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 Σ.
- For each state in the DFA, there must be exactly one transition defined for each symbol in Σ.
  - 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!
Hello, NFA!

\[ q_0 \rightarrow h \rightarrow q_1 \rightarrow i \rightarrow q_2 \]
Hello, NFA!

start \rightarrow q_0 \xrightarrow{h} q_1 \xrightarrow{i} q_2

- h
- i
Hello, NFA!

\[ \text{start} \rightarrow q_0 \xrightarrow{h} q_1 \xrightarrow{i} q_2 \]
Hello, NFA!

start \rightarrow q_0 \xrightarrow{h} q_1 \xrightarrow{i} q_2

SEAL OF APPROVAL

h i
Tragedy in Paradise

\[ q_0 \rightarrow q_1 \rightarrow q_2 \]

Start

\[ q_0 \rightarrow h \rightarrow q_1 \rightarrow i \rightarrow q_2 \]

\[ h \quad i \quad p \]
Tragedy in Paradise

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

h i p
Tragedy in Paradise
Tragedy in Paradise

start \rightarrow q_0 \xrightarrow{h} q_1 \xrightarrow{i} q_2

\begin{array}{ccc}
  h & i & p \\
\end{array}
Tragedy in Paradise

\[
\begin{align*}
q_0 & \xrightarrow{h} q_1 & q_1 & \xrightarrow{i} q_2
\end{align*}
\]
Tragedy in Paradise

\[ q_0 \rightarrow q_1 \rightarrow q_2 \]

Start with \( q_0 \) and follow the path 'hi' to end up in state \( q_2 \).
Tragedy in Paradise
Tragedy in Paradise
NFA Languages

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
\[ \mathcal{L}(N) = \{ w \in \Sigma^* \mid N \text{ accepts } w \}. \]

\[ \Sigma = \{0, 1\} \]
ε-Transitions

• NFAs have a special type of transition called the \textit{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.
\(\varepsilon\)-Transitions

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

![Diagram of ε-transitions in an NFA]
\(\varepsilon\)-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 **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.
\textbf{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.
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.
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 \texttt{ε-transition}.
- An NFA may follow any number of ε-transitions at any time without consuming any input.
- NFAs are not \textit{required} to follow ε-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
Perfect Positive Guessing

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

\[ \Sigma \]

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

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

Input sequence: a b a b a a
Perfect Positive Guessing

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

Input sequence: \texttt{aabababa}

The sequence is recognized by the automaton starting from the initial state \( q_0 \).
Perfect Positive Guessing

\[ \Sigma \]

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

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

a b a b a b a
Perfect Positive Guessing

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

Start state: \( q_0 \)
Input alphabet: \( \Sigma \)

Input sequence: \( abaaba \)
Perfect Positive Guessing

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

\(\Sigma\)

Start \(q_0\)

Input sequence: \(abaaba\)

States:
- \(q_0\)
- \(q_1\)
- \(q_2\)
- \(q_3\)
Perfect Positive Guessing

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

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

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

\[ \Sigma \]

Start

\[ a \quad b \quad a \quad b \quad a \quad b \quad a \]
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

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

Start state: \( q_0 \)

Inputs: \( \Sigma \)

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

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

\[ \sum \]

Input: a b a b a b a
Massive Parallelism

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

\[ \Sigma \]

\[ a \ b \ a \ b \ a \]

\[ \uparrow \]
Massive Parallelism

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

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

Strings: \( a\ b\ a\ b\ a\ b\ a \)
Massive Parallelism

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

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

\[ \Sigma \]

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

Input:

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

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

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

\[ \Sigma \]

start

\[ q_0 \] -> \[ q_1 \] -> \[ q_2 \] -> \[ q_3 \]

\[ a \ b \ a \ b \ a \ b \ a \]

\[ \uparrow \]
Massive Parallelism

\[ \Sigma \]

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

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

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

\[ \sum \]

\[ a \quad b \quad a \quad b \quad a \quad a \]

\[ \text{start} \]
Massive Parallelism

\[ \Sigma \]

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

Input sequence: a b a b a a
Massive Parallelism

\[ \Sigma \]

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

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

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

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

a b a b a a
Massive Parallelism

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

\[ \Sigma \]

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

\[ \Sigma \]

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

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

\[ \sum \]

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

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

\[ \Sigma \]

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

a b a b a a
Massive Parallelism

\[ \sigma \]

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

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

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

\[ \sum \]

a b a b a b a
Massive Parallelism
Massive Parallelism

\[ q_3 \]

\[ q_2 \]

\[ q_1 \]

\[ q_0 \]

\[ \Sigma \]

\[ \text{start} \]
Massive Parallelism

\[
\sum \quad a \quad b \\
q_0 \quad q_1 \quad q_2 \quad q_3
\]

\[a \quad b \quad a \quad b \quad a \quad a\]
We're in at least one accepting state, so there's some path that gets us to an accepting state.
Massive Parallelism

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

\[ \Sigma \]

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

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

\[ \sum \]

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

\[ \text{start} \]

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

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

\[
\Sigma
\]

a b a b b
Massive Parallelism
Massive Parallelism

\[ \begin{array}{c}
\text{start} & \rightarrow & q_0 & \rightarrow & a & q_1 & \rightarrow & b & q_2 & \rightarrow & a & q_3 \\
\Sigma & \rightarrow & \text{start}
\end{array} \]
Massive Parallelism

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

Transition:
- \( a \rightarrow q_1 \)
- \( b \rightarrow q_2 \)
- \( a \rightarrow q_3 \)

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

Start: $q_0$ → $q_1$ → $q_2$ → $q_3$

Symbols: $a$, $b$, $\Sigma$
Massive Parallelism
Massive Parallelism

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

\[ \Sigma \xrightarrow{a, b} \]
Massive Parallelism

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

Start

\[ \Sigma \]

Sequence:

a b a b a b
Massive Parallelism
Massive Parallelism

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

Input:

\[a b a b b\]
Massive Parallelism

\[
\begin{array}{c}
q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_3 \\
\sum
\end{array}
\]
Massive Parallelism
Massive Parallelism
Massive Parallelism

\[ \begin{array}{c}
q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_3
\end{array} \]

\[ \sum \]

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

\[ \Sigma \]

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

\[ \text{start} \]

\[ a \quad b \quad b \quad a \quad b \quad a \quad b \]

\[ \longrightarrow \]
Massive Parallelism

\[ q_0 \stackrel{a}{\rightarrow} q_1 \stackrel{b}{\rightarrow} q_2 \stackrel{a}{\rightarrow} q_3 \]

\[ \Sigma \]

\[ a \ b \ a \ b \]

\[ \text{start} \]
Massive Parallelism
Massive Parallelism

\[ q_3 = q_2 \]

\[ q_2 = q_1 \]

\[ q_1 = q_0 \]

\[ \Sigma \]

\[ \text{start} \rightarrow q_0 \rightarrow q_1 \rightarrow q_2 \rightarrow q_3 \]

\[ a \ b \ a \ b \]

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

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

Input alphabet: \( \sum \)
Massive Parallelism

\[ \Sigma \]

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

\[ \text{start} \rightarrow a \rightarrow b \rightarrow a \rightarrow \boxed{a b a b b} \]
Massive Parallelism

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

\[ \Sigma \]

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

\[ a \quad b \quad a \quad b \quad a \quad b \]
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

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

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

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

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

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

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

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

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

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

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

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

Diagram:

- Start state: \( q_0 \)
- States: \( q_0, q_1, q_2, q_3 \)
- Transitions:
  - \( q_0 \) on \( 0 \) to \( q_0 \)
  - \( q_0 \) on \( 1 \) to \( q_1 \)
  - \( q_1 \) on \( 1 \) to \( q_2 \)
  - \( q_2 \) on \( 0 \) to \( q_3 \)
  - \( q_3 \) on \( 0 \) to \( q_3 \)
  - \( q_3 \) on \( 1 \) to \( q_3 \)
  - \( q_3 \) on \( * \) to \( q_3 \)
Tabular DFAs

These stars indicate accepting states.
Since this is the first row, it's the start state.

<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

 questões:

Por que não há uma coluna aqui para Σ?

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>*q&lt;sub&gt;0&lt;/sub&gt;</td>
<td>q&lt;sub&gt;1&lt;/sub&gt;</td>
<td>q&lt;sub&gt;0&lt;/sub&gt;</td>
</tr>
<tr>
<td>q&lt;sub&gt;1&lt;/sub&gt;</td>
<td>q&lt;sub&gt;3&lt;/sub&gt;</td>
<td>q&lt;sub&gt;2&lt;/sub&gt;</td>
</tr>
<tr>
<td>q&lt;sub&gt;2&lt;/sub&gt;</td>
<td>q&lt;sub&gt;3&lt;/sub&gt;</td>
<td>q&lt;sub&gt;0&lt;/sub&gt;</td>
</tr>
<tr>
<td>*q&lt;sub&gt;3&lt;/sub&gt;</td>
<td>q&lt;sub&gt;3&lt;/sub&gt;</td>
<td>q&lt;sub&gt;3&lt;/sub&gt;</td>
</tr>
</tbody>
</table>
My Turn to Code Things Up!

```cpp
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?
\begin{align*}
\text{start} & \quad a \quad q_0 \\
& \quad \quad \quad \quad a \quad q_1 \\
& \quad \quad \quad \quad b \quad q_2 \\
& \quad \quad \quad \quad a \quad q_3
\end{align*}
The diagram represents a finite automaton with states labeled $q_0$, $q_1$, $q_2$, and $q_3$. The transitions are labeled with inputs 'a' and 'b'. The start state is $q_0$, and the input string is 'ababa'.
Start

$q_0$ → $q_1$ (a) → $q_2$ (b) → $q_3$

Input: ...

Transition: ?? ?? ?? ?? a ?? ?? ?? ?? ...

Next State: ?
\[ q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_3 \]

\[ \Sigma \]

<table>
<thead>
<tr>
<th>Input</th>
<th>States</th>
</tr>
</thead>
<tbody>
<tr>
<td>{q_0}</td>
<td>{q_0, q_1}</td>
</tr>
</tbody>
</table>

Table entries are not specified in the diagram.
\[ \begin{array}{c|c|c}
\text{a} & \text{b} \\
\hline
\{q_0\} & \{q_0, q_1\} & \{q_0\} \\
\hline
\hline
\hline
\end{array} \]
\[ \begin{array}{c|cc}
\text{state} & \text{a} & \text{b} \\
\hline
\{q_0\} & \{q_0, q_1\} & \{q_0\} \\
\end{array} \]
\[
q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_3
\]

<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>{q_0}</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[\Sigma \]

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

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

\( \Sigma \)

\begin{array}{ccc}
\{ q_0 \} & \{ q_0, q_1 \} & \{ q_0 \} \\
\{ q_0, q_1 \} & & \\
\{ q_0, q_1 \} & & \\
\{ q_0, q_1 \} & & \\
\end{array}
\[
\begin{array}{c}
q_0 \\
q_1 \\
q_2 \\
q_3
\end{array}
\]

\[
\begin{array}{c|c|c}
& a & b \\
\hline
\{q_0\} & \{q_0, q_1\} & \{q_0\} \\
\{q_0, q_1\} & \{q_0, q_1\} &  \\
\{q_0, q_1\} &  & \\
\{q_0, q_1\} &  & \\
\end{array}
\]
\[
\begin{array}{c|cc}
\Sigma & a & b \\
\hline
\{q_0\} & \{q_0, q_1\} & \{q_0\} \\
\{q_0, q_1\} & \{q_0, q_1\} & \\
\{q_0, q_1\} & & \\
\end{array}
\]
\[
\begin{array}{c}
\Sigma \\
\end{array}
\]

\[
\begin{array}{c}
\text{start} \\
\end{array}
\]

\[
\begin{array}{c}
q_0 \\
\end{array}
\]

\[
\begin{array}{c}
q_1 \\
\end{array}
\]

\[
\begin{array}{c}
q_2 \\
\end{array}
\]

\[
\begin{array}{c}
q_3 \\
\end{array}
\]

\[
\begin{array}{|c|c|c|}
\hline
& \text{a} & \text{b} \\
\hline
\{q_0\} & \{q_0, q_1\} & \{q_0\} \\
\{q_0, q_1\} & \{q_0, q_1\} & \{q_0, q_2\} \\
\{q_0, q_1\} & \{q_0, q_1\} & \{q_0, q_2\} \\
\hline
\end{array}
\]
\begin{array}{|c|c|c|}
\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 represents a finite automaton with states $q_0, q_1, q_2,$ 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 any input, move back to $q_0$.

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

<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>
<tr>
<td>State</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></td>
<td></td>
</tr>
</tbody>
</table>

Diagram:
- Start state: \(q_0\)
- Transitions:
  - \(q_0 \xrightarrow{a} q_1\)
  - \(q_1 \xrightarrow{b} q_2\)
  - \(q_2 \xrightarrow{a} q_3\)
- \(q_3\) is a final state.
- \(\Sigma\) is the input alphabet.

The diagram shows transitions labeled with symbols from the input alphabet \(\Sigma\) and states labeled with \(q_0, q_1, q_2, q_3\).
<table>
<thead>
<tr>
<th>State Set</th>
<th>Transition</th>
<th>Transition</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>

Start state: \(q_0\)

Transition labels: a, b

Accepting state: \(q_3\)
\[
\begin{array}{c}
\sum
\end{array}
\]

<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:** $q_0$
- **States:** $q_0$, $q_1$, $q_2$, $q_3$
- **Transitions:**
  - $q_0 \xrightarrow{a} q_1$
  - $q_1 \xrightarrow{b} q_2$
  - $q_2 \xrightarrow{a} q_3$
  - $q_0 \xrightarrow{\sum} q_0$
\[
\begin{array}{c}
\Sigma \\
\end{array}
\]

\[
\begin{array}{c}
\begin{array}{c}
q_0 \\
\end{array}
\end{array}
\xrightarrow{a}
\begin{array}{c}
q_1 \\
\end{array}
\xrightarrow{b}
\begin{array}{c}
q_2 \\
\end{array}
\xrightarrow{a}
\begin{array}{c}
q_3 \\
\end{array}
\]

<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>
</tbody>
</table>

- Start state: \(q_0\)
- Final state: \(q_3\)
\[\sum
\]

Diagram:

- Start state: \(q_0\)
- Transitions:
  - \(a\) from \(q_0\) to \(q_1\)
  - \(b\) from \(q_1\) to \(q_2\)
  - \(a\) from \(q_2\) to \(q_3\)

Table:

<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></td>
</tr>
</tbody>
</table>
\[
\begin{array}{ccc}
\{q_0\} & \{q_0, q_1\} & \{q_0\} \\
\{q_0, q_1\} & \{q_0, q_1\} & \{q_0, q_2\} \\
\{q_0, q_2\} & \{q_0, q_1, q_3\} & \\
\end{array}
\]
<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>${q_0}$</td>
</tr>
</tbody>
</table>

The diagram shows a DFA with the following transitions:

- From $q_0$ on $a$, go to $q_1$.
- From $q_1$ on $b$, go to $q_2$.
- From $q_2$ on $a$, go to $q_3$.
- $q_3$ is a final state.
\[
\begin{array}{ccc}
\{q_0\} & \{q_0, q_1\} & \{q_0\} \\
\{q_0, q_1\} & \{q_0, q_1\} & \{q_0, q_2\} \\
\{q_0, q_2\} & \{q_0, q_1, q_3\} & \{q_0\} \\
\end{array}
\]
\[ q_0 \overset{\Sigma}{\rightarrow} q_1 \overset{a}{\rightarrow} q_2 \overset{a}{\rightarrow} q_3 \]

<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>{q_0}</td>
</tr>
<tr>
<td>{q_0, q_1, q_3}</td>
<td>{q_0}</td>
<td></td>
</tr>
</tbody>
</table>
\[ \Sigma \]

**Diagram:**
- Start state: \( q_0 \)
- States: \( q_0, q_1, q_2, q_3 \)
- Transitions:
  - \( a \): \( q_0 \rightarrow q_1 \)
  - \( b \): \( q_1 \rightarrow q_2 \)
  - \( a \): \( q_2 \rightarrow q_3 \)

**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>( { q_0 } )</td>
</tr>
<tr>
<td>( { q_0, q_1, q_3 } )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>State Set</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>{q_0}</td>
</tr>
<tr>
<td>{q_0, q_1, q_3}</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

The diagram shows a transition graph with states \( q_0, q_1, q_2, q_3 \) and transitions labeled with symbols 'a' and 'b'.
<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>{q_0}</td>
</tr>
<tr>
<td>{q_0, q_1, q_3}</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
\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\} & \{q_0, q_1\} & \text{---} \\
\hline
\end{array}
<table>
<thead>
<tr>
<th>State Configuration</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>{q_0, q_1}</td>
<td>a</td>
<td>{q_0, q_1}</td>
</tr>
<tr>
<td>{q_0, q_1}</td>
<td>b</td>
<td>{q_0, q_2}</td>
</tr>
<tr>
<td>{q_0, q_2}</td>
<td>a</td>
<td>{q_0, q_1, q_3}</td>
</tr>
<tr>
<td>{q_0, q_1, q_3}</td>
<td>b</td>
<td>{q_0, q_1}</td>
</tr>
</tbody>
</table>
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>{q_0}</td>
</tr>
<tr>
<td>{q_0, q_1, q_3}</td>
<td>{q_0, q_1}</td>
<td></td>
</tr>
</tbody>
</table>

Diagram:

- Start state: \(q_0\)
- Transition on 'a': \(q_0 \rightarrow q_1\)
- Transition on 'b': \(q_1 \rightarrow q_2\)
- Transition on 'a' from \(q_2\): \(q_2 \rightarrow q_3\)

The diagram shows a finite automaton with states and transitions labeled with inputs 'a' and 'b'.
The given automaton has the following transitions:

- From state $q_0$, on input $a$, move to $q_1$.
- From state $q_1$, on input $b$, move to $q_2$.
- From state $q_2$, on input $a$, move to $q_3$.
- On any input $\Sigma$, loop back to $q_0$.

The transition table is as follows:

<table>
<thead>
<tr>
<th>Current State</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>---</td>
</tr>
<tr>
<td>State</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>{q_0}</td>
</tr>
<tr>
<td>{q_0, q_1, q_3}</td>
<td>{q_0, q_1}</td>
<td></td>
</tr>
</tbody>
</table>
\[
\begin{array}{c|cc}
\Sigma & a & b \\
\hline
\{q_0\} & \{q_0, q_1\} & \{q_0\} \\
\{q_0, q_1\} & \{q_0, q_1\} & \{q_0, q_2\} \\
\{q_0, q_2\} & \{q_0, q_1, q_3\} & \{q_0\} \\
\{q_0, q_1, q_3\} & \{q_0, q_1\} & \{q_0, q_2\} \\
\end{array}
\]
A non-deterministic finite automaton (NFA) with the following transitions:

- **Start state**: $q_0$
- **Symbols in the alphabet**: $\Sigma$
- **Transitions**:
  - 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 $\Sigma$, move to $q_0$.

<table>
<thead>
<tr>
<th>State Transition</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>${q_0, q_2}$</td>
</tr>
</tbody>
</table>
The diagram represents a deterministic finite automaton (DFA) with states \( q_0, q_1, q_2, q_3 \) and alphabet \( \Sigma \). 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 \( q_3 \).
- \( q_3 \) is a sink state.

The input string is 'a b a a a b a a'. The automaton starts in \( q_0 \) and goes through the states as indicated by the transitions.

The states are represented as sets:

- Initial state: \( \{ q_0 \} \)
- \( q_1 \) is reached from \( q_0 \) on 'a': \( \{ q_0, q_1 \} \)
- \( q_2 \) is reached from \( q_1 \) on 'b': \( \{ q_0, q_2 \} \)
- \( q_3 \) is a final state: \( \{ q_0, q_1, q_3 \} \)
\[ q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_3 \]

\[ \Sigma \]

\[
\begin{align*}
&\{q_0\} \\
&a \\
&\{q_0, q_1\} \\
&\{q_0, q_2\} \\
&\{q_0, q_1, q_3\}
\end{align*}
\]
\[ \Sigma \]

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

{\{q_0\}, \{q_0, q_1\}, \{q_0, q_2\}, \{q_0, q_1, q_3\}}

a b a a a b a a

\[
\begin{align*}
\text{start} & \rightarrow \{q_0\} \\
\{q_0\} & \xrightarrow{a} \{q_0, q_1\} \\
\{q_0, q_1\} & \xrightarrow{b} \{q_0, q_2\} \\
\{q_0, q_2\} & \xrightarrow{a} \{q_0, q_1, q_3\}
\end{align*}
\]
Some Caveats

- **Question**: what about ε-transitions?
  - Answer: always include any states you can reach by following ε-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 ε-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 $\mathcal{L}(N) = L$. 
An Important Result

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

**Proof Sketch:**
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.
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.
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 \textit{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 \( \overline{\overline{L}} = L \) 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

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

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

- **Theorem:** If $L$ is a regular language, then $\bar{L}$ 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 $L_1 \cup L_2$?
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 $L_1 \cup L_2$ 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 $L_1 \cup L_2$?

**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?
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

$$L_1 L_2 = \{ wx \in \Sigma^* \mid w \in L_1 \land x \in L_2 \}$$
Concatenation Example

• Let $\Sigma = \{ a, b, ..., z, A, B, ..., Z \}$ and consider these languages over $\Sigma$:
  
  • $Noun = \{ \text{Puppy, Rainbow, Whale, ... } \}$
  
  • $Verb = \{ \text{Hugs, Juggles, Loves, ... } \}$
  
  • $The = \{ \text{The} \}$
  
  • The language $\text{TheNounVerbTheNoun}$ is
    
    • $\{ \text{ThePuppyHugsTheWhale, TheWhaleLovesTheRainbow, TheRainbowJugglesTheRainbow, ... } \}$
Concatenation

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

  \[ L_1L_2 = \{ wx \in \Sigma^* \mid w \in L_1 \land x \in L_2 \} \]

- 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 $y \in L_2$?

![Machine for $L_1$](start_circle)

![Machine for $L_2$](start_circle)
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 $L_1$](image1.png)
![Machine for $L_2$](image2.png)
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 $L_1$](image1)

![Machine for $L_2$](image2)

**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 $L_1 L_2$
Lots and Lots of Concatenation

• Consider the language $L = \{ aa, b \}$
• $LL$ is the set of strings formed by concatenating pairs of strings in $L$.
  
  \{ aaaa, aab, baa, bb \}

• $LLL$ is the set of strings formed by concatenating triples of strings in $L$.
  
  \{ aaaaaa, aaaaab, aabaa, aabb, baaaa, baab, bbbaa, bbb \}

• $LLLL$ is the set of strings formed by concatenating quadruples of strings in $L$.
  
  \{ aaaaaaaaa, aaaaaaaaaab, aaaaaabaa, aaaaabb, aabaaaaa, aabaab, aabbaa, aabbb, baaaaaa, baaaaab, baabaa, baabb, bbaaaaa, 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 \( L = \{ \text{a, bb} \} \), then \( L^* = \{ \)

\[ \varepsilon, \]

\[ \text{a, bb}, \]

\[ \text{aa, abb, bba, bbbb}, \]

\[ \text{aaa, aabb, abba, aaaaa, bbbaa, bbbab, bbbbba, bbbbbbb,} \]

\[ \ldots \]

\}
Reasoning about Infinity

• If $L$ is regular, is $L^*$ necessarily regular?

⚠️ A Bad Line of Reasoning: ⚠️

• $L^0 = \{ \varepsilon \}$ 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
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

$0.9999\bar{9} \not< 1$
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

∞ is finite
Reasoning about Infinity

\( \infty \) is finite

\( ^\wedge \text{not} \)
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

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

Machine for $L$

Machine for $L^*$

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!

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

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

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

\[ \{ q_0, q_3 \} \]

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

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

Start:\n- \( q_0 \) with \( \epsilon \) transitions to \( q_1 \) and \( q_3 \)
- \( q_0 \) with \( a \) transition to \( q_1 \)
- \( q_3 \) with \( b \) transition to \( q_4 \)

\( \Sigma \) transitions:
- \( q_0 \) to \( q_1 \)
- \( q_3 \) to \( q_4 \)
- \( q_2 \) to \( q_3 \)

\( \{ q_0, q_3 \} \)
Once More, With Epsilons!

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

\[ \begin{array}{c|c|c}
\Sigma & a & b \\
\hline
\{q_0, q_3\} & & \\
\hline
\end{array} \]

The diagram shows a transition graph with states \( q_0, q_1, q_2, q_3, q_4 \) and transitions labeled by symbols from the alphabet \( \Sigma \).
Once More, With Epsilons!

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

\[
\begin{array}{c}
\text{start} \\
q_0 \\
\varepsilon \quad \Sigma \\
q_3 \\
\Sigma \\
q_4 \\
\end{array}
\]

\[
\begin{array}{c}
q_2 \\
\Sigma \\
b \\
q_1 \\
q_3 \\
\Sigma \\
q_4 \\
\end{array}
\]

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

\[
\begin{align*}
\text{start} & \rightarrow q_0 \\
q_0 & \xrightarrow{a} q_1 \\
q_0 & \xrightarrow{\varepsilon} q_3 \\
q_2 & \xrightarrow{\Sigma} q_1 \\
q_2 & \xrightarrow{b} q_3 \\
q_3 & \xrightarrow{\Sigma} q_4 \\
q_3 & \xrightarrow{b} q_4
\end{align*}
\]

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

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

\[
\begin{array}{|c|c|c|}
\hline
& a & b \\
\hline
\{q_0, q_3\} & \{q_1, q_4\} & \{q_4\} \\
\{q_1, q_4\} & & \\
\hline
\end{array}
\]
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:
  - \(q_0 \xrightarrow{\epsilon} q_1\)
  - \(q_0 \xrightarrow{a} q_1\)
  - \(q_1 \xrightarrow{\Sigma} q_2\)
  - \(q_1 \xrightarrow{b} q_4\)
  - \(q_3 \xrightarrow{b} q_4\)
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>

Start state: \(q_0\)

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

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

<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></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Once More, With Epsilons!

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

- \( q_0 \) to \( q_1 \) on \( a \)
- \( q_0 \) to \( q_3 \) on \( \varepsilon \)
- \( q_3 \) to \( q_4 \) on \( b \)
- \( q_2 \) on \( \Sigma \)
- \( q_1 \) on \( b \)

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

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

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

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

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

\[
\begin{array}{c}
q_2 \\
q_1 \quad \quad \quad \quad q_4
\end{array}
\]

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

\[
\begin{array}{c}
\epsilon \\
b
\end{array}
\]

\[
\begin{array}{c}
a \\
\{q_0, q_3\} \quad \{q_1, q_4\} \quad \{q_4\}
\end{array}
\]

\[
\begin{array}{c}
b \\
\{q_1, q_4\} \quad \emptyset \quad \{q_2, q_3\}
\end{array}
\]

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

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

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

![Diagram of a deterministic finite automaton (DFA) with states q₀, q₁, q₂, q₃, and q₄. The transitions are labeled with ε (for the start state), a, b, and ∑ (for the start state and q₃). The transitions to q₁ include a and b, and to q₃ include a and b.]

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>{q₀, q₃}</td>
<td>{q₁, q₄}</td>
<td>{q₄}</td>
</tr>
<tr>
<td>{q₁, q₄}</td>
<td>Ø</td>
<td>{q₂, q₃}</td>
</tr>
<tr>
<td>{q₄}</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Once More, With Epsilons!

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

The diagram illustrates a finite automaton with states $q_0, q_1, q_2, q_3, q_4$ and transitions labeled with symbols $a$ and $b$. The transitions are:

- $q_0$ to $q_1$ on $a$.
- $q_0$ to $q_3$ on $\varepsilon$.
- $q_1$ to $q_4$ on $b$.
- $q_2$ to $q_1$ on $\Sigma$.
- $q_3$ to $q_4$ on $\Sigma$.

The transition table is as follows:

<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!

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

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

\begin{itemize}
  \item \textbf{q}_0 \rightarrow \{q_0, q_3\}
  \item \textbf{q}_1 \rightarrow \{q_1, q_4\}
  \item \textbf{q}_2 \rightarrow \{q_4\}
  \item \textbf{q}_3 \rightarrow \{q_4\}
  \item \textbf{q}_4 \rightarrow \{q_4\}
  \item \textbf{s}tart \rightarrow \{q_0, q_3\}
\end{itemize}
Once More, With Epsilons!

![Diagram of a state machine with transitions labeled by symbols and states.]

<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></td>
</tr>
</tbody>
</table>

\(\Sigma\)

\(q_0\)

\(q_1\)

\(q_2\)

\(q_3\)

\(q_4\)
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></td>
</tr>
</tbody>
</table>
Once More, With Epsilons!
Once More, With Epsilons!

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

\[
\begin{array}{c|c|c}
& a & b \\
\hline
\{q_0, q_3\} & \{q_1, q_4\} & \{q_4\} \\
\{q_1, q_4\} & \emptyset & \{q_2, q_3\} \\
\{q_4\} & \emptyset & \{q_3\} \\
\end{array}
\]
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>\emptyset</td>
<td>{q_2, q_3}</td>
</tr>
<tr>
<td>{q_4}</td>
<td>\emptyset</td>
<td>{q_3}</td>
</tr>
</tbody>
</table>
Once More, With Epsilons!
Once More, With Epsilons!

\[
\begin{array}{ccc}
\text{start} & \overset{\varepsilon}{\longrightarrow} & q_0 \\
& \overset{\Sigma}{\longrightarrow} & q_2 \\
& \overset{\Sigma}{\longrightarrow} & q_3 \\
q_0 & \overset{a}{\longrightarrow} & q_1 \\
& \overset{b}{\longrightarrow} & q_4 \\
q_3 & \overset{b}{\longrightarrow} & q_4
\end{array}
\]

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

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

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

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

The diagram shows a finite automaton with states \( q_0, q_1, q_2, q_3, q_4 \) and transitions labeled with symbols \( a \) and \( b \). The transitions include epsilon transitions (\( \epsilon \)) and specific transitions for \( a \) and \( b \).
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></td>
<td></td>
</tr>
</tbody>
</table>

**Diagram:**
- Start state: $q_0$
- Transitions:
  - $q_0 \xrightarrow{a} q_1$
  - $q_0 \xrightarrow{\epsilon} q_2$
  - $q_0 \xrightarrow{\Sigma} q_3$
  - $q_1 \xrightarrow{b} q_4$
  - $q_2 \xrightarrow{b} q_4$
  - $q_3 \xrightarrow{\Sigma} q_4$
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>
</tbody>
</table>

**Diagram:**

- **Start State:** \(q_0\)
- **Final States:** \(q_3, q_4\)
- **Transitions:**
  - \(q_0\) on \(\Sigma\) to \(q_2\)
  - \(q_0\) on \(\epsilon\) to \(q_3\)
  - \(q_2\) on \(a\) to \(q_1\)
  - \(q_2\) on \(b\) to \(q_3\)
  - \(q_1\) on \(a\) to \(q_0\)
  - \(q_4\) on \(b\) to \(q_4\)
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>
</tbody>
</table>

Transition Diagram:

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

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

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

**Table:**

<table>
<thead>
<tr>
<th>States</th>
<th>$\Sigma a$</th>
<th>$\Sigma b$</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>

**Notes:**
- $\Sigma$ represents the alphabet over which the automaton operates.
- $\varepsilon$ denotes the empty word or epsilon.
- The states $q_0, q_1, q_2, q_3, q_4$ are the possible states of the automaton.
Once More, With Epsilons!
Once More, With Epsilons!

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

\[
\begin{array}{c}
\sum \\
a \\
\varepsilon \\
\sum \\
b
\end{array}
\]

<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_4}</td>
<td>{q_4}</td>
<td>{q_4}</td>
</tr>
</tbody>
</table>

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

A state diagram with states and transitions labeled with symbols $\epsilon$ and $\Sigma$. The diagram includes transitions for symbols $a$ and $b$.

A transition table with states and symbols $a$ and $b$. The table includes state transitions for symbols $a$ and $b$.
Once More, With Epsilons!
Once More, With Epsilons!
Once More, With Epsilons!

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

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

\[
\begin{array}{ccc}
\text{start} & q_0 & q_1 \\
\Sigma & a & b \\
q_2 & & \\
\epsilon & q_3 & q_4 \\
q_3 & b & \\
\end{array}
\]

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

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

\begin{center}
\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\} & \\
\hline
\end{tabular}
\end{center}
Once More, With Epsilons!
Once More, With Epsilons!

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

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

![Diagram of a non-deterministic finite automaton (NFA) with states, transitions, and a list of state transitions](image)

<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!
Once More, With Epsilons!
Once More, With Epsilons!

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

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

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

\[
\begin{array}{c|c|c}
 & a & b \\
\hline
\{q_0, q_3\} & \{q_1, q_4\} & \{q_4\} \\
\{q_1, q_4\} & \emptyset & \{q_2, q_3\} \\
\{q_4\} & \emptyset & \{q_3\} \\
\{q_2, q_3\} & \{q_0, q_3, q_4\} & \{q_0, q_3, q_4\} \\
\{q_3\} & \{q_4\} & \{q_4\} \\
\{q_0, q_3, q_4\} & \{q_1, q_4\} & \{q_3, q_4\} \\
\{q_3, q_4\} & \{q_4\} & \{q_4\} \\
\end{array}
\]
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>$\emptyset$</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>

Diagram:

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

Transition Table:

- $q_0 \xrightarrow{\varepsilon} q_3$
- $q_3 \xrightarrow{a} q_0$
- $q_3 \xrightarrow{b} q_4$
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>
Once More, With Epsilons!

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

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

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

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

\[
\begin{array}{|c|c|c|}
\hline
\text{state} & \text{a} & \text{b} \\
\hline
\{q_0, q_3\} & \{q_1, q_4\} & \{q_4\} \\
\{q_1, q_4\} & \emptyset & \{q_2, q_3\} \\
\{q_4\} & \emptyset & \{q_3\} \\
\{q_2, q_3\} & \{q_0, q_3, q_4\} & \{q_0, q_3, q_4\} \\
\{q_3\} & \{q_4\} & \{q_4\} \\
\{q_0, q_3, q_4\} & \{q_1, q_4\} & \{q_3, q_4\} \\
\{q_3, q_4\} & \{q_4\} & \{q_3, q_4\} \\
\emptyset & \emptyset & \emptyset \\
\hline
\end{array}
\]
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_0, q_3, q_4}</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!

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

\[
\begin{array}{|c|c|c|}
\hline
& a & b \\
\hline
*\{q_0, q_3\} & \{q_1, q_4\} & \{q_4\} \\
*\{q_1, q_4\} & \emptyset & \{q_2, q_3\} \\
\{q_4\} & \emptyset & \{q_3\} \\
*\{q_2, q_3\} & \{q_0, q_3, q_4\} & \{q_0, q_3, q_4\} \\
*\{q_3\} & \{q_4\} & \{q_4\} \\
*\{q_0, q_3, q_4\} & \{q_1, q_4\} & \{q_3, q_4\} \\
*\{q_3, q_4\} & \{q_4\} & \{q_3, q_4\} \\
\emptyset & \emptyset & \emptyset \\
\hline
\end{array}
\]