q_0}</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
\[ q_0 \xrightarrow{\Sigma} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_3 \]

Transition table:

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>{q_0}</td>
<td>{q_0, q_1}</td>
<td>{q_0}</td>
</tr>
<tr>
<td>{q_0, q_1}</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>a</td>
<td>b</td>
</tr>
<tr>
<td>----------------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>{q_0}</td>
<td>{q_0, q_1}</td>
<td>{q_0}</td>
</tr>
<tr>
<td>{q_0, q_1}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>{q_0, q_1}</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Diagram:**

- Start state: \(q_0\)
- Transition labels: a, b
- States: \(q_0, q_1, q_2, q_3\)
- Transition arrows:
  - \(q_0 \xrightarrow{a} q_1\)
  - \(q_1 \xrightarrow{b} q_2\)
  - \(q_2 \xrightarrow{a} q_3\)
  - \(q_3\) is a loop state with self-transition labeled \(\Sigma\)

**Table:**

<table>
<thead>
<tr>
<th>State</th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>{q_0}</td>
<td>{q_0, q_1}</td>
<td>{q_0}</td>
</tr>
<tr>
<td>{q_0, q_1}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>{q_0, q_1}</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
\begin{align*}
\Sigma & \rightarrow \{q_0\} \\
& \rightarrow \{q_0, q_1\} \\
& \rightarrow \{q_0, q_1\} \\
& \rightarrow \{q_0\}
\end{align*}
\[
\begin{array}{c|c|c}
\{q_0\} & \{q_0, q_1\} & \{q_0\} \\
\{q_0, q_1\} & \{q_0, q_1\} & \\
\{q_0, q_1\} & & \\
& & \\
\end{array}
\]
The given diagram represents a finite automaton with the following states: $q_0$, $q_1$, $q_2$, and $q_3$. The transitions are as follows:

- From $q_0$ to $q_1$ on input $a$.
- From $q_1$ to $q_2$ on input $b$.
- From $q_2$ to $q_3$ on input $a$.
- $q_3$ is a final state.

The table below shows the transitions for inputs $a$ and $b$.

<table>
<thead>
<tr>
<th>State</th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>${q_0}$</td>
<td>${q_0, q_1}$</td>
<td>${q_0}$</td>
</tr>
<tr>
<td>${q_0, q_1}$</td>
<td>${q_0, q_1}$</td>
<td>${q_0, q_2}$</td>
</tr>
</tbody>
</table>
\[
\begin{array}{c|c|c}
& \text{a} & \text{b} \\
\hline
\{q_0\} & \{q_0, q_1\} & \{q_0\} \\
\{q_0, q_1\} & \{q_0, q_1\} & \{q_0, q_2\} \\
\{q_0, q_2\} & & \\
\end{array}
\]
\[ \begin{array}{ccc}
\{ q_0 \} & \{ q_0, q_1 \} & \{ q_0 \} \\
\{ q_0, q_1 \} & \{ q_0, q_1 \} & \{ q_0, q_2 \} \\
\{ q_0, q_2 \} & & \\
\end{array} \]
The given automaton has the following transitions:

- **Start State:** $q_0$
- **Transition:**
  - From $q_0$: On input $a$, move to $q_1$.
  - From $q_1$: On input $b$, move to $q_2$.
  - From $q_2$: On input $a$, move to $q_3$.
- **States:** $q_0$, $q_1$, $q_2$, $q_3$
- **Input Symbols:** $a$, $b$
- **Initial State:** $q_0$
- **Accepting States:** $q_3$

The transition table is as follows:

<table>
<thead>
<tr>
<th>Current State</th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>${q_0}$</td>
<td>${q_0, q_1}$</td>
<td>${q_0}$</td>
</tr>
<tr>
<td>${q_0, q_1}$</td>
<td>${q_0, q_1}$</td>
<td>${q_0, q_2}$</td>
</tr>
<tr>
<td>${q_0, q_2}$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
\begin{align*}
q_0 & \xrightarrow{a} q_1 \\
q_1 & \xrightarrow{b} q_2 \\
q_2 & \xrightarrow{a} q_3 \\
\Sigma & \xrightarrow{} q_0
\end{align*}

\begin{array}{|c|c|c|}
\hline
\text{States} & \text{a} & \text{b} \\
\hline
\{ q_0 \} & \{ q_0, q_1 \} & \{ q_0 \} \\
\{ q_0, q_1 \} & \{ q_0, q_1 \} & \{ q_0, q_2 \} \\
\{ q_0, q_2 \} & & \\
\hline
\end{array}
The diagram shows a finite automaton with the following states and transitions:

- **States:**
  - $q_0$ (start state)
  - $q_1$
  - $q_2$
  - $q_3$

- **Transitions:**
  - From $q_0$ on input $a$, transition to $q_1$.
  - From $q_1$ on input $b$, transition to $q_2$.
  - From $q_2$ on input $a$, transition to $q_3$.
  - From $q_0$ on any input ($\Sigma$), transition to $q_0$.

The table below shows the transition function:

<table>
<thead>
<tr>
<th>State</th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>{q_0}</td>
<td>{q_0, q_1}</td>
<td>{q_0}</td>
</tr>
<tr>
<td>{q_0, q_1}</td>
<td>{q_0, q_1}</td>
<td>{q_0, q_2}</td>
</tr>
<tr>
<td>{q_0, q_2}</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>a</td>
<td>b</td>
</tr>
<tr>
<td>-------</td>
<td>----------------</td>
<td>----------------</td>
</tr>
<tr>
<td>${q_0}$</td>
<td>${q_0, q_1}$</td>
<td>${q_0}$</td>
</tr>
<tr>
<td>${q_0, q_1}$</td>
<td>${q_0, q_1}$</td>
<td>${q_0, q_2}$</td>
</tr>
<tr>
<td>${q_0, q_2}$</td>
<td>${q_0, q_1, q_3}$</td>
<td></td>
</tr>
</tbody>
</table>

Transition diagram:
- Start state: $q_0$
- Transitions:
  - $q_0 \xrightarrow{a} q_1$
  - $q_1 \xrightarrow{b} q_2$
  - $q_2 \xrightarrow{a} q_3$
  - $\Sigma \xrightarrow{a} q_0$ (loop on $q_0$)
\[
\Sigma
\]

\begin{center}
\begin{tikzpicture}
    \node[state, initial] (q0) at (0,0) {$q_0$};
    \node[state] (q1) at (2,0) {$q_1$};
    \node[state, green] (q2) at (4,0) {$q_2$};
    \node[state, accepting] (q3) at (6,0) {$q_3$};
    \draw[->] (q0) edge node {$a$} (q1);
    \draw[->] (q1) edge node {$b$} (q2);
    \draw[->] (q2) edge node {$a$} (q3);
\end{tikzpicture}
\end{center}

\[
\begin{array}{|c|c|c|}
\hline
& \text{a} & \text{b} \\
\hline
\{q_0\} & \{q_0, q_1\} & \{q_0\} \\
\{q_0, q_1\} & \{q_0, q_1\} & \{q_0, q_2\} \\
\{q_0, q_2\} & \{q_0, q_1, q_3\} & \text{---} \\
\hline
\end{array}
\]
<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>{q_0}</td>
<td>{q_0, q_1}</td>
<td>{q_0}</td>
</tr>
<tr>
<td>{q_0, q_1}</td>
<td>{q_0, q_1}</td>
<td>{q_0, q_2}</td>
</tr>
<tr>
<td>{q_0, q_2}</td>
<td>{q_0, q_1, q_3}</td>
<td></td>
</tr>
</tbody>
</table>

Diagram:
- Start state: \(q_0\)
- Transitions:
  - \(q_0\) on \(a\) to \(q_1\)
  - \(q_1\) on \(b\) to \(q_2\)
  - \(q_2\) on \(a\) to \(q_3\)
  - \(q_3\) on \(\Sigma\) back to \(q_0\)

Transition Table:
- \(\Sigma\):
  - For \(a\): \\{q_0, q_1\}\}
  - For \(b\): \\{q_0\}\}
- \(a\):
  - \\{q_0\}\}
  - \\{q_0, q_1\}\}
  - \\{q_0, q_2\}\}
  - \\{q_0, q_1, q_3\}\}
- \(b\):
  - \\{q_0\}\}
  - \\{q_0, q_2\}\}
  - \\{q_0, q_1, q_3\}\}
The given DFA is:

- **States:** $q_0, q_1, q_2, q_3$
- **Start State:** $q_0$
- **Accepting State:** $q_3$
- **Transitions:**
  - $a$: $q_0 \rightarrow q_1, q_0 \rightarrow q_0$
  - $b$: $q_1 \rightarrow q_2, q_2 \rightarrow q_2, q_3 \rightarrow q_3$

**Transition Table:**

<table>
<thead>
<tr>
<th>State</th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>${q_0}$</td>
<td>${q_0, q_1}$</td>
<td>${q_0}$</td>
</tr>
<tr>
<td>${q_0, q_1}$</td>
<td>${q_0, q_1}$</td>
<td>${q_0, q_2}$</td>
</tr>
<tr>
<td>${q_0, q_2}$</td>
<td>${q_0, q_1, q_3}$</td>
<td></td>
</tr>
<tr>
<td>${q_0, q_1, q_2}$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The diagram illustrates a finite automaton with states $q_0, q_1, q_2,$ and $q_3$. The transitions are labeled with inputs $a$ and $b$. The table below outlines the transitions:

<table>
<thead>
<tr>
<th>State</th>
<th>Transition a</th>
<th>Transition b</th>
</tr>
</thead>
<tbody>
<tr>
<td>${q_0}$</td>
<td>${q_0, q_1}$</td>
<td>${q_0}$</td>
</tr>
<tr>
<td>${q_0, q_1}$</td>
<td>${q_0, q_1}$</td>
<td>${q_0, q_2}$</td>
</tr>
<tr>
<td>${q_0, q_2}$</td>
<td>${q_0, q_1, q_3}$</td>
<td>${q_0}$</td>
</tr>
</tbody>
</table>

The start state is $q_0$.
\[
\Sigma 
\]

\[
\begin{array}{ccc}
\text{a} & \text{b} \\
\{q_0\} & \{q_0, q_1\} & \{q_0\} \\
\{q_0, q_1\} & \{q_0, q_1\} & \{q_0, q_2\} \\
\{q_0, q_2\} & \{q_0, q_1, q_3\} & \{q_0\} \\
\{q_0, q_1, q_3\} & & \\
\end{array}
\]
\[
\begin{align*}
q_0 & \xrightarrow{a} q_1 \\
q_1 & \xrightarrow{b} q_2 \\
q_2 & \xrightarrow{a} q_3
\end{align*}
\]

\[
\begin{array}{|c|c|c|}
\hline
\text{State} & \text{a} & \text{b} \\
\hline
\{q_0\} & \{q_0, q_1\} & \{q_0\} \\
\{q_0, q_1\} & \{q_0, q_1\} & \{q_0, q_2\} \\
\{q_0, q_2\} & \{q_0, q_1, q_3\} & \{q_0\} \\
\{q_0, q_1, q_3\} & & \\
\hline
\end{array}
\]
The given diagram represents a finite automaton with the following transitions:

- Start state: $q_0$
- Transitions:
  - From $q_0$ to $q_1$ on input $a$
  - From $q_1$ to $q_2$ on input $b$
  - From $q_2$ to $q_3$ on input $a$
  - From $q_3$ to $q_0$ on input $b$

The table below lists the next states for inputs $a$ and $b$ for each set of current states:

<table>
<thead>
<tr>
<th>Current States</th>
<th>Input a</th>
<th>Input b</th>
</tr>
</thead>
<tbody>
<tr>
<td>${ q_0 }$</td>
<td>${ q_0, q_1 }$</td>
<td>${ q_0 }$</td>
</tr>
<tr>
<td>${ q_0, q_1 }$</td>
<td>${ q_0, q_1 }$</td>
<td>${ q_0, q_2 }$</td>
</tr>
<tr>
<td>${ q_0, q_2 }$</td>
<td>${ q_0, q_1, q_3 }$</td>
<td>${ q_0 }$</td>
</tr>
<tr>
<td>${ q_0, q_1, q_3 }$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The diagram represents a deterministic finite automaton (DFA). The states are labeled as $q_0$, $q_1$, $q_2$, and $q_3$. The transitions are as follows:

- From $q_0$, on input $a$, move to $q_1$.
- From $q_1$, on input $b$, move to $q_2$.
- From $q_2$, on input $a$, move to $q_3$.
- From $q_3$, on input $a$, return to $q_0$.
- The initial state is $q_0$.

The table below shows the transition function:

<table>
<thead>
<tr>
<th>State Set</th>
<th>$a$</th>
<th>$b$</th>
</tr>
</thead>
<tbody>
<tr>
<td>${q_0}$</td>
<td>${q_0, q_1}$</td>
<td>${q_0}$</td>
</tr>
<tr>
<td>${q_0, q_1}$</td>
<td>${q_0, q_1}$</td>
<td>${q_0, q_2}$</td>
</tr>
<tr>
<td>${q_0, q_2}$</td>
<td>${q_0, q_1, q_3}$</td>
<td>${q_0}$</td>
</tr>
<tr>
<td>${q_0, q_1, q_3}$</td>
<td>${q_0, q_1}$</td>
<td></td>
</tr>
</tbody>
</table>
\begin{align*}
    &\{q_0\} \quad \{q_0, q_1\} \quad \{q_0\} \\
    \{q_0, q_1\} \quad \{q_0, q_1\} \quad \{q_0, q_2\} \\
    \{q_0, q_2\} \quad \{q_0, q_1, q_3\} \quad \{q_0\} \\
    \{q_0, q_1, q_3\} \quad \{q_0, q_1\} \\
\end{align*}
<table>
<thead>
<tr>
<th>State Set</th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>{q_0}</td>
<td>{q_0, q_1}</td>
<td>{q_0}</td>
</tr>
<tr>
<td>{q_0, q_1}</td>
<td>{q_0, q_1}</td>
<td>{q_0, q_2}</td>
</tr>
<tr>
<td>{q_0, q_2}</td>
<td>{q_0, q_1, q_3}</td>
<td>{q_0}</td>
</tr>
<tr>
<td>{q_0, q_1, q_3}</td>
<td>{q_0, q_1}</td>
<td></td>
</tr>
</tbody>
</table>
\[
\begin{array}{c|c|c}
\text{a} & \text{b} \\
\hline
\{q_0\} & \{q_0, q_1\} & \{q_0\} \\
\{q_0, q_1\} & \{q_0, q_1\} & \{q_0, q_2\} \\
\{q_0, q_2\} & \{q_0, q_1, q_3\} & \{q_0\} \\
\{q_0, q_1, q_3\} & \{q_0, q_1\} \\
\end{array}
\]
<table>
<thead>
<tr>
<th>State Set</th>
<th>$a$</th>
<th>$b$</th>
</tr>
</thead>
<tbody>
<tr>
<td>${q_0}$</td>
<td>${q_0, q_1}$</td>
<td>${q_0}$</td>
</tr>
<tr>
<td>${q_0, q_1}$</td>
<td>${q_0, q_1}$</td>
<td>${q_0, q_2}$</td>
</tr>
<tr>
<td>${q_0, q_2}$</td>
<td>${q_0, q_1, q_3}$</td>
<td>${q_0}$</td>
</tr>
<tr>
<td>${q_0, q_1, q_3}$</td>
<td>${q_0, q_1}$</td>
<td></td>
</tr>
</tbody>
</table>
The given automaton has the following transitions:

- Start state: \(q_0\)
- Labels on transitions:
  - \(a\) from \(q_0\) to \(q_1\)
  - \(b\) from \(q_1\) to \(q_2\)
  - \(a\) from \(q_2\) to \(q_3\)

The table below summarizes the transitions for inputs \(a\) and \(b\):

<table>
<thead>
<tr>
<th>State Set</th>
<th>Input a</th>
<th>Input b</th>
</tr>
</thead>
<tbody>
<tr>
<td>({q_0})</td>
<td>({q_0, q_1})</td>
<td>({q_0})</td>
</tr>
<tr>
<td>({q_0, q_1})</td>
<td>({q_0, q_1})</td>
<td>({q_0, q_2})</td>
</tr>
<tr>
<td>({q_0, q_2})</td>
<td>({q_0, q_1, q_3})</td>
<td>({q_0})</td>
</tr>
<tr>
<td>({q_0, q_1, q_3})</td>
<td>({q_0, q_1})</td>
<td>({q_0, q_2})</td>
</tr>
</tbody>
</table>
\[ \begin{array}{c|cc}
 q_0 & a & b \\
 \{q_0\} & \{q_0, q_1\} & \{q_0\} \\
 \{q_0, q_1\} & \{q_0, q_1\} & \{q_0, q_2\} \\
 \{q_0, q_2\} & \{q_0, q_1, q_3\} & \{q_0\} \\
 \{q_0, q_1, q_3\} & \{q_0, q_1\} & \{q_0, q_2\} \\
\end{array} \]
\[
\begin{array}{c|c|c}
\text{a} & \text{b} \\
\hline
\{q_0\} & \{q_0, q_1\} & \{q_0\} \\
\{q_0, q_1\} & \{q_0, q_1\} & \{q_0, q_2\} \\
\{q_0, q_2\} & \{q_0, q_1, q_3\} & \{q_0\} \\
\{q_0, q_1, q_3\} & \{q_0, q_1\} & \{q_0, q_2\} \\
\end{array}
\]
\[ q_3 \rightarrow q_3 \rightarrow q_2 \rightarrow a \rightarrow q_3 \]

\[ \{q_0\} \rightarrow a \rightarrow \{q_0, q_1\} \rightarrow a \rightarrow \{q_0, q_2\} \rightarrow a \rightarrow \{q_0, q_1, q_3\} \]
Some Caveats

- **Question**: what about $\varepsilon$-transitions?
  - Answer: always include any states you can reach by following $\varepsilon$-transitions.

- **Question**: what happens if there are no transitions to follow from a set of states for the character you’re trying to fill in?
  - Answer: then the set of states you can reach is the empty set!

- Example included in the appendix of this lecture showing this construction with both of these scenarios.
The Subset Construction

- This construction for transforming an NFA into a DFA is called the **subset construction** (or sometimes the **powerset construction**).
  - Each state in the DFA is associated with a set of states in the NFA.
  - The start state in the DFA corresponds to the start state of the NFA, plus all states reachable via $\varepsilon$-transitions.
  - If a state $q$ in the DFA corresponds to a set of states $S$ in the NFA, then the transition from state $q$ on a character $a$ is found as follows:
    - Let $S'$ be the set of states in the NFA that can be reached by following a transition labeled $a$ from any of the states in $S$. (*This set may be empty.*)
    - Let $S''$ be the set of states in the NFA reachable from some state in $S'$ by following zero or more epsilon transitions.
    - The state $q$ in the DFA transitions on $a$ to a DFA state corresponding to the set of states $S''$.
- **Read Sipser for a formal account.**
The Subset Construction

- For the purposes of this class, we won’t ask you to actually perform the subset construction.
- Hopefully though, you’ve been convinced that, in principle, you could follow this procedure to turn any NFA into a DFA.
The Subset Construction

- In converting an NFA to a DFA, the DFA's states correspond to sets of NFA states.

- **Useful fact:** $|\mathcal{P}(S)| = 2^{|S|}$ for any finite set $S$.

- In the worst-case, the construction can result in a DFA that is *exponentially larger* than the original NFA.

- **Question to ponder:** Can you find a family of languages that have NFAs of size $n$, but no DFAs of size less than $2^n$?
A language $L$ is called a *regular language* if there exists a DFA $D$ such that $\mathcal{L}(D) = L$. 
An Important Result

**Theorem:** A language $L$ is regular iff there is some NFA $N$ such that $\mathcal{L}(N) = L$. 

Proof Sketch: If $L$ is regular, there exists some DFA for it, which we can easily convert into an NFA. If $L$ is accepted by some NFA, we can use the subset construction to convert it into a DFA that accepts the same language, so $L$ is regular. ■
An Important Result

**Theorem:** A language $L$ is regular iff there is some NFA $N$ such that $\mathcal{L}(N) = L$.

**Proof Sketch:**
Theorem: A language $L$ is regular iff there is some NFA $N$ such that $\mathcal{L}(N) = L$.

Proof Sketch: If $L$ is regular, there exists some DFA for it, which we can easily convert into an NFA.
An Important Result

Theorem: A language $L$ is regular iff there is some NFA $N$ such that $\mathcal{L}(N) = L$.

Proof Sketch: If $L$ is regular, there exists some DFA for it, which we can easily convert into an NFA.

If $L$ is accepted by some NFA, we can use the subset construction to convert it into a DFA that accepts the same language, so $L$ is regular.
**An Important Result**

*Theorem:* A language $L$ is regular iff there is some NFA $N$ such that $\mathcal{L}(N) = L$.

*Proof Sketch:* If $L$ is regular, there exists some DFA for it, which we can easily convert into an NFA.

If $L$ is accepted by some NFA, we can use the subset construction to convert it into a DFA that accepts the same language, so $L$ is regular. ■
Why This Matters

- We now have two perspectives on regular languages:
  - Regular languages are languages accepted by DFAs.
  - Regular languages are languages accepted by NFAs.
- We can now reason about the regular languages in two different ways.
Properties of Regular Languages
The Complement of a Language

- Given a language \( L \subseteq \Sigma^* \), the complement of that language (denoted \( \overline{L} \)) is the language of all strings in \( \Sigma^* \) that aren't in \( L \).

- Formally:

\[
\overline{L} = \Sigma^* - L
\]
The Complement of a Language

• Given a language $L \subseteq \Sigma^*$, the complement of that language (denoted $\overline{L}$) is the language of all strings in $\Sigma^*$ that aren't in $L$.

• Formally:

$$\overline{L} = \Sigma^* - L$$
The Complement of a Language

- Given a language $L \subseteq \Sigma^*$, the complement of that language (denoted $\overline{L}$) is the language of all strings in $\Sigma^*$ that aren't in $L$.
- Formally:
  \[
  \overline{L} = \Sigma^* - L
  \]
The Complement of a Language

- Given a language $L \subseteq \Sigma^*$, the *complement* of that language (denoted $\overline{L}$) is the language of all strings in $\Sigma^*$ that aren't in $L$.

- Formally:

$$\overline{L} = \Sigma^* - L$$
The Complement of a Language

- Given a language $L \subseteq \Sigma^*$, the complement of that language (denoted $\overline{L}$) is the language of all strings in $\Sigma^*$ that aren't in $L$.
- Formally:
  
  $$\overline{L} = \Sigma^* - L$$

Good proofwriting exercise: prove $\overline{L} = L$ for any language $L$. 
Complementing Regular Languages

\[ L = \{ w \in \{a, b\}^* \mid w \text{ contains } aa \text{ as a substring} \} \]

\[ \bar{L} = \{ w \in \{a, b\}^* \mid w \text{ does not contain } aa \text{ as a substring} \} \]
Complementing Regular Languages

\[ \bar{L} = \{ w \in \{a, *, /\}^* \mid w \text{ doesn't represent a C-style comment} \} \]
Complementing Regular Languages

\[ \bar{L} = \{ w \in \{a, *, /\}^* \mid w \text{ doesn't represent a C-style comment} \} \]
Closure Properties

- **Theorem**: If $L$ is a regular language, then $\overline{L}$ is also a regular language.
- As a result, we say that the regular languages are **closed under complementation**.

Question to ponder: are the nonregular languages closed under complementation?
The Union of Two Languages

• If $L_1$ and $L_2$ are languages over the alphabet $\Sigma$, the language $L_1 \cup L_2$ is the language of all strings in at least one of the two languages.

• If $L_1$ and $L_2$ are regular languages, is $L_1 \cup L_2$ regular?
The Union of Two Languages

• If $L_1$ and $L_2$ are languages over the alphabet $\Sigma$, the language $L_1 \cup L_2$ is the language of all strings in at least one of the two languages.

• If $L_1$ and $L_2$ are regular languages, is $L_1 \cup L_2$?
The Union of Two Languages

- If $L_1$ and $L_2$ are languages over the alphabet $\Sigma$, the language $L_1 \cup L_2$ is the language of all strings in at least one of the two languages.
- If $L_1$ and $L_2$ are regular languages, is $L_1 \cup L_2$?
The Union of Two Languages

- If $L_1$ and $L_2$ are languages over the alphabet $\Sigma$, the language $L_1 \cup L_2$ is the language of all strings in at least one of the two languages.
- If $L_1$ and $L_2$ are regular languages, is $L_1 \cup L_2$?
The Union of Two Languages

- If $L_1$ and $L_2$ are languages over the alphabet $\Sigma$, the language $L_1 \cup L_2$ is the language of all strings in at least one of the two languages.

- If $L_1$ and $L_2$ are regular languages, is $L_1 \cup L_2$?

*Question to ponder*: where have you seen this idea before?
The Intersection of Two Languages

- If \( L_1 \) and \( L_2 \) are languages over \( \Sigma \), then \( L_1 \cap L_2 \) is the language of strings in both \( L_1 \) and \( L_2 \).

- Question: If \( L_1 \) and \( L_2 \) are regular, is \( L_1 \cap L_2 \) regular as well?
The Intersection of Two Languages

- If $L_1$ and $L_2$ are languages over $\Sigma$, then $L_1 \cap L_2$ is the language of strings in both $L_1$ and $L_2$.

- Question: If $L_1$ and $L_2$ are regular, is $L_1 \cap L_2$ regular as well?
The Intersection of Two Languages

- If $L_1$ and $L_2$ are languages over $\Sigma$, then $L_1 \cap L_2$ is the language of strings in both $L_1$ and $L_2$.
- Question: If $L_1$ and $L_2$ are regular, is $L_1 \cap L_2$ regular as well?
The Intersection of Two Languages

- If $L_1$ and $L_2$ are languages over $\Sigma$, then $L_1 \cap L_2$ is the language of strings in both $L_1$ and $L_2$.

- Question: If $L_1$ and $L_2$ are regular, is $L_1 \cap L_2$ regular as well?

\[ \overline{L_1} \cup \overline{L_2} \]
The Intersection of Two Languages

- If $L_1$ and $L_2$ are languages over $\Sigma$, then $L_1 \cap L_2$ is the language of strings in both $L_1$ and $L_2$.

- Question: If $L_1$ and $L_2$ are regular, is $L_1 \cap L_2$ regular as well?

Hey, it’s De Morgan’s laws!
Concatenation
String Concatenation

• If $w \in \Sigma^*$ and $x \in \Sigma^*$, the **concatenation** of $w$ and $x$, denoted $wx$, is the string formed by tacking all the characters of $x$ onto the end of $w$.

• Example: if $w = \text{quo}$ and $x = \text{kka}$, the concatenation $wx = \text{quokka}$.

• Analogous to the $+$ operator for strings in many programming languages.

• Some facts about concatenation:
  • The empty string $\varepsilon$ is the **identity element** for concatenation:
    \[ w\varepsilon = \varepsilon w = w \]
  • Concatenation is **associative**:
    \[ wxy = w(xy) = (wx)y \]
Concatenation

• The *concatenation* of two languages \( L_1 \) and \( L_2 \) over the alphabet \( \Sigma \) is the language

\[
L_1L_2 = \{ \text{wx} \in \Sigma^* | \text{w} \in L_1 \land \text{x} \in L_2 \}
\]
Concatenation Example

Let $\Sigma = \{ a, b, \ldots, z, A, B, \ldots, Z \}$ and consider these languages over $\Sigma$:

- **Noun** = \{ Puppy, Rainbow, Whale, ... \}
- **Verb** = \{ Hugs, Juggles, Loves, ... \}
- **The** = \{ The \}
- **The** = \{ The \}

The language **TheNounVerbTheNoun** is

\{ ThePuppyHugsTheWhale,
  TheWhaleLovesTheRainbow,
  TheRainbowJugglesTheRainbow, ... \}
Concatenation

• The *concatenation* of two languages $L_1$ and $L_2$ over the alphabet $\Sigma$ is the language

$$L_1L_2 = \{ wx \in \Sigma^* | w \in L_1 \land x \in L_2 \}$$

• Two views of $L_1L_2$:
  • The set of all strings that can be made by concatenating a string in $L_1$ with a string in $L_2$.
  • The set of strings that can be split into two pieces: a piece from $L_1$ and a piece from $L_2$. 
Concatenating Regular Languages

- If $L_1$ and $L_2$ are regular languages, is $L_1L_2$?
- Intuition – can we split a string $w$ into two strings $xy$ such that $x \in L_1$ and $y \in L_2$?
Concatenating Regular Languages

- If $L_1$ and $L_2$ are regular languages, is $L_1L_2$?
- Intuition – can we split a string $w$ into two strings $xy$ such that $x \in L_1$ and $y \in L_2$?

\[\text{Machine for } L_1 \quad \text{start} \quad \begin{array}{c} \rightarrow \text{ } \rightarrow \end{array} \]

\[\text{Machine for } L_2 \quad \text{start} \quad \begin{array}{c} \rightarrow \text{ } \rightarrow \end{array} \]
Concatenating Regular Languages

- If $L_1$ and $L_2$ are regular languages, is $L_1L_2$?
- Intuition – can we split a string $w$ into two strings $xy$ such that $x \in L_1$ and $y \in L_2$?

Machine for $L_1$      Machine for $L_2$

bookkeeper
Concatenating Regular Languages

- If $L_1$ and $L_2$ are regular languages, is $L_1L_2$?
- Intuition – can we split a string $w$ into two strings $xy$ such that $x \in L_1$ and $y \in L_2$?

Machine for $L_1$  

Machine for $L_2$  

bookkeeper
Concatenating Regular Languages

- If $L_1$ and $L_2$ are regular languages, is $L_1L_2$?
- Intuition – can we split a string $w$ into two strings $xy$ such that $x \in L_1$ and $y \in L_2$?

Machine for $L_1$  
Machine for $L_2$

book  
keeper
Concatenating Regular Languages

- If $L_1$ and $L_2$ are regular languages, is $L_1L_2$?
- Intuition – can we split a string $w$ into two strings $xy$ such that $x \in L_1$ and $y \in L_2$?

**Idea:**
- Run a DFA/NFA for $L_1$ on $w$.
- Whenever it reaches an accepting state, optionally hand the rest of $w$ to a DFA/NFA for $L_2$.
- If the automaton for $L_2$ accepts the rest, $w \in L_1L_2$.
- If the automaton for $L_2$ rejects the remainder, the split was incorrect.
Concatenating Regular Languages
Concatenating Regular Languages

Machine for $L_1$
Concatenating Regular Languages

Machine for $L_1$

Machine for $L_2$
Concatenating Regular Languages

Machine for $L_1$

Machine for $L_2$
Concatenating Regular Languages

Machine for $L_1$

Machine for $L_2$
Concatenating Regular Languages

Machine for $L_1$

Machine for $L_2$

Machine for $L_1L_2$
Lots and Lots of Concatenation

- Consider the language $L = \{ \text{aa, b} \}$
- $LL$ is the set of strings formed by concatenating pairs of strings in $L$.

\[ \{ \text{aaaa, aab, baa, bb} \} \]

- $LLL$ is the set of strings formed by concatenating triples of strings in $L$.

\[ \{ \text{aaaaaa, aaaaab, aabaa, aabb, baaaa, baab, bbbaa, bbb} \} \]

- $LLLL$ is the set of strings formed by concatenating quadruples of strings in $L$.

\[ \{ \text{aaaaaaaa, aaaaaaab, aaaaabaa, aaaaabb, aabaaaaa, aabaab, aabbaa, aabbb, baaaaaa, baaaaab, baabaa, baabb, bbaaaa, bbaab, bbbaa, bbbb} \} \]
Language Exponentiation

- We can define what it means to “exponentiate” a language as follows:
  - $L^0 = \{ \varepsilon \}$
    - The set containing just the empty string.
    - Idea: Any string formed by concatenating zero strings together is the empty string.
  - $L^{n+1} = LL^n$
    - Idea: Concatenating $(n+1)$ strings together works by concatenating $n$ strings, then concatenating one more.

- **Question to ponder:** Why define $L^0 = \{ \varepsilon \}$?
- **Question to ponder:** What is $\emptyset^0$?
The Kleene Closure

• An important operation on languages is the **Kleene Closure**, which is defined as

\[ L^* = \{ w \in \Sigma^* \mid \exists n \in \mathbb{N}. w \in L^n \} \]

• Mathematically:

\[ w \in L^* \quad \text{iff} \quad \exists n \in \mathbb{N}. w \in L^n \]

• Intuitively, all possible ways of concatenating zero or more strings in \( L \) together, possibly with repetition.

• **Question to ponder:** What is \( \emptyset^* \)?
The Kleene Closure

If $L = \{ a, bb \}$, then $L^* = \{ \epsilon, a, bb, aa, aabb, abba, abbb, bbba, bbabb, bbbba, bbbbbbb, \ldots \}$

Think of $L^*$ as the set of strings you can make if you have a collection of stamps – one for each string in $L$ – and you form every possible string that can be made from those stamps.
Reasoning about Infinity

• If \( L \) is regular, is \( L^* \) necessarily regular?

• ⚠️ A Bad Line of Reasoning: ⚠️
  • \( L^0 = \{ \epsilon \} \) is regular.
  • \( L^1 = L \) is regular.
  • \( L^2 = LL \) is regular
  • \( L^3 = L(LL) \) is regular
  • ...
  • Regular languages are closed under union.
  • So the union of all these languages is regular.
Reasoning about Infinity
Reasoning about Infinity
Reasoning about Infinity
Reasoning about Infinity
Reasoning about Infinity
Reasoning about Infinity
Reasoning about Infinity

\[ \chi \]

\[ \chi \]
Reasoning about Infinity

\[ x \neq 2x \]
Reasoning about Infinity

\[ 0.9 < 1 \]
Reasoning about Infinity

0.99 < 1
Reasoning about Infinity

0.999 < 1
Reasoning about Infinity

0.9999 < 1
Reasoning about Infinity

\[0.9999\overline{9} < 1\]
Reasoning about Infinity

\[0.9999\overline{9} \neq 1\]
Reasoning about Infinity

0 is finite
Reasoning about Infinity

1 is finite
Reasoning about Infinity

2 is finite
Reasoning about Infinity

3 is finite
Reasoning about Infinity

4 is finite
Reasoning about Infinity

\[ \infty \text{ is finite} \]
Reasoning about Infinity

\[ \infty \text{ is finite} \]

\[ ^\uparrow \text{ not} \]
Reasoning About the Infinite

- If a series of finite objects all have some property, the “limit” of that process does not necessarily have that property.
- In general, it is not safe to conclude that some property that always holds in the finite case must hold in the infinite case.
  - (This is why calculus is interesting).
**Idea:** Can we directly convert an NFA for language $L$ to an NFA for language $L^*$?
The Kleene Star

Machine for $L$
The Kleene Star

Machine for $L$
The Kleene Star

start \rightarrow \varepsilon \rightarrow \text{Machine for } L
The Kleene Star

Machine for $L$
The Kleene Star

Machine for $L$
The Kleene Star

Machine for $L$

Machine for $L^*$
The Kleene Star

Question: Why add the new state out front? Why not just make the old start state accepting?
Closure Properties

- **Theorem:** If $L_1$ and $L_2$ are regular languages over an alphabet $\Sigma$, then so are the following languages:
  - $\overline{L_1}$
  - $L_1 \cup L_2$
  - $L_1 \cap L_2$
  - $L_1L_2$
  - $L_1^*$

- These properties are called **closure properties of the regular languages.**
Next Time

- **Regular Expressions**
  - Building languages from the ground up!
- **Thompson’s Algorithm**
  - A UNIX Programmer in Theoryland.
- **Kleene’s Theorem**
  - From machines to programs!
Appendix: Extended Subset Construction Example
Once More, With Epsilons!
Once More, With Epsilons!
Once More, With Epsilons!
Once More, With Epsilons!
Once More, With Epsilons!

\[
\begin{array}{|c|c|}
\hline
& a & b \\
\hline
\{ q_0, q_3 \} & & \\
\hline
\end{array}
\]
Once More, With Epsilons!
Once More, With Epsilons!

\[
\begin{array}{c}
\text{start} \\
q_0 \\
q_3 \\
q_2 \\
q_1 \\
q_4 \\
\end{array}
\]

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

\[
\begin{array}{c}
\{q_0, q_3\} \\
\end{array}
\]
Once More, With Epsilons!
Once More, With Epsilons!
Once More, With Epsilons!
Once More, With Epsilons!
Once More, With Epsilons!

\[ q_0 \rightarrow q_1 \rightarrow q_4 \]  
\[ q_2 \rightarrow q_3 \]  
\[ \epsilon \rightarrow q_0 \]  
\[ \Sigma \rightarrow q_3 \]  
\[ \Sigma \rightarrow q_4 \]  

\begin{tabular}{|c|c|}
\hline
a & b \\
\hline
\{q_0, q_3\} & \{q_1, q_4\} \\
\hline
\end{tabular}
Once More, With Epsilons!

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>{q_0, q_3}</td>
<td></td>
<td>{q_1, q_4}</td>
</tr>
</tbody>
</table>

![Diagram of a nondeterministic finite automaton](image-url)

- **Start state**: \( q_0 \)
- **States**: \( q_0, q_1, q_3, q_4 \)
- **Transitions**:
  - \( q_0 \xrightarrow{a} q_1 \)
  - \( q_0 \xrightarrow{\epsilon} q_3 \)
  - \( q_3 \xrightarrow{\Sigma} q_4 \)
  - \( q_3 \xrightarrow{\Sigma} q_2 \)
  - \( q_1 \xrightarrow{b} q_4 \)
  - \( q_2 \xrightarrow{b} q_4 \)

- **Accepting states**: \( q_4 \)
Once More, With Epsilons!
Once More, With Epsilons!
Once More, With Epsilons!

\[ q_0 \xrightarrow{\varepsilon} q_3 \]

\[ q_3 \xrightarrow{\Sigma} q_4 \]

\[ q_2 \]

\[ q_1 \]

\[ \Sigma \]

\[ b \]

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>{q_0, q_3}</td>
<td>{q_1, q_4}</td>
</tr>
</tbody>
</table>
Once More, With Epsilons!
Once More, With Epsilons!

\[
\begin{array}{c}
\text{start} \\
q_0 \\
q_3 \\
q_1 \\
q_2 \\
q_4 \\
\end{array}
\]

\[
\begin{array}{c}
\Sigma \\
b \\
\varepsilon \\
\Sigma \\
b \\
\end{array}
\]

\[
\begin{array}{|c|c|}
\hline
\text{a} & \text{b} \\
\hline
\{q_0, q_3\} & \{q_1, q_4\} & \{q_4\} \\
\{q_1, q_4\} & \text{--} & \text{--} \\
\text{--} & \text{--} & \text{--} \\
\text{--} & \text{--} & \text{--} \\
\hline
\end{array}
\]
Once More, With Epsilons!
Once More, With Epsilons!
Once More, With Epsilons!

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>{q_0, q_3}</td>
<td>{q_1, q_4}</td>
<td>{q_4}</td>
</tr>
<tr>
<td>{q_1, q_4}</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Diagram:

- Start state: \(q_0\)
- States: \(q_0, q_1, q_2, q_3, q_4\)
- Transitions:
  - \(q_0\) on \(a\) to \(q_1\)
  - \(q_0\) on \(\epsilon\) to \(q_3\)
  - \(q_3\) on \(b\) to \(q_4\)
  - \(q_1\) on \(b\) to \(q_4\)
  - \(q_1\) on \(\Sigma\) to \(q_2\)
  - \(q_2\) on \(\Sigma\) to \(q_1\)
  - \(q_2\) on \(b\) to \(q_3\)
  - \(q_3\) on \(\Sigma\) to \(q_4\)
  - \(q_4\) on \(\Sigma\) to \(q_3\)
Once More, With Epsilons!

\[
\begin{array}{ccc}
\text{a} & \text{b} \\
\{q_0, q_3\} & \{q_1, q_4\} & \{q_4\} \\
\{q_1, q_4\} & & \\
& & \\
& & \\
& & \\
\end{array}
\]
Once More, With Epsilons!
Once More, With Epsilons!
Once More, With Epsilons!
Once More, With Epsilons!

\[
\begin{array}{c|cc}
q_0 & a & b \\
\hline
\{q_0, q_3\} & \{q_1, q_4\} & \{q_4\} \\
\{q_1, q_4\} & \emptyset & \\
\end{array}
\]
Once More, With Epsilons!

\[
\begin{array}{c|c|c}
\text{ } & \text{a} & \text{b} \\
\hline
\{q_0, q_3\} & \{q_1, q_4\} & \{q_4\} \\
\{q_1, q_4\} & \emptyset & \text{ } \\
\hline
\end{array}
\]
Once More, With Epsilons!

```
<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>{q_0, q_3}</td>
<td>{q_1, q_4}</td>
<td>{q_4}</td>
</tr>
<tr>
<td>{q_1, q_4}</td>
<td>Ø</td>
<td></td>
</tr>
</tbody>
</table>
```

- Start state: \(q_0\)
- Transitions:
  - \(q_0\) on \(a\) to \(q_1\)
  - \(q_0\) on \(\epsilon\) to \(q_3\)
  - \(q_1\) on \(b\) to \(q_4\)
  - \(q_3\) on \(b\) to \(q_4\)
  - \(q_2\) on \(\Sigma\) to \(q_1\)
  - \(q_2\) on \(b\) to \(q_4\)
Once More, With Epsilons!

\[
\begin{array}{cccc}
\text{a} & \text{b} \\
\{q_0, q_3\} & \{q_1, q_4\} & \{q_4\} \\
\{q_1, q_4\} & \emptyset & \emptyset \\
\end{array}
\]
Once More, With Epsilons!
Once More, With Epsilons!

\[
\begin{array}{c}
\text{start} \\
\begin{array}{c}
q_0 \\
q_1 \\
q_3 \\
q_4 \\
q_2 \\
\end{array}
\end{array}
\begin{array}{cccc}
\text{a} & \text{b} \\
\{q_0, q_3\} & \{q_1, q_4\} & \{q_4\} \\
\{q_1, q_4\} & \emptyset & \{q_2, q_3\} \\
\end{array}
\]

\[
\begin{array}{c}
\Sigma \\
\Sigma \\
\end{array}
\begin{array}{c}
a \\
b \\
\end{array}
\begin{array}{c}
\epsilon \\
\epsilon \\
\end{array}
\]
Once More, With Epsilons!

\[
\begin{array}{c}
\text{start} \\
q_0 \xrightarrow{a} q_1 \\
q_3 \xrightarrow{\varepsilon} \xrightarrow{\Sigma} q_4 \\
q_2 \xrightarrow{\Sigma} q_0 \\
q_1 \xrightarrow{b} q_4
\end{array}
\]

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>{q_0, q_3}</td>
<td>{q_1, q_4}</td>
<td>{q_4}</td>
</tr>
<tr>
<td>{q_1, q_4}</td>
<td>\Ø</td>
<td>{q_2, q_3}</td>
</tr>
<tr>
<td>{q_4}</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Once More, With Epsilons!

\[
\begin{align*}
\text{start} & \quad \epsilon & \quad \{q_0, q_3\} & \quad \{q_1, q_4\} & \quad \{q_4\} \\
q_0 & \quad a & \quad \{q_1, q_4\} & \quad \emptyset & \quad \{q_2, q_3\} \\
q_3 & \quad \epsilon & \quad \{q_4\} & \quad \emptyset & \quad \emptyset \\
q_2 & \quad \Sigma & \quad \emptyset & \quad \emptyset & \quad \emptyset \\
q_1 & \quad b & \quad \emptyset & \quad \emptyset & \quad \emptyset \\
q_4 & \quad \emptyset & \quad \emptyset & \quad \emptyset & \quad \emptyset
\end{align*}
\]
Once More, With Epsilons!
Once More, With Epsilons!
Once More, With Epsilons!

\[ q_0 \xrightarrow{a} q_1 \]
\[ q_0 \xrightarrow{\varepsilon} q_4 \]
\[ q_2 \xrightarrow{\Sigma} q_1 \]
\[ q_3 \xrightarrow{\Sigma} q_4 \]

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>{q_0, q_3}</td>
<td>{q_1, q_4}</td>
<td>{q_4}</td>
</tr>
<tr>
<td>{q_1, q_4}</td>
<td>\emptyset</td>
<td>{q_2, q_3}</td>
</tr>
<tr>
<td>{q_4}</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Once More, With Epsilons!

$$\begin{array}{c}
\begin{array}{ccc}
\text{start} & \longrightarrow & q_0 \\
& & a \\
\downarrow & & \downarrow \Sigma \\
q_3 & \longrightarrow & q_1 \\
& & b \\
\downarrow & & \uparrow \Sigma \\
\varepsilon & & q_2 \\
\longrightarrow & & b \\
\downarrow & & \leftarrow \Sigma \end{array}
\end{array}$$

$$\begin{array}{|c|c|c|}
\hline
& \text{a} & \text{b} \\
\hline
\{q_0, q_3\} & \{q_1, q_4\} & \{q_4\} \\
\hline
\{q_1, q_4\} & \emptyset & \{q_2, q_3\} \\
\hline
\{q_4\} & \emptyset & \emptyset \\
\hline
\end{array}$$
Once More, With Epsilons!

\[
\begin{array}{c}
\text{start} \\
q_0 \xrightarrow{\varepsilon} q_3 \xrightarrow{\Sigma} q_4 \\
q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2
\end{array}
\]

\[
\begin{array}{|c|c|c|}
\hline
& a & b \\
\hline
\{q_0, q_3\} & \{q_1, q_4\} & \{q_4\} \\
\{q_1, q_4\} & \emptyset & \{q_2, q_3\} \\
\{q_4\} & \emptyset & \emptyset \\
\hline
\end{array}
\]
Once More, With Epsilons!

\[
\begin{array}{ccc}
\{q_0, q_3\} & \{q_1, q_4\} & \{q_4\} \\
\{q_1, q_4\} & \emptyset & \{q_2, q_3\} \\
\{q_4\} & \emptyset & \\
\end{array}
\]
Once More, With Epsilons!
Once More, With Epsilons!
Once More, With Epsilons!
Once More, With Epsilons!
Once More, With Epsilons!

\[ q_4 \]

\[ \varepsilon \]

\[ \{ q_0, q_3 \} \]

\[ \{ q_1, q_4 \} \]

\[ \{ q_4 \} \]

\[ \emptyset \]

\[ \{ q_2, q_3 \} \]

\[ \{ q_3 \} \]

\[ a \]

\[ b \]

\[ \Sigma \]

\[ \text{chart} \]
Once More, With Epsilons!

```
\begin{align*}
\text{start} & \quad \varepsilon \quad \circ \quad \Sigma \quad b \\
q_0 & \quad a \quad \quad q_1 \\
q_3 & \quad \varepsilon \quad \Sigma \quad b \\
q_4 &
\end{align*}
```

<table>
<thead>
<tr>
<th></th>
<th>(a)</th>
<th>(b)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(q_0, q_3)</td>
<td>(q_1, q_4)</td>
<td>(q_4)</td>
</tr>
<tr>
<td>(q_1, q_4)</td>
<td>(\emptyset)</td>
<td>(q_2, q_3)</td>
</tr>
<tr>
<td>(q_4)</td>
<td>(\emptyset)</td>
<td>(q_3)</td>
</tr>
<tr>
<td>(q_2, q_3)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[\Sigma\]
Once More, With Epsilons!

\[
\begin{array}{c|c|c}
 & a & b \\
\hline
\{q_0, q_3\} & \{q_1, q_4\} & \{q_4\} \\
\{q_1, q_4\} & \emptyset & \{q_2, q_3\} \\
\{q_4\} & \emptyset & \{q_3\} \\
\{q_2, q_3\} & & \\
\end{array}
\]
Once More, With Epsilons!
Once More, With Epsilons!

\[
\begin{array}{cccc}
\text{start} & q_0 & q_1 & q_2 \\
\varepsilon & a & b & \text{a} \\
\sum & b & \sum & \sum \\
q_3 & q_4 & q_4 & \emptyset \\
\{q_0, q_3\} & \{q_1, q_4\} & \{q_4\} & \{q_4\} \\
\{q_1, q_4\} & \emptyset & \{q_2, q_3\} & \emptyset \\
\{q_4\} & \emptyset & \{q_3\} & \emptyset \\
\{q_2, q_3\} & \emptyset & \emptyset & \emptyset \\
\end{array}
\]
Once More, With Epsilons!
Once More, With Epsilons!

**Diagram:**

- **Start State:** $q_0$
- **Transitions:**
  - $q_0$ to $q_1$: $a$
  - $q_0$ to $q_3$: $\varepsilon$
  - $q_2$ to $q_1$: $\Sigma$
  - $q_3$ to $q_4$: $b$
  - $q_2$ to $q_4$: $b$

**Table:**

<table>
<thead>
<tr>
<th>States</th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>${q_0, q_3}$</td>
<td>${q_1, q_4}$</td>
<td>${q_4}$</td>
</tr>
<tr>
<td>${q_1, q_4}$</td>
<td>Ø</td>
<td>${q_2, q_3}$</td>
</tr>
<tr>
<td>${q_4}$</td>
<td>Ø</td>
<td>${q_3}$</td>
</tr>
<tr>
<td>${q_2, q_3}$</td>
<td>Ø</td>
<td>Ø</td>
</tr>
</tbody>
</table>
Once More, With Epsilons!

**Diagram:**
- Start state: $q_0$
- States: $q_0, q_1, q_2, q_3, q_4$
- Transitions:
  - $q_0 \xrightarrow{a} q_1$
  - $q_0 \xrightarrow{\epsilon} q_3$
  - $q_3 \xrightarrow{b} q_4$
  - $q_1 \xrightarrow{b} q_4$

**Table:***

<table>
<thead>
<tr>
<th></th>
<th>$a$</th>
<th>$b$</th>
</tr>
</thead>
<tbody>
<tr>
<td>${q_0, q_3}$</td>
<td>${q_1, q_4}$</td>
<td>${q_4}$</td>
</tr>
<tr>
<td>${q_1, q_4}$</td>
<td>$\emptyset$</td>
<td>${q_2, q_3}$</td>
</tr>
<tr>
<td>${q_4}$</td>
<td>$\emptyset$</td>
<td>${q_3}$</td>
</tr>
<tr>
<td>${q_2, q_3}$</td>
<td>${q_0, q_3, q_4}$</td>
<td>${q_0, q_3, q_4}$</td>
</tr>
</tbody>
</table>
Once More, With Epsilons!
Once More, With Epsilons!

![Diagram of a finite automaton with states and transitions.]

### Transition Table

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>{q_0, q_3}</td>
<td>{q_1, q_4}</td>
<td>{q_4}</td>
</tr>
<tr>
<td>{q_1, q_4}</td>
<td>Ø</td>
<td>{q_2, q_3}</td>
</tr>
<tr>
<td>{q_4}</td>
<td>Ø</td>
<td>{q_3}</td>
</tr>
<tr>
<td>{q_2, q_3}</td>
<td>{q_0, q_3, q_4}</td>
<td>{q_0, q_3, q_4}</td>
</tr>
<tr>
<td>{q_3}</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Once More, With Epsilons!

\[
\begin{array}{c}
\text{start} \quad \epsilon \quad \{q_0, q_3\} \\
q_0 \quad a \quad \{q_1, q_4\} \\
q_1 \quad \emptyset \quad \{q_2, q_3\} \\
q_2 \quad \Sigma \quad \emptyset \quad \{q_3\} \\
q_3 \quad \Sigma \quad \{q_0, q_3, q_4\} \quad \{q_0, q_3, q_4\} \\
q_4 \quad b \quad \emptyset \quad \emptyset
\end{array}
\]
Once More, With Epsilons!
Once More, With Epsilons!

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>${q_0, q_3}$</td>
<td>${q_1, q_4}$</td>
<td>${q_4}$</td>
</tr>
<tr>
<td>${q_1, q_4}$</td>
<td>$\emptyset$</td>
<td>${q_2, q_3}$</td>
</tr>
<tr>
<td>${q_4}$</td>
<td>$\emptyset$</td>
<td>${q_3}$</td>
</tr>
<tr>
<td>${q_2, q_3}$</td>
<td>${q_0, q_3, q_4}$</td>
<td>${q_0, q_3, q_4}$</td>
</tr>
<tr>
<td>${q_3}$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Diagram:

- Start state: $q_0$
- Final states: $q_1$, $q_4$
- Transitions:
  - $q_0 \xrightarrow{a} q_1$
  - $q_0 \xrightarrow{\Sigma} q_2$
  - $q_1 \xrightarrow{\Sigma} q_4$
  - $q_2 \xrightarrow{\Sigma} q_3$
  - $q_3 \xrightarrow{b} q_4$
  - $q_3 \xrightarrow{\varepsilon} q_0$
  - $q_2 \xrightarrow{b} q_4$
Once More, With Epsilons!

\[ q_0 \xrightarrow{\varepsilon} q_3 \]
\[ q_3 \xrightarrow{\Sigma} q_4 \]
\[ q_4 \xrightarrow{b} q_1 \]
\[ q_1 \xrightarrow{a} q_0 \]
\[ q_0 \xrightarrow{\Sigma} q_2 \]

Transition Table:

<table>
<thead>
<tr>
<th>Input</th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>{q_0, q_3}</td>
<td>{q_1, q_4}</td>
<td>{q_4}</td>
</tr>
<tr>
<td>{q_1, q_4}</td>
<td>\emptyset</td>
<td>{q_2, q_3}</td>
</tr>
<tr>
<td>{q_4}</td>
<td>\emptyset</td>
<td>{q_3}</td>
</tr>
<tr>
<td>{q_2, q_3}</td>
<td>{q_0, q_3, q_4}</td>
<td>{q_0, q_3, q_4}</td>
</tr>
<tr>
<td>{q_3}</td>
<td>\emptyset</td>
<td>\emptyset</td>
</tr>
</tbody>
</table>
Once More, With Epsilons!

\[
\begin{array}{ccc}
\text{a} & \text{b} \\
\{q_0, q_3\} & \{q_1, q_4\} & \{q_4\} \\
\{q_1, q_4\} & \emptyset & \{q_2, q_3\} \\
\{q_4\} & \emptyset & \{q_3\} \\
\{q_2, q_3\} & \{q_0, q_3, q_4\} & \{q_0, q_3, q_4\} \\
\{q_3\} & \{q_4\} & \{q_4\} \\
\end{array}
\]
Once More, With Epsilons!
Once More, With Epsilons!

Table:

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>{q_0, q_3}</td>
<td>{q_1, q_4}</td>
<td>{q_4}</td>
</tr>
<tr>
<td>{q_1, q_4}</td>
<td>\emptyset</td>
<td>{q_2, q_3}</td>
</tr>
<tr>
<td>{q_4}</td>
<td>\emptyset</td>
<td>{q_3}</td>
</tr>
<tr>
<td>{q_2, q_3}</td>
<td>{q_0, q_3, q_4}</td>
<td>{q_0, q_3, q_4}</td>
</tr>
<tr>
<td>{q_3}</td>
<td>{q_4}</td>
<td>{q_4}</td>
</tr>
<tr>
<td>{q_0, q_3, q_4}</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Diagram:

- Start at \(q_0\).
- \(q_0\) on \(\varepsilon\) to \(q_3\).
- \(q_0\) on \(a\) to \(q_1\).
- \(q_1\) on \(b\) to \(q_4\).
- \(q_3\) on \(b\) to \(q_4\).
Once More, With Epsilons!

**Diagram:**
- **Start State:** $q_0$
- **Transitions:**
  - $q_0 \xrightarrow{a} q_1$
  - $q_0 \xrightarrow{\varepsilon} q_3$
  - $q_3 \xrightarrow{b} q_4$
  - $q_2 \xrightarrow{\Sigma} q_1$

**Transition Table:**

<table>
<thead>
<tr>
<th></th>
<th>$a$</th>
<th>$b$</th>
</tr>
</thead>
<tbody>
<tr>
<td>${q_0, q_3}$</td>
<td>${q_1, q_4}$</td>
<td>${q_4}$</td>
</tr>
<tr>
<td>${q_1, q_4}$</td>
<td>$\emptyset$</td>
<td>${q_2, q_3}$</td>
</tr>
<tr>
<td>${q_4}$</td>
<td>$\emptyset$</td>
<td>${q_3}$</td>
</tr>
<tr>
<td>${q_2, q_3}$</td>
<td>${q_0, q_3, q_4}$</td>
<td>${q_0, q_3, q_4}$</td>
</tr>
<tr>
<td>${q_3}$</td>
<td>${q_4}$</td>
<td>${q_4}$</td>
</tr>
<tr>
<td>${q_0, q_3, q_4}$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Once More, With Epsilons!
Once More, With Epsilons!

\[ \begin{align*}
q_0 & \rightarrow q_1 & b & \rightarrow \{ q_1, q_4 \} & \{ q_4 \} \\
q_2 & \rightarrow q_1 & b & \rightarrow \emptyset & \{ q_2, q_3 \} \\
q_1 & \rightarrow q_1 & b & \rightarrow \emptyset & \{ q_3 \} \\
q_3 & \rightarrow q_3 & b & \rightarrow \{ q_0, q_3, q_4 \} & \{ q_0, q_3, q_4 \} \\
q_4 & \rightarrow q_4 & b & \rightarrow \emptyset & \{ q_4 \} \\
q_0, q_3, q_4 & \rightarrow q_4 & b & \rightarrow \emptyset & \{ q_4 \}
\end{align*} \]
Once More, With Epsilons!
Once More, With Epsilons!

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>{q_0, q_3}</td>
<td>{q_1, q_4}</td>
<td>{q_4}</td>
</tr>
<tr>
<td>{q_1, q_4}</td>
<td>Ø</td>
<td>{q_2, q_3}</td>
</tr>
<tr>
<td>{q_4}</td>
<td>Ø</td>
<td>{q_3}</td>
</tr>
<tr>
<td>{q_2, q_3}</td>
<td>{q_0, q_3, q_4}</td>
<td>{q_0, q_3, q_4}</td>
</tr>
<tr>
<td>{q_3}</td>
<td>{q_4}</td>
<td>{q_4}</td>
</tr>
<tr>
<td>{q_0, q_3, q_4}</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Once More, With Epsilons!

**Diagram:**
- **Start state:** $q_0$
- **States:** $q_0, q_1, q_2, q_3, q_4$
- **Transition Labels:**
  - $a$: $\{q_0, q_3\} \rightarrow \{q_1, q_4\}$
  - $b$: $\{q_1, q_4\} \rightarrow \{q_0, q_3\}$
  - $\varepsilon$: $\{q_3\} \rightarrow \{q_0, q_3, q_4\}$

**Table:**

<table>
<thead>
<tr>
<th>State</th>
<th>$a$</th>
<th>$b$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q_0$</td>
<td>${q_1, q_4}$</td>
<td>${q_4}$</td>
</tr>
<tr>
<td>$q_1$</td>
<td>$\emptyset$</td>
<td>${q_2, q_3}$</td>
</tr>
<tr>
<td>$q_2$</td>
<td>${q_4}$</td>
<td>$\emptyset$</td>
</tr>
<tr>
<td>$q_3$</td>
<td>${q_0, q_3, q_4}$</td>
<td>${q_0, q_3, q_4}$</td>
</tr>
<tr>
<td>$q_4$</td>
<td>${q_4}$</td>
<td>${q_4}$</td>
</tr>
<tr>
<td>${q_0, q_3, q_4}$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Once More, With Epsilons!

\[
\begin{array}{c}
\text{start} \\
q_0 \quad q_2 \quad q_1 \quad q_3 \quad q_4
\end{array}
\]

\[
\begin{array}{c}
\Sigma \quad \epsilon \\
a \quad b
\end{array}
\]

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>{q_0, q_3}</td>
<td>{q_1, q_4}</td>
<td>{q_4}</td>
</tr>
<tr>
<td>{q_1, q_4}</td>
<td>\emptyset</td>
<td>{q_2, q_3}</td>
</tr>
<tr>
<td>{q_4}</td>
<td>\emptyset</td>
<td>{q_3}</td>
</tr>
<tr>
<td>{q_2, q_3}</td>
<td>{q_0, q_3, q_4}</td>
<td>{q_0, q_3, q_4}</td>
</tr>
<tr>
<td>{q_3}</td>
<td>{q_4}</td>
<td>{q_4}</td>
</tr>
<tr>
<td>{q_0, q_3, q_4}</td>
<td>{q_1, q_4}</td>
<td></td>
</tr>
</tbody>
</table>
Once More, With Epsilons!

<table>
<thead>
<tr>
<th>States</th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>{q_0, q_3}</td>
<td>{q_1, q_4}</td>
<td>{q_4}</td>
</tr>
<tr>
<td>{q_1, q_4}</td>
<td>\emptyset</td>
<td>{q_2, q_3}</td>
</tr>
<tr>
<td>{q_4}</td>
<td>\emptyset</td>
<td>{q_3}</td>
</tr>
<tr>
<td>{q_2, q_3}</td>
<td>{q_0, q_3, q_4}</td>
<td>{q_0, q_3, q_4}</td>
</tr>
<tr>
<td>{q_3}</td>
<td>{q_4}</td>
<td>{q_4}</td>
</tr>
<tr>
<td>{q_0, q_3, q_4}</td>
<td>{q_1, q_4}</td>
<td></td>
</tr>
</tbody>
</table>
Once More, With Epsilons!
Once More, With Epsilons!
Once More, With Epsilons!

\[
\begin{array}{cccc}
\text{start} & q_0 & q_1 & q_2 \\
\epsilon & a & \Sigma & b \\
q_3 & b & \Sigma & \\
\end{array}
\]

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>{q_0, q_3}</td>
<td>{q_1, q_4}</td>
<td>{q_4}</td>
</tr>
<tr>
<td>{q_1, q_4}</td>
<td>\emptyset</td>
<td>{q_2, q_3}</td>
</tr>
<tr>
<td>{q_4}</td>
<td>\emptyset</td>
<td>{q_3}</td>
</tr>
<tr>
<td>{q_2, q_3}</td>
<td>{q_0, q_3, q_4}</td>
<td>{q_0, q_3, q_4}</td>
</tr>
<tr>
<td>{q_3}</td>
<td>{q_4}</td>
<td>{q_4}</td>
</tr>
<tr>
<td>{q_0, q_3, q_4}</td>
<td>{q_1, q_4}</td>
<td></td>
</tr>
</tbody>
</table>
\]
Once More, With Epsilons!
Once More, With Epsilons!
Once More, With Epsilons!

\begin{itemize}
  \item \textbf{Start State:} $q_0$
  \item \textbf{Transitions:}
    \begin{itemize}
      \item $q_0 \xrightarrow{\varepsilon} q_3$
      \item $q_0 \xrightarrow{a} q_1$
      \item $q_0 \xrightarrow{b} q_4$
      \item $q_2 \xrightarrow{\Sigma} q_0$
      \item $q_2 \xrightarrow{b} q_4$
      \item $q_1 \xrightarrow{a} q_0$
      \item $q_1 \xrightarrow{b} q_4$
      \item $q_3 \xrightarrow{b} q_4$
    \end{itemize}
\end{itemize}

\begin{table}[h]
\begin{tabular}{|c|c|c|}
\hline
   & $a$ & $b$ \\
\hline
$\{q_0, q_3\}$ & $\{q_1, q_4\}$ & $\{q_4\}$ \\
$\{q_1, q_4\}$ & $\emptyset$ & $\{q_2, q_3\}$ \\
$\{q_4\}$ & $\emptyset$ & $\{q_3\}$ \\
$\{q_2, q_3\}$ & $\{q_0, q_3, q_4\}$ & $\{q_0, q_3, q_4\}$ \\
$\{q_3\}$ & $\{q_4\}$ & $\{q_4\}$ \\
$\{q_0, q_3, q_4\}$ & $\{q_1, q_4\}$ & $\{q_3, q_4\}$ \\
\hline
\end{tabular}
\end{table}
Once More, With Epsilons!

The diagram shows a finite automaton with states $q_0, q_1, q_2, q_3,$ and $q_4$. The labels on the transitions indicate the input symbols that move the automaton from one state to another:

- $q_0$ is the start state.
- $q_4$ is the only accepting state.
- The transition from $q_0$ to $q_1$ is labeled with $a$.
- The transition from $q_0$ to $q_3$ is labeled with $\epsilon$.
- The transition from $q_1$ to $q_2$ is labeled with $a$.
- The transition from $q_1$ to $q_4$ is labeled with $b$.
- The transition from $q_2$ to $q_3$ is labeled with $b$.
- The transition from $q_3$ to $q_4$ is labeled with $b$.

The table below lists the transitions for input symbols $a$ and $b$:

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>${q_0, q_3}$</td>
<td>${q_1, q_4}$</td>
<td>${q_4}$</td>
</tr>
<tr>
<td>${q_1, q_4}$</td>
<td>$\emptyset$</td>
<td>${q_2, q_3}$</td>
</tr>
<tr>
<td>${q_4}$</td>
<td>$\emptyset$</td>
<td>${q_3}$</td>
</tr>
<tr>
<td>${q_2, q_3}$</td>
<td>${q_0, q_3, q_4}$</td>
<td>${q_0, q_3, q_4}$</td>
</tr>
<tr>
<td>${q_3}$</td>
<td>${q_4}$</td>
<td>${q_4}$</td>
</tr>
<tr>
<td>${q_0, q_3, q_4}$</td>
<td>${q_1, q_4}$</td>
<td>${q_3, q_4}$</td>
</tr>
</tbody>
</table>
Once More, With Epsilons!

<table>
<thead>
<tr>
<th>States</th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>{q_0, q_3}</td>
<td>{q_1, q_4}</td>
<td>{q_4}</td>
</tr>
<tr>
<td>{q_1, q_4}</td>
<td>\emptyset</td>
<td>{q_2, q_3}</td>
</tr>
<tr>
<td>{q_4}</td>
<td>\emptyset</td>
<td>{q_3}</td>
</tr>
<tr>
<td>{q_2, q_3}</td>
<td>{q_0, q_3, q_4}</td>
<td>{q_0, q_3, q_4}</td>
</tr>
<tr>
<td>{q_3}</td>
<td>{q_4}</td>
<td>{q_4}</td>
</tr>
<tr>
<td>{q_0, q_3, q_4}</td>
<td>{q_1, q_4}</td>
<td>{q_3, q_4}</td>
</tr>
<tr>
<td>{q_3, q_4}</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Diagram:

- Start state: \( q_0 \)
- Transitions:
  - \( q_0 \) on \( a \) to \( q_1 \)
  - \( q_0 \) on \( \epsilon \) to \( q_3 \)
  - \( q_1 \) on \( b \) to \( q_4 \)
  - \( q_3 \) on \( b \) to \( q_4 \)
Once More, With Epsilons!
Once More, With Epsilons!

<table>
<thead>
<tr>
<th>States</th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>{q_0, q_3}</td>
<td>{q_1, q_4}</td>
<td>{q_4}</td>
</tr>
<tr>
<td>{q_1, q_4}</td>
<td>\emptyset</td>
<td>{q_2, q_3}</td>
</tr>
<tr>
<td>{q_4}</td>
<td>\emptyset</td>
<td>{q_3}</td>
</tr>
<tr>
<td>{q_2, q_3}</td>
<td>{q_0, q_3, q_4}</td>
<td>{q_0, q_3, q_4}</td>
</tr>
<tr>
<td>{q_3}</td>
<td>{q_4}</td>
<td>{q_4}</td>
</tr>
<tr>
<td>{q_0, q_3, q_4}</td>
<td>{q_1, q_4}</td>
<td>{q_3, q_4}</td>
</tr>
<tr>
<td>{q_3, q_4}</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Once More, With Epsilons!

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>{q_0, q_3}</td>
<td>{q_1, q_4}</td>
<td>{q_4}</td>
</tr>
<tr>
<td>{q_1, q_4}</td>
<td>\Ø</td>
<td>{q_2, q_3}</td>
</tr>
<tr>
<td>{q_4}</td>
<td>\Ø</td>
<td>{q_3}</td>
</tr>
<tr>
<td>{q_2, q_3}</td>
<td>{q_0, q_3, q_4}</td>
<td>{q_0, q_3, q_4}</td>
</tr>
<tr>
<td>{q_3}</td>
<td>{q_4}</td>
<td>{q_4}</td>
</tr>
<tr>
<td>{q_0, q_3, q_4}</td>
<td>{q_1, q_4}</td>
<td>{q_3, q_4}</td>
</tr>
<tr>
<td>{q_3, q_4}</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Once More, With Epsilons!
Once More, With Epsilons!
Once More, With Epsilons!
Once More, With Epsilons!

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>{q_0, q_3}</td>
<td>{q_1, q_4}</td>
<td>{q_4}</td>
</tr>
<tr>
<td>{q_1, q_4}</td>
<td>\ø</td>
<td>{q_2, q_3}</td>
</tr>
<tr>
<td>{q_4}</td>
<td>\ø</td>
<td>{q_3}</td>
</tr>
<tr>
<td>{q_2, q_3}</td>
<td>{q_0, q_3, q_4}</td>
<td>{q_0, q_3, q_4}</td>
</tr>
<tr>
<td>{q_3}</td>
<td>{q_4}</td>
<td>{q_4}</td>
</tr>
<tr>
<td>{q_0, q_3, q_4}</td>
<td>{q_1, q_4}</td>
<td>{q_3, q_4}</td>
</tr>
<tr>
<td>{q_3, q_4}</td>
<td>{q_4}</td>
<td>\ø</td>
</tr>
</tbody>
</table>

Diagram:

- Start state: \(q_0\)
- States: \(q_0, q_1, q_2, q_3, q_4\)
- Transitions:
  - \(q_0\) to \(q_3\) with \(\varepsilon\)
  - \(q_0\) to \(q_1\) with \(a\)
  - \(q_1\) to \(q_4\) with \(b\)
  - \(q_2\) to \(q_0\) with \(\Sigma\)
  - \(q_2\) to \(q_3\) with \(b\)
  - \(q_3\) to \(q_4\) with \(b\)
Once More, With Epsilons!

\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|}
\hline
 & a & b \\
\hline
\{q_0, q_3\} & \{q_1, q_4\} & \{q_4\} \\
\{q_1, q_4\} & \emptyset & \{q_2, q_3\} \\
\{q_4\} & \emptyset & \{q_3\} \\
\{q_2, q_3\} & \{q_0, q_3, q_4\} & \{q_0, q_3, q_4\} \\
\{q_3\} & \{q_4\} & \{q_4\} \\
\{q_0, q_3, q_4\} & \{q_1, q_4\} & \{q_3, q_4\} \\
\{q_3, q_4\} & \{q_4\} & \{q_4\} \\
\hline
\end{tabular}
\end{table}
Once More, With Epsilons!
Once More, With Epsilons!
Once More, With Epsilons!

---

**Diagram:**
- **Start state:** $q_0$
- **States:** $q_0, q_1, q_2, q_3, q_4$
- **Transitions:**
  - $q_0 \xrightarrow{a} q_1$
  - $q_0 \xrightarrow{\varepsilon} q_3$
  - $q_3 \xrightarrow{b} q_4$

**Transition Table**:

<table>
<thead>
<tr>
<th></th>
<th>$a$</th>
<th>$b$</th>
</tr>
</thead>
<tbody>
<tr>
<td>${q_0, q_3}$</td>
<td>${q_1, q_4}$</td>
<td>${q_4}$</td>
</tr>
<tr>
<td>${q_1, q_4}$</td>
<td>$\emptyset$</td>
<td>${q_2, q_3}$</td>
</tr>
<tr>
<td>${q_4}$</td>
<td>$\emptyset$</td>
<td>${q_3}$</td>
</tr>
<tr>
<td>${q_2, q_3}$</td>
<td>${q_0, q_3, q_4}$</td>
<td>${q_0, q_3, q_4}$</td>
</tr>
<tr>
<td>${q_3}$</td>
<td>${q_4}$</td>
<td>${q_4}$</td>
</tr>
<tr>
<td>${q_0, q_3, q_4}$</td>
<td>${q_1, q_4}$</td>
<td>${q_3, q_4}$</td>
</tr>
<tr>
<td>${q_3, q_4}$</td>
<td>${q_4}$</td>
<td>${q_4}$</td>
</tr>
</tbody>
</table>
Once More, With Epsilons!
Once More, With Epsilons!
Once More, With Epsilons!
Once More, With Epsilons!

![Diagram of a finite automaton]

<table>
<thead>
<tr>
<th>State Set</th>
<th>Transition on a</th>
<th>Transition on b</th>
</tr>
</thead>
<tbody>
<tr>
<td>{q_0, q_3}</td>
<td>{q_1, q_4}</td>
<td>{q_4}</td>
</tr>
<tr>
<td>{q_1, q_4}</td>
<td>\emptyset</td>
<td>{q_2, q_3}</td>
</tr>
<tr>
<td>{q_4}</td>
<td>\emptyset</td>
<td>{q_3}</td>
</tr>
<tr>
<td>{q_2, q_3}</td>
<td>{q_0, q_3, q_4}</td>
<td>{q_0, q_3, q_4}</td>
</tr>
<tr>
<td>{q_3}</td>
<td>{q_4}</td>
<td>{q_4}</td>
</tr>
<tr>
<td>{q_0, q_3, q_4}</td>
<td>{q_1, q_4}</td>
<td>{q_3, q_4}</td>
</tr>
<tr>
<td>{q_3, q_4}</td>
<td>{q_4}</td>
<td>{q_3, q_4}</td>
</tr>
<tr>
<td>\emptyset</td>
<td>\emptyset</td>
<td>\emptyset</td>
</tr>
</tbody>
</table>
Once More, With Epsilons!
Once More, With Epsilons!
Once More, With Epsilons!

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>{*q_0, q_3}</td>
<td>{q_1, q_4}</td>
<td>{q_4}</td>
</tr>
<tr>
<td>{*q_1, q_4}</td>
<td>Ø</td>
<td>{q_2, q_3}</td>
</tr>
<tr>
<td>{q_4}</td>
<td>Ø</td>
<td>{q_3}</td>
</tr>
<tr>
<td>{*q_2, q_3}</td>
<td>{q_0, q_3, q_4}</td>
<td>{q_0, q_3, q_4}</td>
</tr>
<tr>
<td>{*q_3}</td>
<td>{q_4}</td>
<td>{q_4}</td>
</tr>
<tr>
<td>{*q_0, q_3, q_4}</td>
<td>{q_1, q_4}</td>
<td>{q_3, q_4}</td>
</tr>
<tr>
<td>{*q_3, q_4}</td>
<td>{q_4}</td>
<td>{q_3, q_4}</td>
</tr>
<tr>
<td>Ø</td>
<td>Ø</td>
<td>Ø</td>
</tr>
</tbody>
</table>