Regular Expressions

and

The Limits of Regular Languages
Announcements

• Midterm *tonight* in Cubberly Auditorium, 7PM – 10PM.
  • Open-book, open-note, open-computer, closed-network.
  • Covers material up to and including last Monday's lecture.
Regular Expressions
Atomic Regular Expressions

• The regular expressions begin with three simple building blocks.
  
• The symbol $\emptyset$ is a regular expression that represents the empty language $\emptyset$.

• The symbol $\varepsilon$ is a regular expression that represents the language $\{ \varepsilon \}$
  • This is not the same as $\emptyset$!

• For any $a \in \Sigma$, the symbol $a$ is a regular expression for the language $\{ a \}$
Compound Regular Expressions

- We can combine together existing regular expressions in four ways.
- If \( R_1 \) and \( R_2 \) are regular expressions, \( R_1 R_2 \) is a regular expression represents the concatenation of the languages of \( R_1 \) and \( R_2 \).
- If \( R_1 \) and \( R_2 \) are regular expressions, \( R_1 | R_2 \) is a regular expression representing the union of \( R_1 \) and \( R_2 \).
- If \( R \) is a regular expression, \( R^* \) is a regular expression for the Kleene closure of \( R \).
- If \( R \) is a regular expression, \( (R) \) is a regular expression with the same meaning as \( R \).
Operator Precedence

- Regular expression operator precedence is

\[(R)\]
\[R^*\]
\[R_1R_2\]
\[R_1 | R_2\]

- So \(ab^*c|d\) is parsed as \(((a(b^*))c)|d\)
Regular Expressions, Formally

- The **language of a regular expression** is the language described by that regular expression.

- Formally:
  - $\mathcal{L}(\varepsilon) = \{\varepsilon\}$
  - $\mathcal{L}(\emptyset) = \emptyset$
  - $\mathcal{L}(a) = \{a\}$
  - $\mathcal{L}(R_1 R_2) = \mathcal{L}(R_1) \mathcal{L}(R_2)$
  - $\mathcal{L}(R_1 | R_2) = \mathcal{L}(R_1) \cup \mathcal{L}(R_2)$
  - $\mathcal{L}(R^*) = \mathcal{L}(R)^*$
  - $\mathcal{L}((R)) = \mathcal{L}(R)$
Regular Expressions are Awesome

• Let $\Sigma = \{0, 1\}$

• Let $L = \{ w \in \Sigma^* \mid w \text{ contains } 00 \text{ as a substring } \}$
Regular Expressions are Awesome

- Let $\Sigma = \{0, 1\}$
- Let $L = \{ w \in \Sigma^* \mid w$ contains $00$ as a substring $\}$

$$\begin{align*}
(0 | 1)^*00(0 | 1)^*
\end{align*}$$
Regular Expressions are Awesome

- Let $\Sigma = \{0, 1\}$
- Let $L = \{ w \in \Sigma^* \mid w \text{ contains } 00 \text{ as a substring } \}$

$(0 \mid 1)^*00(0 \mid 1)^*$
Regular Expressions are Awesome

- Let $\Sigma = \{0, 1\}$
- Let $L = \{ w \in \Sigma^* \mid w \text{ contains } 00 \text{ as a substring} \}$

$$(0 \mid 1)^*00(0 \mid 1)^*$$

11011100101
0000
11111011110011111
Regular Expressions are Awesome

- Let $\Sigma = \{0, 1\}$
- Let $L = \{ w \in \Sigma^* \mid w$ contains $00$ as a substring $\}$

$$(0 \mid 1)^*00(0 \mid 1)^*$$

11011100101
0000
0000
11111011110011111
Regular Expressions are Awesome

- Let $\Sigma = \{0, 1\}$
- Let $L = \{ w \in \Sigma^* \mid |w| = 4 \}$
Regular Expressions are Awesome

Let $\Sigma = \{0, 1\}$

Let $L = \{ w \in \Sigma^* \mid |w| = 4 \}$
Regular Expressions are Awesome

Let $\Sigma = \{0, 1\}$

Let $L = \{w \in \Sigma^* \mid |w| = 4\}$

The length of a string $w$ is denoted $|w|$.
Regular Expressions are Awesome

• Let $\Sigma = \{0, 1\}$

• Let $L = \{ w \in \Sigma^* \mid |w| = 4 \}$
Regular Expressions are Awesome

- Let $\Sigma = \{0, 1\}$
- Let $L = \{ w \in \Sigma^* \mid |w| = 4 \}$

$(0|1)(0|1)(0|1)(0|1)$
Regular Expressions are Awesome

- Let $\Sigma = \{0, 1\}$
- Let $L = \{ w \in \Sigma^* \mid |w| = 4 \}$

$(0|1)(0|1)(0|1)(0|1)$
Regular Expressions are Awesome

• Let $\Sigma = \{0, 1\}$

• Let $L = \{ w \in \Sigma^* \mid |w| = 4 \}$

(0|1)(0|1)(0|1)(0|1)

0000
1010
1111
1000
Regular Expressions are Awesome

- Let $\Sigma = \{0, 1\}$
- Let $L = \{ w \in \Sigma^* \mid |w| = 4 \}$

$(0|1)(0|1)(0|1)(0|1)$

0000
1010
1111
1000
Regular Expressions are Awesome

• Let $\Sigma = \{0, 1\}$
• Let $L = \{ w \in \Sigma^* \mid |w| = 4 \}$

$(0|1)^4$

0000
1010
1111
1000
Regular Expressions are Awesome

• Let $\Sigma = \{0, 1\}$
• Let $L = \{ w \in \Sigma^* \mid |w| = 4 \}$

$(0|1)^4$

0000
1010
1111
1000
Regular Expressions are Awesome

- Let $\Sigma = \{0, 1\}$
- Let $L = \{ w \in \Sigma^* \mid w \text{ contains at most one } 0 \}$
Regular Expressions are Awesome

- Let $\Sigma = \{0, 1\}$
- Let $L = \{w \in \Sigma^* \mid w$ contains at most one $0 \}$

$$1^*(0 \mid \varepsilon)1^*$$
Regular Expressions are Awesome

- Let $\Sigma = \{0, 1\}$
- Let $L = \{ w \in \Sigma^* \mid w \text{ contains at most one } 0 \}$

$1^* (0 \mid \varepsilon) 1^*$
Regular Expressions are Awesome

- Let $\Sigma = \{0, 1\}$
- Let $L = \{w \in \Sigma^* \mid w \text{ contains at most one } 0\}$

$1^*(0 \mid \varepsilon)1^*$

11110111
111111
0111
0
Regular Expressions are Awesome

- Let $\Sigma = \{0, 1\}$
- Let $L = \{ w \in \Sigma^* \mid w$ contains at most one 0 $\}$

$$1^* (0 | \varepsilon) 1^*$$
Regular Expressions are Awesome

- Let $\Sigma = \{0, 1\}$
- Let $L = \{ w \in \Sigma^* | w$ contains at most one 0 $\}$

\[
1^*0?1^*
\]

\[
\begin{align*}
11110111 \\
111111 \\
0111 \\
0
\end{align*}
\]
Regular Expressions are Awesome

• Let $\Sigma = \{ a, ., @ \}$, where $a$ represents "some letter."

• Regular expression for email addresses:

$$aa^* (.aa^*)^* @ aa^*.aa^* (.aa^*)^*$$
Regular Expressions are Awesome

- Let $\Sigma = \{ a, ., @ \}$, where $a$ represents “some letter.”
- Regular expression for email addresses:
  
  $$aa^* (.aa^*)^* @ aa^*.aa^* (.aa^*)^*$$

  - cs103@cs.stanford.edu
  - first.middle.last@mail.site.org
  - barack.obama@whitehouse.gov
Regular Expressions are Awesome

• Let $\Sigma = \{a, ., @\}$, where $a$ represents “some letter.”

• Regular expression for email addresses:

$$aa^* (.aa^*)^* @ aa^*.aa^* (.aa^*)^*$$

    cs103@cs.stanford.edu
    first.middle.last@mail.site.org
    barack.obama@whitehouse.gov
Regular Expressions are Awesome

• Let $\Sigma = \{ a, ., @ \}$, where $a$ represents “some letter.”

• Regular expression for email addresses:

\[
aa^* (.aa^*)^* @ aa^*.aa^* (.aa^*)^*
\]

\[
\text{cs103@cs.stanford.edu}
\]
\[
\text{first.middle.last@mail.site.org}
\]
\[
\text{barack.obama@whitehouse.gov}
\]
Regular Expressions are Awesome

• Let $\Sigma = \{ a, ., @ \}$, where $a$ represents “some letter.”

• Regular expression for email addresses:

$$aa^* (.aa^*)^* @ aa^*.aa^* (.aa^*)^*$$

- cs103@cs.stanford.edu
- first.middle.last@mail.site.org
- barack.obama@whitehouse.gov
Regular Expressions are Awesome

• Let $\Sigma = \{ a, ., @ \}$, where $a$ represents “some letter.”

• Regular expression for email addresses:

   $a^+ (.aa*)* @ aa*.aa* (.aa*)*$

   - cs103@cs.stanford.edu
   - first.middle.last@mail.site.org
   - barack.obama@whitehouse.gov
Regular Expressions are Awesome

• Let $\Sigma = \{ a, ., @ \}$, where $a$ represents “some letter.”

• Regular expression for email addresses:

\[ a^+ (.a^+)* @ a^+.a^+ (.a^+)* \]

- cs103@cs.stanford.edu
- first.middle.last@mail.site.org
- barack.obama@whitehouse.gov
Regular Expressions are Awesome

• Let $\Sigma = \{ \text{a, ., @} \}$, where $\text{a}$ represents “some letter.”

• Regular expression for email addresses:

$$a^+ (.a^+)^* @ a^+.a^+ (.a^+)^*$$

- cs103@cs.stanford.edu
- first.middle.last@mail.site.org
- barack.obama@whitehouse.gov
Regular Expressions are Awesome

• Let $\Sigma = \{ \text{a, ., @} \}$, where a represents “some letter.”

• Regular expression for email addresses:

$$a^+ (.a^+)* @ a^+ (.a^+)*$$

cs103@cs.stanford.edu
first.middle.last@mail.site.org
barack.obama@whitehouse.gov
Regular Expressions are Awesome

- Let $\Sigma = \{ \text{a, ., @} \}$, where a represents “some letter.”

- Regular expression for email addresses:

  $$a^+(a^+)*@a^+(a^+)^+$$

  - cs103@cs.stanford.edu
  - first.middle.last@mail.site.org
  - barack.obama@whitehouse.gov
Regular Expressions are Awesome

\[ a^+ (a^*)^* @ a^+ (a^*)^+ \]
The Power of Regular Expressions

**Theorem:** If $R$ is a regular expression, then $\mathcal{L}(R)$ is regular.

**Proof idea:** Induction over the structure of regular expressions. Atomic regular expressions are the base cases, and the inductive step handles each way of combining regular expressions.

Sketch of proof at the appendix of these slides.
The Power of Regular Expressions

**Theorem:** If $L$ is a regular language, then there is a regular expression for $L$. 

*This is not obvious!*

**Proof idea:** Show how to convert an arbitrary NFA into a regular expression.
From NFAs to Regular Expressions

\[ s_1, s_2, \ldots, s_n \]
From NFAs to Regular Expressions

Regular expression: \((s_1 | s_2 | ... | s_n)^*\)
From NFAs to Regular Expressions

Regular expression: \((s_1 \mid s_2 \mid \ldots \mid s_n)^*\)
From NFAs to Regular Expressions

Regular expression: \((s_1 \mid s_2 \mid \ldots \mid s_n)^*\)

Key idea: Label transitions with arbitrary regular expressions.
From NFAs to Regular Expressions
From NFAs to Regular Expressions

Regular expression: $R$
From NFAs to Regular Expressions

Key idea: If we can convert any NFA into something that looks like this, we can easily read off the regular expression.

Regular expression: $R$
From NFAs to Regular Expressions

Regular expression: $R$

$\begin{align*}
\text{start} & \quad R \\
\text{start} & \quad s_1 \mid s_2 \mid \ldots \mid s_n
\end{align*}$
From NFAs to Regular Expressions

Regular expression: $R$

Regular expression: $(s_1 | s_2 | \ldots | s_n)^*$
From NFAs to Regular Expressions

Regular expression: $R$

Regular expression: $(s_1 \mid s_2 \mid \ldots \mid s_n)^*$
From NFAs to Regular Expressions

Regular expression: \( R \)
From NFAs to Regular Expressions

Regular expression: $R$

$s_1 \mid s_2 \mid \ldots \mid s_n$
From NFAs to Regular Expressions

Regular expression: \( R \)

Regular expression: \( \emptyset \)
From NFAs to Regular Expressions

Regular expression: $R$

Regular expression: $\emptyset$
From NFAs to Regular Expressions

Regular expression: $R$

Diagram:

- Start state
- Transition labeled $R$
- Accepting state
From NFAs to Regular Expressions

Regular expression: $R$

Regular expression:

$Q_1 \xrightarrow{R_{11}} Q_2 \xrightarrow{R_{22}}$

$Q_1 \xrightarrow{R_{12}} Q_2 \xrightarrow{R_{21}}$

Start
From NFAs to Regular Expressions

Regular expression: $R$

Start state $q_1$ transitions to $q_2$ via the regular expression $R_{11} \ast R_{12} (R_{22} \mid R_{21} R_{11} \ast R_{12}) \ast$. 

Start state transitions to state via the regular expression $R$. 

Diagram:

- Start state $q_1$ transitions to state $q_2$ via the regular expression $R_{11} \ast R_{12} (R_{22} \mid R_{21} R_{11} \ast R_{12}) \ast$. 
- Start state transitions to state via the regular expression $R$. 

Diagram:

- Start state $q_1$ transitions to state $q_2$ via the regular expression $R_{11} \ast R_{12} (R_{22} \mid R_{21} R_{11} \ast R_{12}) \ast$. 
- Start state transitions to state via the regular expression $R$. 

Diagram:

- Start state $q_1$ transitions to state $q_2$ via the regular expression $R_{11} \ast R_{12} (R_{22} \mid R_{21} R_{11} \ast R_{12}) \ast$. 
- Start state transitions to state via the regular expression $R$. 

Diagram:
From NFAs to Regular Expressions

Regular expression: $R$

$R_{11}^* R_{12} (R_{22} \mid R_{21} R_{11}^* R_{12})^*$
From NFAs to Regular Expressions
From NFAs to Regular Expressions
From NFAs to Regular Expressions
From NFAs to Regular Expressions
From NFAs to Regular Expressions
From NFAs to Regular Expressions

Could we eliminate this state from the NFA?
From NFAs to Regular Expressions
From NFAs to Regular Expressions
From NFAs to Regular Expressions

Note: We’re using concatenation and Kleene closure in order to skip this state.
From NFAs to Regular Expressions
From NFAs to Regular Expressions

\[ \varepsilon R_{11} * R_{12} \]
From NFAs to Regular Expressions

\[ \varepsilon R_{11} \ast R_{12} \]
From NFAs to Regular Expressions

\[
\begin{align*}
\varepsilon R_{11} & \cdot R_{12} \\
R_{21} & \cdot R_{11} \cdot R_{12} \\
\end{align*}
\]
From NFAs to Regular Expressions
From NFAs to Regular Expressions
From NFAs to Regular Expressions

\[
\begin{align*}
R_{11} & \cdot R_{12} \\
R_{21} & \cdot R_{11} \cdot R_{12}
\end{align*}
\]
From NFAs to Regular Expressions

\[ R_{11} \ast R_{12} \]

Note: We’re using union to combine these transitions together.
From NFAs to Regular Expressions

\[ q_s \xrightarrow{R_{11} \times R_{12}} q_2 \xrightarrow{\epsilon} q_f \]

\[ R_{22} \mid R_{21} R_{11} \times R_{12} \]
From NFAs to Regular Expressions

\[ \text{start} \xrightarrow{R_{11} \ast R_{12}} q_2 \xrightarrow{\varepsilon} q_f \]

\[ R_{22} \lor R_{21} R_{11} \ast R_{12} \]
From NFAs to Regular Expressions

\[ q_s \xrightarrow{R_{11} \ast R_{12}} q_2 \xrightarrow{\varepsilon} q_f \]

\[ R_{22} \mid R_{21} R_{11} \ast R_{12} \]
From NFAs to Regular Expressions

\[
\text{start} \rightarrow q_s \xrightarrow{R_{11} \ast R_{12}} q_2 \xrightarrow{\varepsilon} q_f
\]

\[
R_{22} \mid R_{21} R_{11} \ast R_{12}
\]
From NFAs to Regular Expressions

\[ R_{11} \cdot R_{12} \ (R_{22} \mid R_{21} R_{11} \cdot R_{12})^* \epsilon \]
From NFAs to Regular Expressions

\[ R_{11} \ast R_{12} \ (R_{22} \mid R_{21} R_{11} \ast R_{12}) \ast \varepsilon \]
From NFAs to Regular Expressions

\[ R_{11}^* R_{12} (R_{22} \mid R_{21} R_{11}^* R_{12})^* \varepsilon \]
From NFAs to Regular Expressions

\[ R_{11}^* R_{12} (R_{22} \mid R_{21} R_{11}^* R_{12})^* \]
From NFAs to Regular Expressions

\[ R_{11}^* R_{12} (R_{22} \mid R_{21} R_{11}^* R_{12})^* \]
From NFAs to Regular Expressions

\[ R_{11} \ast R_{12} (R_{22} | R_{21} R_{11} \ast R_{12})^\ast \]
The Construction at a Glance

- Start with an NFA for the language $L$.
- Add a new start state $q_s$ and accept state $q_f$ to the NFA.
  - Add $\varepsilon$-transitions from each original accepting state to $q_f$, then mark them as not accepting.
- Repeatedly remove states other than $q_s$ and $q_f$ from the NFA by “shortcutting” them until only two states remain: $q_s$ and $q_f$.
- The transition from $q_s$ to $q_f$ is then a regular expression for the NFA.
There's another example!

Check the appendix to this slide deck.
Our Transformations

DFA

NFA

Regexp

direct conversion

subset construction

state elimination

recursive transform
Regular Languages

• A language $L$ is regular iff
  • $L$ is accepted by some DFA.
  • $L$ is accepted by some NFA.
  • $L$ is described by some regular expression.

• What constructions on regular languages can we do with regular expressions?
String Homomorphism

- Let $\Sigma_1$ and $\Sigma_2$ be alphabets.
- Consider any function $h : \Sigma_1 \rightarrow \Sigma_2^*$ that associates symbols in $\Sigma_1$ with strings in $\Sigma_2^*$.
- For example:
  - $\Sigma_1 = \{ 0, 1 \}$
  - $\Sigma_2 = \{ a, b, c, d \}$
  - $h(0) = \text{acdb}$
  - $h(1) = \text{ccc}$
String Homomorphism

- Let $\Sigma_1$ and $\Sigma_2$ be alphabets.
- Consider any function $h : \Sigma_1 \rightarrow \Sigma_2^*$ that associates symbols in $\Sigma_1$ with strings in $\Sigma_2^*$.
- For example:
  - $\Sigma_1 = \{a, b, c, d, \ldots\}$
  - $\Sigma_2 = \{A, B, C, D, \ldots\}$
  - $h(a) = A$
  - $h(b) = B$
  - $\ldots$
String Homomorphism

- Let $\Sigma_1$ and $\Sigma_2$ be alphabets.
- Consider any function $h : \Sigma_1 \to \Sigma_2^*$ that associates symbols in $\Sigma_1$ with strings in $\Sigma_2^*$.
- For example:
  - $\Sigma_1 = \{ 0, 1 \}$
  - $\Sigma_2 = \{ 0, 1 \}$
  - $h(0) = \varepsilon$
  - $h(1) = 1$
String Homomorphism

- Given a function \( h : \Sigma_1 \rightarrow \Sigma_2^* \), the function \( h^* : \Sigma_1^* \rightarrow \Sigma_2^* \) is formed by applying \( h \) to each character of a string \( w \).

- This function is called a **string homomorphism**.
  - From Greek “same shape.”
String Homomorphism, Intuitively

- Example: Let $\Sigma_1 = \{ 0, 1, 2 \}$ and consider the string $0121$
- If $\Sigma_2 = \{ A, B, C, \ldots, Z, a, b, \ldots, z, ', [ , ] , . \}$, define $h : \Sigma_1 \to \Sigma_2^*$ as
  - $h(0) =$ That's the way
  - $h(1) =$ [Uh huh uh huh]
  - $h(2) =$ I like it
- Then $h^*(0121) =$ That's the way [Uh huh uh huh] I like it [Uh huh uh huh]
- Note that $h^*(0121)$ has the same structure as $0121$, just expressed differently.
Homomorphisms of Languages

- If $L \subseteq \Sigma_1^*$ is a language and $h^*: \Sigma_1^* \rightarrow \Sigma_2^*$ is a homomorphism, the language $h^*(L)$ is defined as
  
  $$h^*(L) = \{ h^*(w) \mid w \in L \}$$

- The language formed by applying the homomorphism to every string in $L$. 
Homomorphisms of Regular Languages

• **Theorem:** If $L$ is a regular language over $\Sigma_1$ and $h^* : \Sigma_1^* \rightarrow \Sigma_2^*$ is a homomorphism, then $h^*(L)$ is a regular language.

• **Proof sketch:** Transform a regular expression for $L$ into a regular expression for $h^*(L)$ by replacing all characters in the regular expression with the value of $h$ applied to that character.

• Examples at the end of these slides.
The Big List of Closure Properties

- The regular languages are closed under
  - Union
  - Intersection
  - Complement
  - Concatenation
  - Kleene Closure
  - String Homomorphism
  - Plus a whole lot more!
The Limits of Regular Languages
Is every language regular?
An Important Observation
An Important Observation
An Important Observation

![Diagram of a state machine with transitions labeled by 0, 1, and 0, 1.]
An Important Observation
An Important Observation
An Important Observation

The diagram illustrates a finite automaton with states labeled $q_0$, $q_1$, $q_2$, $q_3$, and $q_4$. The transitions are labeled with input symbols 0 and 1. The start state is $q_0$, and the accepting state is $q_4$. The transitions are as follows:

- From $q_0$ to $q_1$ on input 0
- From $q_1$ to $q_0$ on input 0
- From $q_1$ to $q_2$ on input 1
- From $q_2$ to $q_0$ on input 0
- From $q_2$ to $q_3$ on input 1
- From $q_3$ to $q_4$ on input 1

The bottom line represents a sequence of inputs: 0 1 1 0 1 1 1 1.
An Important Observation

Start

$q_0 \rightarrow q_1 \rightarrow q_3 \rightarrow q_4$

$q_0 \rightarrow 0 \rightarrow q_1 \rightarrow 0 \rightarrow q_3 \rightarrow 1 \rightarrow q_4$

$q_0 \rightarrow 0 \rightarrow q_5 \rightarrow 0,1 \rightarrow q_2 \rightarrow 0 \rightarrow q_5 \rightarrow 0,1 \rightarrow q_2 \rightarrow 1 \rightarrow q_3 \rightarrow 1 \rightarrow q_4$

0 1 1 1 0 1 1 1
An Important Observation
An Important Observation
An Important Observation

![Diagram of a finite automaton with states $q_0$, $q_1$, $q_2$, $q_3$, $q_5$, and $q_4$. The transitions are labeled with 0, 1, or 0, 1. The start state is $q_0$, and $q_4$ is a final state. The input sequence is 0 1 1 1 0 1 1 1 1.](image)
An Important Observation
An Important Observation
An Important Observation
An Important Observation

start → q_0 → 0 → q_1 → 1 → q_2 → 0 → q_5 → 0, 1 → q_4

0, 1 → q_3 → 1 → q_4

0 1 1 1 0 1 1 1 1
An Important Observation
An Important Observation
An Important Observation
An Important Observation

[Diagram of a finite automaton with states q₀, q₁, q₂, q₅, q₄ and transitions labeled with inputs 0 and 1.]

0 1 1 1 0 1 1 1 1
An Important Observation

The diagram represents a finite automaton with states labeled as follows:

- Start state: $q_0$
- State $q_1$ transitions to $q_2$, $q_3$, and $q_5$ on input 0 and 1.
- State $q_2$ transitions to $q_5$ on input 0.
- State $q_3$ transitions to $q_4$ on input 1.
- State $q_5$ transitions back to itself on input 0, 1.

The transitions are labeled with the input symbols.
An Important Observation

Diagram:

- Start state: $q_0$
- States: $q_0, q_1, q_2, q_3, q_4$
- Transitions:
  - $q_0$ on 0 to $q_1$
  - $q_1$ on 1 to $q_5$
  - $q_1$ on 0 to $q_2$
  - $q_2$ on 1 to $q_3$
  - $q_3$ on 1 to $q_4$
  - $q_5$ on 0, 1 to itself

Symbolic Representation:

- $q_0$: 0
- $q_1$: 1
- $q_2$: 0, 1
- $q_3$: 1
- $q_4$: 1

Input Tape:

- Symbols: 0, 1
- Tape: 0 1 1 1 0 1 1 1 1

Transition Matrix:

<table>
<thead>
<tr>
<th></th>
<th>$q_0$</th>
<th>$q_1$</th>
<th>$q_2$</th>
<th>$q_3$</th>
<th>$q_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
An Important Observation
An Important Observation
An Important Observation

The diagram illustrates a finite automaton with states labeled $q_0, q_1, q_2, q_3,$ and $q_4$. The transitions are labeled with symbols 0 and 1. The start state is $q_0$, and the accepting states are $q_3$ and $q_4$. The transitions are as follows:

- $q_0$ transitions to $q_1$ on 0.
- $q_1$ transitions to $q_2$ on 1, to $q_3$ on 0, and back to $q_1$ on 0.
- $q_2$ transitions to $q_3$ on 1.
- $q_3$ transitions to $q_4$ on 1.

The states $q_3$ and $q_4$ are accepting states.
An Important Observation
Visiting Multiple States

• Let $D$ be a DFA with $n$ states.

• Any string $w$ accepted by $D$ that has length at least $n$ must visit some state twice.
  • Number of states visited is equal to the length of the string plus one.
  • By the pigeonhole principle, some state is duplicated.

• The substring of $w$ between those revisited states can be removed, duplicated, tripled, etc. without changing the fact that $D$ accepts $w$. 
Intuitively
Informally

- Let $L$ be a regular language.
- If we have a string $w \in L$ that is “sufficiently long,” then we can split the string into three pieces and “pump” the middle.
- We can write $w = xyz$ such that $xy^0z, xy^1z, xy^2z, \ldots, xy^n z, \ldots$ are all in $L$.
  - **Notation**: $y^n$ means “$n$ copies of $y$.”
The Weak Pumping Lemma

The Weak Pumping Lemma for Regular Languages states that

For any regular language $L$,

There exists a positive natural number $n$ such that

For any $w \in L$ with $|w| \geq n$,

There exists strings $x, y, z$ such that

For any natural number $i$,

\[ w = xyz, \]
\[ y \neq \varepsilon \]
\[ xy^iz \in L \]
The Weak Pumping Lemma

• The **Weak Pumping Lemma for Regular Languages** states that

\[ \forall \text{ regular language } L, \]

\[ \exists \text{ a positive natural number } n \text{ such that } \]

\[ \forall w \in L \text{ with } |w| \geq n, \]

\[ \exists \text{ strings } x, y, z \text{ such that } \]

\[ \forall \text{ natural number } i, \]

\[ w = xyz, \]

\[ y \neq \varepsilon \]

\[ xyz^i \in L \]
The Weak Pumping Lemma

- The **Weak Pumping Lemma for Regular Languages** states that:

  \[ \forall \text{ regular language } L, \exists \text{ a positive natural number } n \ such \ that \ \\
  \forall w \in L \ with |w| \geq n, \exists \text{ strings } x, y, z \ such \ that \ \\
  \forall \text{ natural number } i, w = xyz, y \neq \epsilon \ \\
  xy^iz \in L \]
The Weak Pumping Lemma

- The **Weak Pumping Lemma for Regular Languages** states that

  **For any** regular language $L$, there exists a positive natural number $n$ such that

  **For any** $w \in L$ with $|w| \geq n$, there exists strings $x, y, z$ such that

  **For any** natural number $i$,

  $$w = xyz,$$

  $$y \neq \varepsilon$$

  $$xy^iz \in L$$
The Weak Pumping Lemma

- The **Weak Pumping Lemma for Regular Languages** states that

  For any regular language $L$,

  There exists a positive natural number $n$ such that

  For any $w \in L$ with $|w| \geq n$,

  There exists strings $x, y, z$ such that

  For any natural number $i$,

  $w = xyz$,

  $y \neq \varepsilon$

  $xy^i z \in L$

  This number $n$ is sometimes called the pumping length.
The Weak Pumping Lemma

- The **Weak Pumping Lemma for Regular Languages** states that

  For any regular language \( L \),

  **There exists** a positive natural number \( n \) such that

  For any \( w \in L \) with \( |w| \geq n \),

  **There exists** strings \( x, y, z \) such that

  For any natural number \( i \),

  \[
  w = xyz, \quad y \neq \varepsilon \quad \text{ and } \quad xy^iz \in L
  \]

  Strings longer than the pumping length must have a special property.
The Weak Pumping Lemma

- The **Weak Pumping Lemma for Regular Languages** states that

  For any regular language \( L \),

  **There exists** a positive natural number \( n \) such that

  For any \( w \in L \) with \(|w| \geq n\),

  **There exists** strings \( x, y, z \) such that

  For any natural number \( i \),

  \[ w = xyz, \quad w \text{ can be broken into three pieces,} \]

  \[ y \neq \varepsilon \]

  \[ xy^iz \in L \]
The Weak Pumping Lemma

The Weak Pumping Lemma for Regular Languages states that

For any regular language $L$,

There exists a positive natural number $n$ such that

For any $w \in L$ with $|w| \geq n$,

There exists strings $x, y, z$ such that

For any natural number $i$,

$w = xyz$, $w$ can be broken into three pieces, $y \neq \varepsilon$ where the middle piece isn't empty, $xy^iz \in L$
The Weak Pumping Lemma

The **Weak Pumping Lemma for Regular Languages** states that

For any regular language $L$,

There exists a positive natural number $n$ such that

For any $w \in L$ with $|w| \geq n$,

There exists strings $x, y, z$ such that

For any natural number $i$,

$$w = xyz,$$

$w$ can be broken into three pieces,

$$y \neq \varepsilon$$

where the middle piece isn't empty,

$$xy^iz \in L$$

where the middle piece can be replicated zero or more times.
The Weak Pumping Lemma

- Let $\Sigma = \{0, 1\}$ and $L = \{ w \in \Sigma^* \mid w \text{ contains } 00 \text{ as a substring.} \}$

- Any string of length 3 or greater can be split into three pieces, the second of which can be “pumped.”

\[
\begin{array}{cccccc}
1 & 0 & 0 & 1 & 0
\end{array}
\]
The Weak Pumping Lemma

- Let $\Sigma = \{0, 1\}$ and $L = \{ w \in \Sigma^* \mid w$ contains $00$ as a substring. $\}$
- Any string of length 3 or greater can be split into three pieces, the second of which can be “pumped.”

\[
\begin{array}{c|c|c|c|c|c|c|c|c|c}
  & 1 & 0 & 0 & 1 & 0 \\
\end{array}
\]
The Weak Pumping Lemma

- Let $\Sigma = \{0, 1\}$ and $L = \{ w \in \Sigma^* | w$ contains 00 as a substring. $\}$

- Any string of length 3 or greater can be split into three pieces, the second of which can be “pumped.”

1 0 0 0
The Weak Pumping Lemma

• Let $\Sigma = \{0, 1\}$ and $L = \{ w \in \Sigma^* \mid w \text{ contains } 00 \text{ as a substring.} \}$

• Any string of length 3 or greater can be split into three pieces, the second of which can be “pumped.”

\[
\begin{array}{cccc}
1 & 0 & 0 & 1 & 0 \\
\end{array}
\]
The Weak Pumping Lemma

- Let $\Sigma = \{0, 1\}$ and $L = \{ w \in \Sigma^* \mid w$ contains 00 as a substring. $\}$
- Any string of length 3 or greater can be split into three pieces, the second of which can be “pumped.”

```
1 0 0 1 1 1 0
```
The Weak Pumping Lemma

- Let $\Sigma = \{0, 1\}$ and $L = \{ w \in \Sigma^* \mid w$ contains 00 as a substring. $\}$

- Any string of length 3 or greater can be split into three pieces, the second of which can be “pumped.”

```
1 0 0 1 1 1 1 0
```
The Weak Pumping Lemma

- Let $\Sigma = \{0, 1\}$ and $L = \{ w \in \Sigma^* | w$ contains $00$ as a substring. $\}$

- Any string of length 3 or greater can be split into three pieces, the second of which can be “pumped.”

\[
\begin{array}{ccc}
1 & 0 & 0 \\
\end{array}
\]
The Weak Pumping Lemma

- Let $\Sigma = \{0, 1\}$ and $L = \{ w \in \Sigma^* \mid w$ contains $00$ as a substring. $\}$

- Any string of length 3 or greater can be split into three pieces, the second of which can be “pumped.”

\[
\begin{array}{c|c|c}
1 & 0 & 0 \\
\end{array}
\]
The Weak Pumping Lemma

- Let $\Sigma = \{0, 1\}$ and $L = \{ w \in \Sigma^* \mid w$ contains $00$ as a substring. $\}$
- Any string of length 3 or greater can be split into three pieces, the second of which can be “pumped.”

The first piece is just the empty string! This is perfectly fine.
The Weak Pumping Lemma

- Let $\Sigma = \{0, 1\}$ and $L = \{ w \in \Sigma^* \mid w \text{ contains } 00 \text{ as a substring.} \}$

- Any string of length 3 or greater can be split into three pieces, the second of which can be “pumped.”

  1 0 0
The Weak Pumping Lemma

- Let $\Sigma = \{0, 1\}$ and $L = \{ w \in \Sigma^* \mid w \text{ contains } 00 \text{ as a substring.} \}$
- Any string of length 3 or greater can be split into three pieces, the second of which can be “pumped.”
The Weak Pumping Lemma

- Let $\Sigma = \{0, 1\}$ and $L = \{ w \in \Sigma^* \mid w \text{ contains } 00 \text{ as a substring.} \}$

- Any string of length 3 or greater can be split into three pieces, the second of which can be “pumped.”

\[
\begin{array}{c|c|c}
1 & 0 & 0 \\
\end{array}
\]
The Weak Pumping Lemma

- Let $\Sigma = \{0, 1\}$ and $L = \{w \in \Sigma^* \mid w$ contains 00 as a substring. }$

- Any string of length 3 or greater can be split into three pieces, the second of which can be "pumped."
The Weak Pumping Lemma

- Let $\Sigma = \{0, 1\}$ and $L = \{ w \in \Sigma^* | w$ contains $00$ as a substring. $\}$

- Any string of length 3 or greater can be split into three pieces, the second of which can be “pumped.”
The Weak Pumping Lemma

- Let $\Sigma = \{0, 1\}$ and $L = \{ w \in \Sigma^* \mid w \text{ contains } 00 \text{ as a substring. } \}$
- Any string of length 3 or greater can be split into three pieces, the second of which can be “pumped.”
The Weak Pumping Lemma

- Let $\Sigma = \{0, 1\}$ and $L = \{ w \in \Sigma^* \mid w$ contains 00 as a substring. \}$

- Any string of length 3 or greater can be split into three pieces, the second of which can be “pumped.”
The Weak Pumping Lemma

- Let $\Sigma = \{0, 1\}$ and $L = \{w \in \Sigma^* | w$ contains 00 as a substring. $\}$

- Any string of length 3 or greater can be split into three pieces, the second of which can be “pumped.”

\[1 \quad 1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1\]
The Weak Pumping Lemma

• Let $\Sigma = \{0, 1\}$ and $L = \{ w \in \Sigma^* \mid w$ contains $00$ as a substring. $\}$

• Any string of length 3 or greater can be split into three pieces, the second of which can be “pumped.”

1 1 0 0
The Weak Pumping Lemma

• Let $\Sigma = \{0, 1\}$ and $L = \{ w \in \Sigma^* \mid w$ contains 00 as a substring. $\}$

• Any string of length 3 or greater can be split into three pieces, the second of which can be “pumped.”

\[
\begin{array}{cccccc}
1 & 1 & 0 & 0 & 0 & 0 & 0 & 1
\end{array}
\]
The Weak Pumping Lemma

- Let $\Sigma = \{0, 1\}$ and $L = \{ w \in \Sigma^* \mid w \text{ contains } 00 \text{ as a substring.} \}$
- Any string of length 3 or greater can be split into three pieces, the second of which can be “pumped.”
The Weak Pumping Lemma

• Let $\Sigma = \{0, 1\}$ and $L = \{ w \in \Sigma^* \mid w$ contains $00$ as a substring. $\}$

• Any string of length 3 or greater can be split into three pieces, the second of which can be “pumped.”

1 1 0 0 0 0 1 0 0 1 0 0 0 1
The Weak Pumping Lemma
The Weak Pumping Lemma

- Let $\Sigma = \{0, 1\}$ and
  $L = \{ \varepsilon, 0, 1, 00, 01, 10, 11 \}$
The Weak Pumping Lemma

- Let $\Sigma = \{0, 1\}$ and $L = \{\varepsilon, 0, 1, 00, 01, 10, 11\}$
- Any string of length 3 or greater can be split into three pieces, the second of which can be “pumped.”
The Weak Pumping Lemma

- Let $\Sigma = \{0, 1\}$ and $L = \{\varepsilon, 0, 1, 00, 01, 10, 11\}$
- Any string of length 3 or greater can be split into three pieces, the second of which can be “pumped.”
Testing Equality

• The **equality problem** is defined as follows: 
  Given two strings \( x \) and \( y \), decide if \( x = y \).

• Let \( \Sigma = \{0, 1, \? \} \). We can encode the equality problem as a string of the form \( x\?y \).
  • “Is 001 equal to 110 ?” would be 001?110
  • “Is 11 equal to 11 ?” would be 11?11
  • “Is 110 equal to 110 ?” would be 110?110

• Let \( EQUAL = \{ w\?w \mid w \in \{0, 1\}* \} \)

• **Question**: Is \( EQUAL \) a regular language?
The Weak Pumping Lemma

- The **Weak Pumping Lemma for Regular Languages** states that

  **For any** regular language \( L \),

  **There exists** a positive natural number \( n \) such that

  **For any** \( w \in L \) with \( |w| \geq n \),

  **There exists** strings \( x, y, z \) such that

  **For any** natural number \( i \),

  \[ w = xyz, \quad w \text{ can be broken into three pieces,} \]

  \[ y \neq \varepsilon \quad \text{where the middle piece isn't empty,} \]

  \[ xy^iz \in L \quad \text{where the middle piece can be replicated zero or more times.} \]
Using the Weak Pumping Lemma

\[ EQUAL = \{ w?w \mid w \in \{0, 1\}^* \} \]
Using the Weak Pumping Lemma

\[ \text{EQUAL} = \{ \, w?w \mid w \in \{0, 1\}^* \, \} \]
Using the Weak Pumping Lemma

\[ \text{EQUAL} = \{ w^2w \mid w \in \{0, 1\}^* \} \]
Using the Weak Pumping Lemma

\[ \text{EQUAL} = \{ \, w?w \mid w \in \{0, 1\}^* \, \} \]
Using the Weak Pumping Lemma

\[ EQUAL = \{ w?w \mid w \in \{0, 1\}^* \} \]
Using the Weak Pumping Lemma

\[ \text{EQUAL} = \{ w \? w \mid w \in \{0, 1\}^* \} \]
Using the Weak Pumping Lemma

\[ EQUAL = \{ w?w \mid w \in \{0, 1\}^* \} \]
Using the Weak Pumping Lemma

\[ \text{EQUAL} = \{ w^2 \mid w \in \{0, 1\}^* \} \]
Using the Weak Pumping Lemma

\[ EQUAL = \{ \, w?w \mid w \in \{0, 1\}^* \, \} \]

| 0 | 0 | 0 | ? | 0 | 0 | 0 | 0 |
Using the Weak Pumping Lemma

\[ EQUAL = \{ w?w \mid w \in \{0, 1\}^* \} \]
Using the Weak Pumping Lemma

\[ EQUAL = \{ w?w \mid w \in \{0, 1\}* \} \]
Using the Weak Pumping Lemma

$$EQUAL = \{ \, w?w \mid w \in \{0, 1\}^* \, \}$$
What's Going On?

- The weak pumping lemma says that for “sufficiently long” strings, we should be able to pump some part of the string.
- We can't pump any part containing the ?, because we can't duplicate or remove it.
- We can't pump just one part of the string, because then the strings on opposite sides of the ? wouldn't match.
- Can we formally show that *EQUAL* is not regular?
Theorem:EQUAL is not regular.

Proof: By contradiction; assume that EQUAL is regular. Let \(n\) be the pumping length guaranteed by the weak pumping lemma. Let \(w = 0^n?0^n\). Then \(w \in \text{EQUAL}\) and \(|w| = 2n + 1 \geq n\). Thus by the weak pumping lemma, we can write \(w = xyz\) such that \(y \neq \varepsilon\) and for any \(i \in \mathbb{N}\), \(xy^iz \in \text{EQUAL}\). Then \(y\) cannot contain \(?\), since otherwise if we let \(i = 0\), then \(xy^iz = xz\) does not contain \(?\) and would not be in \(\text{EQUAL}\). So \(y\) is either completely to the left of the \(?\) or completely to the right of the \(?\). Let \(|y| = k\), so \(k > 0\). Since \(y\) is completely to the left or right of the \(?\), then \(y = 0^k\). Now, we consider two cases:

Case 1: \(y\) is to the left of the \(?\). Then \(xy_2z = 0^n+k?0^n \notin \text{EQUAL}\), contradicting the weak pumping lemma.

Case 2: \(y\) is to the right of the \(?\). Then \(xy_2z = 0^n?0^n+k \notin \text{EQUAL}\), contradicting the weak pumping lemma.

In either case we reach a contradiction, so our assumption was wrong. Thus EQUAL is not regular. ■
Theorem: \textit{EQUAL} is not regular.

Proof: By contradiction; assume that \textit{EQUAL} is regular.

For any regular language \( L \),

\textbf{There exists} a positive natural number \( n \) such that

For any \( w \in L \) with \( |w| \geq n \),

\textbf{There exists} strings \( x, y, z \) such that

For any natural number \( i \),

\[ w = xyz, \]

\( y \neq \varepsilon \)

\( xy^iz \in L \)
For any regular language $L$, there exists a positive natural number $n$ such that for any $w \in L$ with $|w| \geq n$, there exist strings $x, y, z$ such that for any natural number $i$, $w = xyz$, $y \neq \varepsilon$, and $xy^iz \in L$.

Theorem: EQUAL is not regular.
Proof: By contradiction; assume that EQUAL is regular.
Theorem: EQUAL is not regular.

Proof: By contradiction; assume that EQUAL is regular. Let $n$ be the pumping length guaranteed by the weak pumping lemma.

For any regular language $L$,
There exists a positive natural number $n$ such that
For any $w \in L$ with $|w| \geq n$,
There exists strings $x, y, z$ such that
For any natural number $i$,
    $w = xyz$,
    $y \neq \varepsilon$
    $xy^iz \in L$
Theorem: EQUAL is not regular.
Proof: By contradiction; assume that EQUAL is regular. Let $n$ be the pumping length guaranteed by the weak pumping lemma.
Theorem: EQUAL is not regular.

Proof: By contradiction; assume that EQUAL is regular. Let \( n \) be the pumping length guaranteed by the weak pumping lemma.

For any regular language \( L \),

There exists a positive natural number \( n \) such that

For any \( w \in L \) with \( |w| \geq n \),

There exists strings \( x, y, z \) such that

For any natural number \( i \),

\( w = xyz \),

\( y \neq \varepsilon \)

\( xy^iz \in L \)
Theorem: \textit{EQUAL} is not regular.

Proof: By contradiction; assume that \textit{EQUAL} is regular. Let \( n \) be the pumping length guaranteed by the weak pumping lemma.

For any regular language \( L \),
There exists a positive natural number \( n \) such that

For any \( w \in L \) with \( |w| \geq n \),
There exists strings \( x, y, z \) such that
For any natural number \( i \),
\[ w = xyz, \]
\[ y \neq \varepsilon \]
\[ xy^iz \in L \]

The hardest part of most proofs with the pumping lemma is choosing some string that we should be able to pump but cannot.
Theorem: \textit{EQUAL} is not regular.

Proof: By contradiction, assume that \textit{EQUAL} is regular. Let \( n \) be the pumping length guaranteed by the weak pumping lemma. Let \( w = 0^n ? 0^n \).
Theorem: $EQUAL$ is not regular.

Proof: By contradiction; assume that $EQUAL$ is regular. Let $n$ be the pumping length guaranteed by the weak pumping lemma. Let $w = 0^n?0^n$. For any regular language $L$, there exists a positive natural number $n$ such that:

For any $w \in L$ with $|w| \geq n$,

There exists strings $x$, $y$, $z$ such that:

For any natural number $i$,

$w = xyz$,

$y \neq \varepsilon$

$xy^iz \in L$
Theorem: \textit{EQUAL} is not regular.

Proof: By contradiction; assume that \textit{EQUAL} is regular. Let $n$ be the pumping length guaranteed by the weak pumping lemma. Let $w = 0^n?0^n$. Then $w \in \textit{EQUAL}$ and $|w| = 2n + 1 \geq n$. 

For any regular language $L$, there exists a positive natural number $n$ such that:

\[
\begin{align*}
\text{For any } w & \in L \text{ with } |w| \geq n, \\
\text{there exists strings } x, y, z \text{ such that } & \\
\text{for any } \\n\text{natural number } i, & \\
\text{w = xyz, } & \\
y \neq \varepsilon & \\
xy^iz \in L & 
\end{align*}
\]
Theorem: $EQUAL$ is not regular.

Proof: By contradiction; assume that $EQUAL$ is regular. Let $n$ be the pumping length guaranteed by the weak pumping lemma. Let $w = 0^n?0^n$. Then $w \in EQUAL$ and $|w| = 2n + 1 \geq n$. For any regular language $L$,

There exists a positive natural number $n$ such that For any $w \in L$ with $|w| \geq n$, There exists strings $x$, $y$, $z$ such that For any natural number $i$, $w = xyz$, $y \neq \varepsilon$ $xy^iz \in L$
Theorem: \(\text{EQUAL}\) is not regular.

Proof: By contradiction; assume that \(\text{EQUAL}\) is regular. Let \(n\) be the pumping length guaranteed by the weak pumping lemma. Let \(w = 0^n?0^n\). Then \(w \in \text{EQUAL}\) and \(|w| = 2n + 1 \geq n\).
Theorem: \( \text{EQUAL} \) is not regular.

Proof: By contradiction; assume that \( \text{EQUAL} \) is regular. Let \( n \) be the pumping length guaranteed by the weak pumping lemma. Let \( w = 0^n \rightarrow 0^n \). Then \( w \in \text{EQUAL} \) and \( |w| = 2n + 1 \geq n \). Thus by the weak pumping lemma, we can write \( w = xyz \) such that \( y \neq \varepsilon \) and for any \( i \in \mathbb{N} \), \( xy^iz \in \text{EQUAL} \).
For any regular language $L$, there exists a positive natural number $n$ such that for any $w \in L$ with $|w| \geq n$, there exists strings $x$, $y$, $z$ such that for any natural number $i$, $w = xyz$, $y \neq \varepsilon$, $xy^iz \in L$.

**Theorem:** $EQUAL$ is not regular.

**Proof:** By contradiction; assume that $EQUAL$ is regular. Let $n$ be the pumping length guaranteed by the weak pumping lemma. Let $w = 0^n?0^n$. Then $w \in EQUAL$ and $|w| = 2n + 1 \geq n$. Thus by the weak pumping lemma, we can write $w = xyz$ such that $y \neq \varepsilon$ and for any $i \in \mathbb{N}$, $xy^iz \in EQUAL$. 
For any regular language $L$, there exists a positive natural number $n$ such that

For any $w \in L$ with $|w| \geq n$,

there exists strings $x, y, z$ such that

for any natural number $i$,

$w = xyz$, $y \neq \varepsilon$,

$xy^iz \in L$

**Theorem:** $EQLA$ is not regular.

**Proof:** By contradiction; assume that $EQLA$ is regular. Let $n$ be the pumping length guaranteed by the weak pumping lemma. Let $w = 0^n?0^n$. Then $w \in EQLA$ and $|w| = 2n + 1 \geq n$. Thus by the weak pumping lemma, we can write $w = xyz$ such that $y \neq \varepsilon$ and for any $i \in N$, $xy^iz \in EQLA$.

At this point, we have some string that we should be able to split into pieces and pump. The rest of the proof shows that no matter what choice we made, the middle can't be pumped.
For any regular language L,
   There exists a positive natural number n such that
      For any \( w \in L \) with \( |w| \geq n \),
         There exists strings x, y, z such that
            For any natural number i,
               \( w = xyz \),
               y \neq \varepsilon
               xy^iz \in L

Theorem: EQUAL is not regular.

Proof: By contradiction; assume that EQUAL is regular. Let n be the pumping length guaranteed by the weak pumping lemma. Let \( w = 0^n?0^n \). Then \( w \in EQUAL \) and \( |w| = 2n + 1 \geq n \). Thus by the weak pumping lemma, we can write \( w = xyz \) such that \( y \neq \varepsilon \) and for any \( i \in \mathbb{N} \), \( xy^iz \in EQUAL \).
**Theorem:** EQUAL is not regular.

**Proof:** By contradiction; assume that EQUAL is regular. Let $n$ be the pumping length guaranteed by the weak pumping lemma. Let $w = 0^n?0^n$. Then $w \in$ EQUAL and $|w| = 2n + 1 \geq n$. Thus by the weak pumping lemma, we can write $w = xyz$ such that $y \neq \varepsilon$ and for any $i \in \mathbb{N}$, $xy^i z \in$ EQUAL. Then $y$ cannot contain $?$, since otherwise if we let $i = 0$, then $xy^i z = xz$ does not contain $?$ and would not be in EQUAL.
Theorem: \textit{EQUAL} is not regular.

Proof: By contradiction; assume that \textit{EQUAL} is regular. Let $n$ be the pumping length guaranteed by the weak pumping lemma. Let $w = 0^n?0^n$. Then $w \in \textit{EQUAL}$ and $|w| = 2n + 1 \geq n$. Thus by the weak pumping lemma, we can write $w = xyz$ such that $y \neq \varepsilon$ and for any $i \in \mathbb{N}$, $xy^iz \in \textit{EQUAL}$. Then $y$ cannot contain $?$, since otherwise if we let $i = 0$, then $xy^iz = xz$ does not contain $?$ and would not be in \textit{EQUAL}. So $y$ is either completely to the left of the $?$ or completely to the right of the $?$. 

For any regular language $L$,

\textbf{There exists} a positive natural number $n$ such that

For any $w \in L$ with $|w| \geq n$,

\textbf{There exists} strings $x, y, z$ such that

For any natural number $i$,

\begin{align*}
    w &= xyz, \\
    y &\neq \varepsilon \\
    xy^iz &\in L
\end{align*}
Theorem: \( \text{EQUAL} \) is not regular.

Proof: By contradiction; assume that \( \text{EQUAL} \) is regular. Let \( n \) be the pumping length guaranteed by the weak pumping lemma. Let \( w = 0^n?0^n \). Then \( w \in \text{EQUAL} \) and \( |w| = 2n + 1 \geq n \). Thus by the weak pumping lemma, we can write \( w = xyz \) such that \( y \neq \varepsilon \) and for any \( i \in \mathbb{N} \), \( xy^iz \in \text{EQUAL} \). Then \( y \) cannot contain \( ? \), since otherwise if we let \( i = 0 \), then \( xy^iz = xz \) does not contain \( ? \) and would not be in \( \text{EQUAL} \). So \( y \) is either completely to the left of the \( ? \) or completely to the right of the \( ? \). Let \( |y| = k \), so \( k > 0 \).
Theorem: \( \text{EQUAL} \) is not regular.

Proof: By contradiction; assume that \( \text{EQUAL} \) is regular. Let \( n \) be the pumping length guaranteed by the weak pumping lemma. Let \( w = 0^n?0^n \). Then \( w \in \text{EQUAL} \) and \( |w| = 2n + 1 \geq n \). Thus by the weak pumping lemma, we can write \( w = xyz \) such that \( y \neq \varepsilon \) and for any \( i \in \mathbb{N} \), \( xy^i z \in \text{EQUAL} \). Then \( y \) cannot contain \( ? \), since otherwise if we let \( i = 0 \), then \( xy^i z = xz \) does not contain \( ? \) and would not be in \( \text{EQUAL} \). So \( y \) is either completely to the left of the \( ? \) or completely to the right of the \( ? \). Let \( |y| = k \), so \( k > 0 \). Since \( y \) is completely to the left or right of the \( ? \), then \( y = 0^k \).
For any regular language L,
There exists a positive natural number n such that
For any \( w \in L \) with \( |w| \geq n \),
There exists strings \( x, y, z \) such that
For any natural number \( i \),
\[
\begin{align*}
  w &= xyz, \\
  y &\neq \varepsilon \\
  xy^iz &\in L
\end{align*}
\]

**Theorem:** \( EQUAL \) is not regular.

**Proof:** By contradiction; assume that \( EQUAL \) is regular. Let \( n \) be the pumping length guaranteed by the weak pumping lemma. Let \( w = 0^n?0^n \). Then \( w \in EQUAL \) and \( |w| = 2n + 1 \geq n \). Thus by the weak pumping lemma, we can write \( w = xyz \) such that \( y \neq \varepsilon \) and for any \( i \in \mathbb{N} \), \( xy^iz \in EQUAL \). Then \( y \) cannot contain \(?\), since otherwise if we let \( i = 0 \), then \( xy^iz = xz \) does not contain \(?\) and would not be in \( EQUAL \). So \( y \) is either completely to the left of the \(?\) or completely to the right of the \(?\). Let \( |y| = k \), so \( k > 0 \). Since \( y \) is completely to the left or right of the \(?\), then \( y = 0^k \). Now, we consider two cases:

**Case 1:** \( y \) is to the left of the \(?\).

**Case 2:** \( y \) is to the right of the \(?\).
For any regular language L,
There exists a positive natural number n such that
For any \( w \in L \) with \( |w| \geq n \),
There exists strings x, y, z such that
For any natural number i,
\[
\begin{align*}
    w &= xyz, \\
    y &\neq \varepsilon \\
    xy^iz &\in L
\end{align*}
\]

**Theorem:** EQUAŁ is not regular.

**Proof:** By contradiction; assume that EQUAŁ is regular. Let \( n \) be the pumping length guaranteed by the weak pumping lemma. Let \( w = 0^n?0^n \). Then \( w \in EQUAŁ \) and \( |w| = 2n + 1 \geq n \). Thus by the weak pumping lemma, we can write \( w = xyz \) such that \( y \neq \varepsilon \) and for any \( i \in \mathbb{N} \), \( xy^iz \in EQUAŁ \). Then \( y \) cannot contain \( ? \), since otherwise if we let \( i = 0 \), then \( xy^iz = xz \) does not contain \( ? \) and would not be in EQUAŁ. So \( y \) is either completely to the left of the \( ? \) or completely to the right of the \( ? \). Let \( |y| = k \), so \( k > 0 \). Since \( y \) is completely to the left or right of the \( ? \), then \( y = 0^k \). Now, we consider two cases:

**Case 1:** \( y \) is to the left of the \( ? \). Then \( xy^2z = 0^{n+k}?0^n \notin EQUAŁ \), contradicting the weak pumping lemma.

**Case 2:** \( y \) is to the right of the \( ? \).
Theorem: EQUAL is not regular.

Proof: By contradiction; assume that EQUAL is regular. Let \( n \) be the pumping length guaranteed by the weak pumping lemma. Let \( w = 0^n?0^n \). Then \( w \in \text{EQUAL} \) and \( |w| = 2n + 1 \geq n \). Thus by the weak pumping lemma, we can write \( w = xyz \) such that \( y \neq \varepsilon \) and for any \( i \in \mathbb{N} \), \( xy^iz \in \text{EQUAL} \). Then \( y \) cannot contain \(?\), since otherwise if we let \( i = 0 \), then \( xy^iz = xz \) does not contain \( ? \) and would not be in \( \text{EQUAL} \). So \( y \) is either completely to the left of the \( ? \) or completely to the right of the \( ? \). Let \( |y| = k \), so \( k > 0 \). Since \( y \) is completely to the left or right of the \( ? \), then \( y = 0^k \). Now, we consider two cases:

**Case 1:** \( y \) is to the left of the \( ? \). Then \( xy^2z = 0^{n+k}?0^n \notin \text{EQUAL} \), contradicting the weak pumping lemma.

**Case 2:** \( y \) is to the right of the \( ? \). Then \( xy^2z = 0^n?0^{n+k} \notin \text{EQUAL} \), contradicting the weak pumping lemma.
For any regular language $L$, there exists a positive natural number $n$ such that for any $w \in L$ with $|w| \geq n$, there exists strings $x, y, z$ such that for any natural number $i$,

$$w = xyz,$$

$$y \neq \varepsilon$$

$$xy^iz \in L$$

**Theorem:** $EQUAL$ is not regular.

**Proof:** By contradiction; assume that $EQUAL$ is regular. Let $n$ be the pumping length guaranteed by the weak pumping lemma. Let $w = 0^n?0^n$. Then $w \in EQUAL$ and $|w| = 2n + 1 \geq n$. Thus by the weak pumping lemma, we can write $w = xyz$ such that $y \neq \varepsilon$ and for any $i \in \mathbb{N}$, $xy^iz \in EQUAL$. Then $y$ cannot contain $?$, since otherwise if we let $i = 0$, then $xy^iz = xz$ does not contain $?$ and would not be in $EQUAL$. So $y$ is either completely to the left of the $?$ or completely to the right of the $?$. Let $|y| = k$, so $k > 0$. Since $y$ is completely to the left or right of the $?$, then $y = 0^k$. Now, we consider two cases:

**Case 1:** $y$ is to the left of the $?$. Then $xy^2z = 0^n?0^{n+k} \notin EQUAL$, contradicting the weak pumping lemma.

**Case 2:** $y$ is to the right of the $?$. Then $xy^2z = 0^n?0^n \notin EQUAL$, contradicting the weak pumping lemma.

In either case we reach a contradiction, so our assumption was wrong.
Theorem: \textit{EQUAL} is not regular.

Proof: By contradiction; assume that \textit{EQUAL} is regular. Let \( n \) be the pumping length guaranteed by the weak pumping lemma. Let \( w = 0^n?0^n \). Then \( w \in \text{EQUAL} \) and \(|w| = 2n + 1 \geq n\). Thus by the weak pumping lemma, we can write \( w = xyz \) such that \( y \neq \varepsilon \) and for any \( i \in \mathbb{N} \), \( xy^iz \in \text{EQUAL} \). Then \( y \) cannot contain \( ? \), since otherwise if we let \( i = 0 \), then \( xy^iz = xz \) does not contain \( ? \) and would not be in \text{EQUAL}. So \( y \) is either completely to the left of the \( ? \) or completely to the right of the \( ? \). Let \(|y| = k \), so \( k > 0 \). Since \( y \) is completely to the left or right of the \( ? \), then \( y = 0^k \). Now, we consider two cases:

\textit{Case 1:} \( y \) is to the left of the \( ? \). Then \( xy^2z = 0^{n+k}?0^n \notin \text{EQUAL} \), contradicting the weak pumping lemma.

\textit{Case 2:} \( y \) is to the right of the \( ? \). Then \( xy^2z = 0^n?0^{n+k} \notin \text{EQUAL} \), contradicting the weak pumping lemma.

In either case we reach a contradiction, so our assumption was wrong. Thus \textit{EQUAL} is not regular.
**Theorem:** $EQUAL$ is not regular.

**Proof:** By contradiction; assume that $EQUAL$ is regular. Let $n$ be the pumping length guaranteed by the weak pumping lemma. Let $w = 0^n?0^n$. Then $w \in EQUAL$ and $|w| = 2n + 1 \geq n$. Thus by the weak pumping lemma, we can write $w = xyz$ such that $y \neq \varepsilon$ and for any $i \in \mathbb{N}$, $xy^iz \in EQUAL$. Then $y$ cannot contain $?$, since otherwise if we let $i = 0$, then $xy^0z = xz$ does not contain $?$ and would not be in $EQUAL$. So $y$ is either completely to the left of the $?$ or completely to the right of the $?$. Let $|y| = k$, so $k > 0$. Since $y$ is completely to the left or right of the $?$, then $y = 0^k$. Now, we consider two cases:

**Case 1:** $y$ is to the left of the $?$. Then $xy^2z = 0^{n+k}0^n \notin EQUAL$, contradicting the weak pumping lemma.

**Case 2:** $y$ is to the right of the $?$. Then $xy^2z = 0^n0^{n+k} \notin EQUAL$, contradicting the weak pumping lemma.

In either case we reach a contradiction, so our assumption was wrong. Thus $EQUAL$ is not regular. ■
Nonregular Languages

- The weak pumping lemma describes a property common to all regular languages.
- Any language \( L \) which does not have this property cannot be regular.
- What other languages can we find that are not regular?
A Canonical Nonregular Language

• Consider the language \( L = \{ 0^n1^n \mid n \in \mathbb{N} \} \).
  
  \[ L = \{ \varepsilon, 01, 0011, 000111, 00001111, \ldots \} \]

• \( L \) is a classic example of a nonregular language.

• Intuitively: If you have only finitely many states in a DFA, you can't "remember" an arbitrary number of 0s.

• How would we prove that \( L \) is nonregular?
The Pumping Lemma as a Game

- The weak pumping lemma can be thought of as a game between you and an adversary.
- **You win** if you can prove that the pumping lemma fails.
- **The adversary wins** if the adversary can make a choice for which the pumping lemma succeeds.
- The game goes as follows:
  - The adversary chooses a pumping length \( n \).
  - You choose a string \( w \) with \(|w| \geq n\) and \( w \in L \).
  - The adversary breaks it into \( x, y, \) and \( z \).
  - You choose an \( i \) such that \( xy^iz \notin L \) (if you can't, you lose!)
The Pumping Lemma Game

ADVERSARY

YOU
The Pumping Lemma Game

ADVERSARY

Maliciously choose pumping length $n$.

YOU
### The Pumping Lemma Game

<table>
<thead>
<tr>
<th>ADVERSARY</th>
<th>YOU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maliciously choose pumping length ( n ).</td>
<td>Cleverly choose a string ( w \in L, \</td>
</tr>
</tbody>
</table>
The Pumping Lemma Game

<table>
<thead>
<tr>
<th><strong>ADVERSARY</strong></th>
<th><strong>YOU</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Maliciously choose pumping length $n$.</td>
<td>[ w \in L, ,</td>
</tr>
<tr>
<td>Maliciously split [ w = xyz, , y \neq \varepsilon ]</td>
<td>[ w \in L, ,</td>
</tr>
</tbody>
</table>
## The Pumping Lemma Game

<table>
<thead>
<tr>
<th>ADVERSARY</th>
<th>YOU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maliciously choose pumping length $n$.</td>
<td>Cleverly choose a string $w \in L$, $</td>
</tr>
<tr>
<td>Maliciously split $w = xyz$, $y \neq \varepsilon$</td>
<td>Cleverly choose $i$ such that $xy^iz \notin L$</td>
</tr>
</tbody>
</table>
The Pumping Lemma Game

**ADVERSARY**

Maliciously choose pumping length $n$.

Maliciously split $w = xyz$, $y \neq \varepsilon$

Grrr! Aaaargh!

**YOU**

Cleverly choose a string $w \in L$, $|w| \geq n$

Cleverly choose $i$ such that $xy^iz \notin L$
The Pumping Lemma Game

\[ L = \{ 0^n 1^n \mid n \in \mathbb{N} \} \]

**ADVERSARY**
- Maliciously choose pumping length \( n \).
- Maliciously split \( w = xyz, y \neq \varepsilon \).
- Grrr! Aaaargh!

**YOU**
- Cleverly choose a string \( w \in L, |w| \geq n \).
- Cleverly choose \( i \) such that \( xy^i z \notin L \).
The Pumping Lemma Game

\[ L = \{ 0^n1^n \mid n \in \mathbb{N} \} \]

**ADVERSARY**
- Maliciously choose pumping length \( n \).
- Maliciously split \( w = xyz, y \neq \varepsilon \).
- Grrr! Aaaargh!

**YOU**
- Cleverly choose a string \( w \in L, |w| \geq n \).
- Cleverly choose \( i \) such that \( xy^iz \notin L \).

Try your best!
Theorem: $L = \{ 0^n1^n \mid n \in \mathbb{N} \}$ is not regular.
Theorem: $L = \{ 0^n1^n | n \in \mathbb{N} \}$ is not regular.
Proof: By contradiction; assume $L$ is regular.
**Theorem:** $L = \{0^n1^n \mid n \in \mathbb{N}\}$ is not regular.

**Proof:** By contradiction; assume $L$ is regular. Let $n$ be the pumping length guaranteed by the weak pumping lemma.

Consider the string $w = 0^n1^n$. Then $|w| = 2n \geq n$ and $w \in L$, so we can write $w = xyz$ such that $y \neq \varepsilon$ and for any $i \in \mathbb{N}$, we have $xy^iz \in L$.

We consider three cases:

- **Case 1:** $y$ consists solely of $0$s. Then $xy^0z = xz = 0^n1^n - |y|$, and since $|y| > 0$, $xz \notin L$.

- **Case 2:** $y$ consists solely of $1$s. Then $xy^0z = xz = 0^n1^n - |y|$, and since $|y| > 0$, $xz \notin L$.

- **Case 3:** $y$ consists of $k > 0$ $0$s followed by $m > 0$ $1$s. Then $xy^2z$ has the form $0^n1^m0^k1^n$, so $xy^2z \notin L$.

In all three cases we reach a contradiction, so our assumption was wrong and $L$ is not regular. ■
Theorem: \( L = \{ 0^n1^n \mid n \in \mathbb{N} \} \) is not regular.

Proof: By contradiction; assume \( L \) is regular. Let \( n \) be the pumping length guaranteed by the weak pumping lemma. Consider the string \( w = 0^n1^n \).

In all three cases we reach a contradiction, so our assumption was wrong and \( L \) is not regular. ■
**Theorem:** $L = \{ 0^n1^n \mid n \in \mathbb{N} \}$ is not regular.

**Proof:** By contradiction; assume $L$ is regular. Let $n$ be the pumping length guaranteed by the weak pumping lemma. Consider the string $w = 0^n1^n$. Then $|w| = 2n \geq n$ and $w \in L$, so we can write $w = xyz$ such that $y \neq \varepsilon$ and for any $i \in \mathbb{N}$, we have $xy^iz \in L$. In all three cases we reach a contradiction, so our assumption was wrong and $L$ is not regular. ■
**Theorem:** \( L = \{ 0^n1^n \mid n \in \mathbb{N} \} \) is not regular.

**Proof:** By contradiction; assume \( L \) is regular. Let \( n \) be the pumping length guaranteed by the weak pumping lemma. Consider the string \( w = 0^n1^n \). Then \(|w| = 2n \geq n\) and \( w \in L \), so we can write \( w = xyz \) such that \( y \neq \varepsilon \) and for any \( i \in \mathbb{N} \), we have \( xy^i z \in L \). We consider three cases:
Theorem: $L = \{ 0^n1^n \mid n \in \mathbb{N} \}$ is not regular.

Proof: By contradiction; assume $L$ is regular. Let $n$ be the pumping length guaranteed by the weak pumping lemma. Consider the string $w = 0^n1^n$. Then $|w| = 2n \geq n$ and $w \in L$, so we can write $w = xyz$ such that $y \neq \varepsilon$ and for any $i \in \mathbb{N}$, we have $xy^iz \in L$. We consider three cases:

Case 1: $y$ consists solely of 0s.
Theorem: $L = \{ \ 0^n1^n \mid n \in \mathbb{N} \ \}$ is not regular.

Proof: By contradiction; assume $L$ is regular. Let $n$ be the pumping length guaranteed by the weak pumping lemma. Consider the string $w = 0^n1^n$. Then $|w| = 2n \geq n$ and $w \in L$, so we can write $w = xyz$ such that $y \neq \varepsilon$ and for any $i \in \mathbb{N}$, we have $xy^iz \in L$. We consider three cases:

Case 1: $y$ consists solely of 0s. Then

$$xy^0z = xz = 0^{n-|y|}1^n,$$

and since $|y| > 0$, $xz \notin L$. 


Theorem: $L = \{0^n1^n \mid n \in \mathbb{N}\}$ is not regular.

Proof: By contradiction; assume $L$ is regular. Let $n$ be the pumping length guaranteed by the weak pumping lemma. Consider the string $w = 0^n1^n$. Then $|w| = 2n \geq n$ and $w \in L$, so we can write $w = xyz$ such that $y \neq \varepsilon$ and for any $i \in \mathbb{N}$, we have $xy^iz \in L$. We consider three cases:

Case 1: $y$ consists solely of 0s. Then $xy^0z = xz = 0^{n-|y|}1^n$, and since $|y| > 0$, $xz \notin L$.

Case 2: $y$ consists solely of 1s.

Case 3: $y$ consists of $k > 0$ 0s followed by $m > 0$ 1s. Then $xy^2z$ has the form $0^n1^m0^k1^n$, so $xy^2z \notin L$. In all three cases we reach a contradiction, so our assumption was wrong and $L$ is not regular. ■
Theorem: \( L = \{ \, 0^n1^n \mid n \in \mathbb{N} \, \} \) is not regular.

Proof: By contradiction; assume \( L \) is regular. Let \( n \) be the pumping length guaranteed by the weak pumping lemma. Consider the string \( w = 0^n1^n \). Then \( |w| = 2n \geq n \) and \( w \in L \), so we can write \( w = xyz \) such that \( y \neq \varepsilon \) and for any \( i \in \mathbb{N} \), we have \( xy^iz \in L \). We consider three cases:

Case 1: \( y \) consists solely of 0s. Then 
\[ xy^0z = xz = 0^{n-|y|}1^n, \] and since \( |y| > 0 \), \( xz \notin L \).

Case 2: \( y \) consists solely of 1s. Then 
\[ xy^0z = xz = 0^n1^{n-|y|}, \] and since \( |y| > 0 \), \( xz \notin L \).
Theorem: $L = \{ 0^n1^n \mid n \in \mathbb{N} \}$ is not regular.

Proof: By contradiction; assume $L$ is regular. Let $n$ be the pumping length guaranteed by the weak pumping lemma. Consider the string $w = 0^n1^n$. Then $|w| = 2n \geq n$ and $w \in L$, so we can write $w = xyz$ such that $y \neq \varepsilon$ and for any $i \in \mathbb{N}$, we have $xy^i z \in L$. We consider three cases:

Case 1: $y$ consists solely of 0s. Then $xy^0 z = xz = 0^{n-|y|}1^n$, and since $|y| > 0$, $xz \notin L$.

Case 2: $y$ consists solely of 1s. Then $xy^0 z = xz = 0^n1^{n-|y|}$, and since $|y| > 0$, $xz \notin L$.

Case 3: $y$ consists of $k > 0$ 0s followed by $m > 0$ 1s.
Theorem: $L = \{ 0^n1^n \mid n \in \mathbb{N} \}$ is not regular.

Proof: By contradiction; assume $L$ is regular. Let $n$ be the pumping length guaranteed by the weak pumping lemma. Consider the string $w = 0^n1^n$. Then $|w| = 2n \geq n$ and $w \in L$, so we can write $w = xyz$ such that $y \neq \varepsilon$ and for any $i \in \mathbb{N}$, we have $xy^iz \in L$.

We consider three cases:

Case 1: $y$ consists solely of 0s. Then $x0^0z = xz = 0^n1^n - |y|$, and since $|y| > 0$, $xz \notin L$.

Case 2: $y$ consists solely of 1s. Then $x0^0z = xz = 0^n1^n - |y|$, and since $|y| > 0$, $xz \notin L$.

Case 3: $y$ consists of $k > 0$ 0s followed by $m > 0$ 1s.
Theorem: $L = \{ 0^n1^n \mid n \in \mathbb{N} \}$ is not regular.

Proof: By contradiction; assume $L$ is regular. Let $n$ be the pumping length guaranteed by the weak pumping lemma. Consider the string $w = 0^n1^n$. Then $|w| = 2n \geq n$ and $w \in L$, so we can write $w = xyz$ such that $y \neq \varepsilon$ and for any $i \in \mathbb{N}$, we have $xy^iz \in L$. We consider three cases:

**Case 1:** $y$ consists solely of 0s. Then $xy^0z = xz = 0^{n-|y|}1^n$, and since $|y| > 0$, $xz \notin L$.

**Case 2:** $y$ consists solely of 1s. Then $xy^0z = xz = 0^n1^{n-|y|}$, and since $|y| > 0$, $xz \notin L$.

**Case 3:** $y$ consists of $k > 0$ 0s followed by $m > 0$ 1s.
Theorem: $L = \{ 0^n1^n | n \in \mathbb{N} \}$ is not regular.

Proof: By contradiction; assume $L$ is regular. Let $n$ be the pumping length guaranteed by the weak pumping lemma. Consider the string $w = 0^n1^n$. Then $|w| = 2n \geq n$ and $w \in L$, so we can write $w = xyz$ such that $y \neq \varepsilon$ and for any $i \in \mathbb{N}$, we have $xy^iz \in L$. We consider three cases:

Case 1: $y$ consists solely of 0s. Then $xy^0z = xz = 0^{n-|y|}1^n$, and since $|y| > 0$, $xz \notin L$.

Case 2: $y$ consists solely of 1s. Then $xy^0z = xz = 0^n1^{n-|y|}$, and since $|y| > 0$, $xz \notin L$.

Case 3: $y$ consists of $k > 0$ 0s followed by $m > 0$ 1s.
Theorem: $L = \{ 0^n1^n \mid n \in \mathbb{N} \}$ is not regular.

Proof: By contradiction; assume $L$ is regular. Let $n$ be the pumping length guaranteed by the weak pumping lemma. Consider the string $w = 0^n1^n$. Then $|w| = 2n \geq n$ and $w \in L$, so we can write $w = xyz$ such that $y \neq \varepsilon$ and for any $i \in \mathbb{N}$, we have $xy^iz \in L$. We consider three cases:

Case 1: $y$ consists solely of 0s. Then $xy^0z = xz = 0^{n-|y|}1^n$, and since $|y| > 0$, $xz \notin L$.

Case 2: $y$ consists solely of 1s. Then $xy^0z = xz = 0^n1^{n-|y|}$, and since $|y| > 0$, $xz \notin L$.

Case 3: $y$ consists of $k > 0$ 0s followed by $m > 0$ 1s.

0 0 0 0 0 0 1 1 1 1 1 0 0 1 1 1 1 1 1 1
Theorem: \( L = \{ \, 0^n1^n \mid n \in \mathbb{N} \, \} \) is not regular.

Proof: By contradiction; assume \( L \) is regular. Let \( n \) be the pumping length guaranteed by the weak pumping lemma. Consider the string \( w = 0^n1^n \). Then \( |w| = 2n \geq n \) and \( w \in L \), so we can write \( w = xyz \) such that \( y \neq \varepsilon \) and for any \( i \in \mathbb{N} \), we have \( xy^iz \in L \).

We consider three cases:

**Case 1:** \( y \) consists solely of \( 0 \)s. Then \( xy^0z = xz = 0^{n-|y|}1^n \), and since \( |y| > 0 \), \( xz \notin L \).

**Case 2:** \( y \) consists solely of \( 1 \)s. Then \( xy^0z = xz = 0^n1^{n-|y|} \), and since \( |y| > 0 \), \( xz \notin L \).

**Case 3:** \( y \) consists of \( k > 0 \) \( 0 \)s followed by \( m > 0 \) \( 1 \)s. Then \( xy^2z \) has the form \( 0^n1^m0^k1^n \), so \( xy^2z \notin L \).
Theorem: $L = \{ 0^n1^n \mid n \in \mathbb{N} \}$ is not regular.

Proof: By contradiction; assume $L$ is regular. Let $n$ be the pumping length guaranteed by the weak pumping lemma. Consider the string $w = 0^n1^n$. Then $|w| = 2n \geq n$ and $w \in L$, so we can write $w = xyz$ such that $y \neq \varepsilon$ and for any $i \in \mathbb{N}$, we have $xy^iz \in L$. We consider three cases:

**Case 1:** $y$ consists solely of 0s. Then $xy^0z = xz = 0^{n-|y|}1^n$, and since $|y| > 0$, $xz \notin L$.

**Case 2:** $y$ consists solely of 1s. Then $xy^0z = xz = 0^n1^{n-|y|}$, and since $|y| > 0$, $xz \notin L$.

**Case 3:** $y$ consists of $k > 0$ 0s followed by $m > 0$ 1s. Then $xy^2z$ has the form $0^n1^m0^k1^n$, so $xy^2z \notin L$.

In all three cases we reach a contradiction, so our assumption was wrong and $L$ is not regular.
Theorem: $L = \{ 0^n1^n \mid n \in \mathbb{N} \}$ is not regular.

Proof: By contradiction; assume $L$ is regular. Let $n$ be the pumping length guaranteed by the weak pumping lemma. Consider the string $w = 0^n1^n$. Then $|w| = 2n \geq n$ and $w \in L$, so we can write $w = xyz$ such that $y \neq \varepsilon$ and for any $i \in \mathbb{N}$, we have $xy^iz \in L$. We consider three cases:

**Case 1:** $y$ consists solely of 0s. Then $xy^0z = xz = 0^{n-|y|}1^n$, and since $|y| > 0$, $xz \notin L$.

**Case 2:** $y$ consists solely of 1s. Then $xy^0z = xz = 0^n1^{n-|y|}$, and since $|y| > 0$, $xz \notin L$.

**Case 3:** $y$ consists of $k > 0$ 0s followed by $m > 0$ 1s. Then $xy^2z$ has the form $0^m1^k0^n1^n$, so $xy^2z \notin L$.

In all three cases we reach a contradiction, so our assumption was wrong and $L$ is not regular. ■
Counting Symbols

- Consider the alphabet $\Sigma = \{ 0, 1 \}$ and the language

$BALANCE = \{ w \in \Sigma^* \mid w \text{ contains an equal number of } 0\text{s and } 1\text{s.} \}$

- For example:
  - $01 \in BALANCE$
  - $110010 \in BALANCE$
  - $11011 \notin BALANCE$

- **Question:** Is $BALANCE$ a regular language?
$BALANCE$ and the Weak Pumping Lemma

$BALANCE = \{ w \in \{0, 1\}^* | w \text{ contains an equal number of } 0\text{s and } 1\text{s.} \}$
$BALANCE$ and the Weak Pumping Lemma

$BALANCE = \{ \ w \in \{0, 1\}^* \mid w \text{ contains an equal number of 0s and 1s.} \}$
\( \text{BALANCE and the Weak Pumping Lemma} \)

\[ \text{BALANCE} = \{ w \in \{0, 1\}^* | w \text{ contains an equal number of 0s and 1s. } \} \]
**BALANCE** and the Weak Pumping Lemma

\[ BALANCE = \{ w \in \{0, 1\}^* \mid w \text{ contains an equal number of 0s and 1s.} \} \]
BALANCE and the Weak Pumping Lemma

\[ \text{BALANCE} = \{ w \in \{0, 1\}^* | \text{w contains an equal number of 0s and 1s.} \} \]
BALANCE and the Weak Pumping Lemma

\[ BALANCE = \{ w \in \{0, 1\}^* | \text{w contains an equal number of 0s and 1s.} \} \]
**BALANCE and the Weak Pumping Lemma**

\[ BALANCE = \{ w \in \{0, 1\}^* \mid w \text{ contains an equal number of } 0\text{s and } 1\text{s.} \} \]
**BALANCE** and the Weak Pumping Lemma

\[ BALANCE = \{ w \in \{0, 1\}^* | w \text{ contains an equal number of 0s and 1s.} \} \]
\( BALANCE \) and the Weak Pumping Lemma

\[ BALANCE = \{ \ w \in \{0, 1\}^* \mid \text{w contains an equal number of 0s and 1s} \. \} \]
BALANCE and the Weak Pumping Lemma

$BALANCE = \{ w \in \{0, 1\}^* | w \text{ contains an equal number of 0s and 1s.} \}$

1 0 0 1
BALANCE and the Weak Pumping Lemma

\[ \text{BALANCE} = \{ w \in \{0, 1\}^* | \text{w contains an equal number of 0s and 1s.} \} \]
BALANCE and the Weak Pumping Lemma

\[ BALANCE = \{ w \in \{0, 1\}^* | w \text{ contains an equal number of } 0\text{s and } 1\text{s.} \} \]
BALANCE and the Weak Pumping Lemma

\[ \text{BALANCE} = \{ w \in \{0, 1\}^* \mid \text{w contains an equal number of 0s and 1s.} \} \]

1 0 0 1 1 0 0 1
BALANCE and the Weak Pumping Lemma

\[ \text{BALANCE} = \{ w \in \{0, 1\}^* | w \text{ contains an equal number of } 0\text{s and } 1\text{s.} \} \]
Theorem: BALANCE is regular.

Proof: We show that BALANCE satisfies the condition of the pumping lemma. Let $n = 2$ and consider any string $w \in BALANCE$ such that $|w| \geq 2$. Then we can write $w = xyz$ such that $x = z = \varepsilon$ and $y = w$, so $y \neq \varepsilon$. Then for any natural number $i$, $xy^iz = w^i$, which has the same number of 0s and 1s. Since BALANCE passes the conditions of the weak pumping lemma, BALANCE is regular. ■
An Incorrect Proof

**Theorem:** \( \text{BALANCE} \) is regular.

**Proof:** We show that \( \text{BALANCE} \) satisfies the condition of the pumping lemma. Let \( n = 2 \) and consider any string \( w \in \text{BALANCE} \) such that \( |w| \geq 2 \). Then we can write \( w = xyz \) such that \( x = z = \varepsilon \) and \( y \neq \varepsilon \). Then for any natural number \( i \), \( xy^iz = w^i \), which has the same number of 0s and 1s. Since \( \text{BALANCE} \) passes the pumping lemma, \( \text{BALANCE} \) is regular. ■
The Weak Pumping Lemma

- The **Weak Pumping Lemma for Regular Languages** states that

  **For any** regular language \( L \),

  **There exists** a positive natural number \( n \) such that

  **For any** \( w \in L \) with \(|w| \geq n\),

  **There exists** strings \( x, y, z \) such that

  **For any** natural number \( i \),

  \[ w = xyz, \ w \text{ can be broken into three pieces}, \]

  \[ y \neq \varepsilon \text{ where the middle piece isn't empty}, \]

  \[ xy^iz \in L \text{ where the middle piece can be replicated zero or more times}. \]
The Weak Pumping Lemma

- The **Weak Pumping Lemma for Regular Languages** states that
  
  **For any** regular language \( L \),

  **There exists** a positive natural number \( n \) such that

  **For any** \( w \in L \) with \( |w| \geq n \),

  **There exists** strings \( x, y, z \) such that

  **For any** natural number \( i \),

  \[
  w = xyz, \ \text{where the middle piece isn't empty,}
  \]

  \[
  y \neq \varepsilon, \ \text{where the middle piece can be replicated zero or more times.}
  \]

  \[
  xy^i z \in L
  \]
Caution with the Pumping Lemma

- The weak and full pumping lemmas describe a **necessary** condition of regular languages.
  - If $L$ is regular, $L$ passes the conditions of the pumping lemma.
- The weak and full pumping lemmas are not a **sufficient** condition of regular languages.
  - If $L$ is *not* regular, it still might pass the conditions of the pumping lemma!
- If a language fails the pumping lemma, it is definitely not regular.
- If a language passes the pumping lemma, we learn nothing about whether it is regular or not.
**BALANCE** is Not Regular

- The language **BALANCE** can be proven not to be regular using a stronger version of the pumping lemma.
- To see the full pumping lemma, we need to revisit our original insight.
An Important Observation
An Important Observation
An Important Observation
An Important Observation
Weak Pumping Lemma Intuition

• Let $D$ be a DFA with $n$ states.

• Any string $w$ accepted by $D$ that has length at least $n$ must visit some state twice.

  • Number of states visited is equal to $|w| + 1$.
  • By the pigeonhole principle, some state is duplicated.

• The substring of $w$ in-between those revisited states can be removed, duplicated, tripled, quadrupled, etc. without changing the fact that $w$ is accepted by $D$. 
Pumping Lemma Intuition

- Let $D$ be a DFA with $n$ states.
- Any string $w$ accepted by $D$ that has length at least $n$ must visit some state twice **within its first $n$ characters**.
  - Number of states visited is equal $n + 1$.
  - By the pigeonhole principle, some state is duplicated.
- The substring of $w$ in-between those revisited states can be removed, duplicated, tripled, quadrupled, etc. without changing the fact that $w$ is accepted by $D$. 

The Weak Pumping Lemma

For any regular language $L$,

There exists a positive natural number $n$ such that

For any $w \in L$ with $|w| \geq n$,

There exists strings $x, y, z$ such that

For any natural number $i$,

$$w = xyz$$

where $w$ can be broken into three pieces,

$y \neq \varepsilon$ where the middle piece isn't empty,

$xy^iz \in L$ where the middle piece can be replicated zero or more times.
The Pumping Lemma

For any regular language $L$,

There exists a positive natural number $n$ such that

For any $w \in L$ with $|w| \geq n$,

There exists strings $x, y, z$ such that

For any natural number $i$,

$$w = xyz,$$  

$w$ can be broken into three pieces,

$$|xy| \leq n,$$  

where the first two pieces occur at the start of the string,

$y \neq \varepsilon$  

where the middle piece isn’t empty,

$xy^iz \in L$  

where the middle piece can be replicated zero or more times.
Why This Change Matters

- The restriction $|xy| \leq n$ means that we can limit where the string to pump must be.
- If we specifically craft the first $n$ characters of the string to pump, we can force $y$ to have a specific property.
- We can then show that $y$ cannot be pumped arbitrarily many times.
**BALANCE** and the Pumping Lemma

**BALANCE** = \{ w \in \{0, 1\}^* \mid w \text{ contains an equal number of } 0 \text{ s and } 1 \text{ s.} \}
BALANCE and the Pumping Lemma

\[ \text{BALANCE} = \{ w \in \{0, 1\}^* \mid w \text{ contains an equal number of 0s and 1s.} \} \]

Suppose the pumping length is 4.
**BALANCE** and the Pumping Lemma

**BALANCE** = \{ w \in \{0, 1\}^* | w \text{ contains an equal number of } 0\text{s and } 1\text{s.} \}

Suppose the pumping length is 4.

\[
\begin{array}{ccccccccc}
0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\
\end{array}
\]
**BALANCE and the Pumping Lemma**

\[ \text{BALANCE} = \{ w \in \{0, 1\}^* | \text{w contains an equal number of 0s and 1s.} \} \]

Suppose the pumping length is 4.

\[
0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1
\]

Since \(|xy| \leq 4\), the string to pump must be somewhere in here.
**BALANCE and the Pumping Lemma**

\[ BALANCE = \{ w \in \{0, 1\}^* \mid w \text{ contains an equal number of } 0\text{s and } 1\text{s.} \} \]

Suppose the pumping length is 4.
\[ \textit{BALANCE} = \{ w \in \{0, 1\}^* \mid w \text{ contains an equal number of 0s and 1s.} \} \]

Suppose the pumping length is 4.
\textit{BALANCE} and the Pumping Lemma

\[ \text{BALANCE} = \{ w \in \{0, 1\}^* | \text{w contains an equal number of 0s and 1s.} \} \]

Suppose the pumping length is 4.

\[ \begin{array}{c c c c c c c c}
0 & 1 & 1 & 1 & 1 & 1 \\
\end{array} \]
**BALANCE and the Pumping Lemma**

\[ BALANCE = \{ w \in \{0, 1\}^* | w \text{ contains an equal number of 0s and 1s.} \} \]

Suppose the pumping length is 4.

```
0 0 0 0 1 1 1 1 1
```
BALANCE and the Pumping Lemma

BALANCE = \{ w \in \{0, 1\}^* \mid w \text{ contains an equal number of 0s and 1s.} \}

Suppose the pumping length is 4.
**BALANCE** and the Pumping Lemma

\[ \text{BALANCE} = \{ w \in \{0, 1\}^* | \text{\textit{w} contains an equal number of 0s and 1s.} \} \]

Suppose the pumping length is 4.

\[ \begin{array}{cccc}
1 & 1 & 1 & 1 \\
\end{array} \]
\textit{BALANCE} and the Pumping Lemma

\[ \text{BALANCE} = \{ w \in \{0, 1\}^* \mid w \text{ contains an equal number of 0s and 1s.} \} \]

Suppose the pumping length is 4.
**BALANCE and the Pumping Lemma**

\[ \text{BALANCE} = \{ w \in \{0, 1\}^* | \text{w contains an equal number of 0s and 1s} \} \]

Suppose the pumping length is 4.

\[
\begin{array}{cccccccc}
0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\
\end{array}
\]
\[ BALANCE = \{ w \in \{0, 1\}^* | \text{w contains an equal number of 0s and 1s.} \} \]

Suppose the pumping length is 4.
Theorem: BALANCE is not regular.
Theorem: BALANCE is not regular.

Proof: By contradiction; assume that BALANCE is regular.

Let $n$ be the length guaranteed by the pumping lemma. Consider the string $w = 0^n 1^n$. Then $|w| = 2n \geq n$ and $w \in \text{BALANCE}$. Therefore, there exist strings $x$, $y$, and $z$ such that $w = xyz$, $|xy| \leq n$, $y \neq \varepsilon$, and for any natural number $i$, $xy^iz \in \text{BALANCE}$. Since $|xy| \leq n$, $y$ must consist solely of 0s. But then $xy^2z = 0^{n+|y|} 1^n$, and since $|y| > 0$, $xy^2z \notin \text{BALANCE}$.

We have reached a contradiction, so our assumption was wrong and BALANCE is not regular. ■
Theorem: BALANCE is not regular.

Proof: By contradiction; assume that BALANCE is regular. Let $n$ be the length guaranteed by the pumping lemma.

Let $w = 0^n 1^n$. Then $|w| = 2n \geq n$ and $w \in \text{BALANCE}$. Therefore, there exist strings $x$, $y$, and $z$ such that $w = xyz$, $|xy| \leq n$, $y \neq \varepsilon$, and for any natural number $i$, $xy^iz \in \text{BALANCE}$. Since $|xy| \leq n$, $y$ must consist solely of 0s. But then $xy^2z = 0^{n+|y|} 1^n$, and since $|y| > 0$, $xy^2z \notin \text{BALANCE}$.

We have reached a contradiction, so our assumption was wrong and BALANCE is not regular. ■
**Theorem:** BALANCE is not regular.

**Proof:** By contradiction; assume that BALANCE is regular. Let $n$ be the length guaranteed by the pumping lemma. Consider the string $w = 0^n1^n$. Since $|w| = 2n \geq n$ and $w \in \text{BALANCE}$, there exist strings $x$, $y$, and $z$ such that $w = xyz$, $|xy| \leq n$, $y \neq \epsilon$, and for any natural number $i$, $xy^iz \in \text{BALANCE}$. Since $|xy| \leq n$, $y$ must consist solely of 0s. But then $xy^2z = 0^{n+|y|}1^n$, and since $|y| > 0$, $xy^2z \notin \text{BALANCE}$. We have reached a contradiction, so our assumption was wrong and BALANCE is not regular. □
Theorem: BALANCE is not regular.

Proof: By contradiction; assume that BALANCE is regular. Let $n$ be the length guaranteed by the pumping lemma. Consider the string $w = 0^n1^n$. Then $|w| = 2n \geq n$ and $w \in BALANCE$. 

We have reached a contradiction, so our assumption was wrong and BALANCE is not regular. ■
Theorem: BALANCE is not regular.

Proof: By contradiction; assume that BALANCE is regular. Let \( n \) be the length guaranteed by the pumping lemma. Consider the string \( w = 0^n1^n \). Then \( |w| = 2n \geq n \) and \( w \in BALANCE \). Therefore, there exist strings \( x, y, \) and \( z \) such that \( w = xyz \), \( |xy| \leq n \), \( y \neq \varepsilon \), and for any natural number \( i \), \( xy^iz \in BALANCE \).
**Theorem:** BALANCE is not regular.

**Proof:** By contradiction; assume that BALANCE is regular. Let $n$ be the length guaranteed by the pumping lemma. Consider the string $w = 0^n1^n$. Then $|w| = 2n \geq n$ and $w \in BALANCE$. Therefore, there exist strings $x$, $y$, and $z$ such that $w = xyz$, $|xy| \leq n$, $y \neq \varepsilon$, and for any natural number $i$, $xy^iz \in BALANCE$. Since $|xy| \leq n$, $y$ must consist solely of 0s.
**Theorem:** BALANCE is not regular.

**Proof:** By contradiction; assume that BALANCE is regular. Let \( n \) be the length guaranteed by the pumping lemma. Consider the string \( w = 0^n1^n \). Then \( |w| = 2n \geq n \) and \( w \in BALANCE \). Therefore, there exist strings \( x, y, \) and \( z \) such that \( w = xyz \), \( |xy| \leq n \), \( y \neq \varepsilon \), and for any natural number \( i \), \( xy^iz \in BALANCE \). Since \( |xy| \leq n \), \( y \) must consist solely of 0s.

This is why the pumping lemma is more powerful than the weak pumping lemma. We can force \( y \) to be made purely of 0s, rather than some combination of 0s and 1s.
Theorem: BALANCE is not regular.

Proof: By contradiction; assume that BALANCE is regular. Let \( n \) be the length guaranteed by the pumping lemma. Consider the string \( w = 0^n1^n \). Then \( |w| = 2n \geq n \) and \( w \in \text{BALANCE} \). Therefore, there exist strings \( x, y, \) and \( z \) such that \( w = xyz \), \( |xy| \leq n \), \( y \neq \varepsilon \), and for any natural number \( i \), \( xy^iz \in \text{BALANCE} \). Since \( |xy| \leq n \), \( y \) must consist solely of 0s. But then \( xy^2z = 0^{n+|y|}1^n \), and since \( |y| > 0 \), \( xy^2z \notin \text{BALANCE} \).
**Theorem:** BALANCE is not regular.

**Proof:** By contradiction; assume that BALANCE is regular. Let $n$ be the length guaranteed by the pumping lemma. Consider the string $w = 0^n1^n$. Then $|w| = 2n \geq n$ and $w \in BALANCE$. Therefore, there exist strings $x$, $y$, and $z$ such that $w = xyz$, $|xy| \leq n$, $y \neq \varepsilon$, and for any natural number $i$, $xy^iz \in BALANCE$. Since $|xy| \leq n$, $y$ must consist solely of 0s. But then $xy^2z = 0^{n+|y|}1^n$, and since $|y| > 0$, $xy^2z \notin BALANCE$.

We have reached a contradiction, so our assumption was wrong and BALANCE is not regular.
**Theorem:** BALANCE is not regular.

**Proof:** By contradiction; assume that BALANCE is regular. Let $n$ be the length guaranteed by the pumping lemma. Consider the string $w = 0^n1^n$. Then $|w| = 2n \geq n$ and $w \in BALANCE$. Therefore, there exist strings $x$, $y$, and $z$ such that $w = xyz$, $|xy| \leq n$, $y \neq \varepsilon$, and for any natural number $i$, $xy^iz \in BALANCE$. Since $|xy| \leq n$, $y$ must consist solely of $0$s. But then $xy^2z = 0^{n+|y|}1^n$, and since $|y| > 0$, $xy^2z \notin BALANCE$.

We have reached a contradiction, so our assumption was wrong and BALANCE is not regular. $\blacksquare$
Summary of the Pumping Lemma

• Using the pigeonhole principle, we can prove the weak pumping lemma and pumping lemma.

• These lemmas describe essential properties of the regular languages.

• Any language that fails to have these properties cannot be regular.
Next Time

• **Beyond Regular Languages**
  • Context-free languages.
  • Context-free grammars.
Appendix: From Regular Expressions to NFAs
A Marvelous Construction

- To show that any language described by a regular expression is regular, we show how to convert a regular expression into an NFA.
- **Theorem:** For any regular expression $R$, there is an NFA $N$ such that
  - $\mathcal{L}(R) = \mathcal{L}(N)$
  - $N$ has exactly one accepting state.
  - $N$ has no transitions into its start state.
  - $N$ has no transitions out of its accepting state.
To show that any language described by a regular expression is regular, we show how to convert a regular expression into an NFA.

Theorem: For any regular expression $R$, there is an NFA $N$ such that $\mathcal{L}(R) = \mathcal{L}(N)$

- $N$ has exactly one accepting state.
- $N$ has no transitions into its start state.
- $N$ has no transitions out of its accepting state.
To show that any language described by a regular expression is regular, we show how to convert a regular expression into an NFA.

**Theorem:** For any regular expression \( R \), there is an NFA \( N \) such that \( \mathcal{L}(R) = \mathcal{L}(N) \).

- \( N \) has exactly one accepting state.
- \( N \) has no transitions into its start state.
- \( N \) has no transitions out of its accepting state.

These are stronger requirements than are necessary for a normal NFA. We enforce these rules to simplify the construction.
Base Cases

Automaton for $\varepsilon$

Automaton for $\emptyset$

Automaton for single character $a$
Construction for $R_1 R_2$
Construction for $R_1R_2$
Construction for $R_1 R_2$
Construction for $R_1R_2$
Construction for $R_1 R_2$

![Diagram of construction process involving $R_1$ and $R_2$ with an arrow labeled $\varepsilon$.]
Construction for $R_1 \mid R_2$
Construction for $R_1 \mid R_2$
Construction for $R_1 \mid R_2$
Construction for $R_1 \mid R_2$

```
\begin{figure}
\centering
\begin{tikzpicture}
\node (start) [circle, draw] at (0,0) {start};
\node (r1) [circle, fill=lightgray] at (2,0) {$R_1$};
\node (r2) [circle, fill=lightgray] at (2,-2) {$R_2$};
\draw[->, dashed] (start) -- (r1);
\draw[->, dashed] (r1) -- (r2);
\draw[->, dashed] (r2) -- (r1);
\draw[->, dashed] (r1) -- (r2);
\end{tikzpicture}
\end{figure}
```
Construction for $R_1 \mid R_2$
Construction for $R_1 \mid R_2$
Construction for $R_1 \mid R_2$
Construction for R*
Construction for $R^*$
Construction for R*
Construction for $R^*$
Construction for $R^*$
Construction for $R^*$
Construction for R*
Appendix: From NFAs to Regular Expressions
From NFAs to Regular Expressions
From NFAs to Regular Expressions

![NFA Diagram]
From NFAs to Regular Expressions

$q_s$

$q_0$ -> $q_1$: 0

$q_0$ -> $q_2$: 1

$q_2$ -> $q_f$: 0, 1

$q_1$ -> $q_f$: 0 | 1

Start

$q_f$
From NFAs to Regular Expressions
From NFAs to Regular Expressions
From NFAs to Regular Expressions

\[
\begin{array}{c}
\text{start} \\
\downarrow \\
s_0 \\
\downarrow \\
0 | 1 \\
\downarrow \\
q_f \\
\end{array}
\]
From NFAs to Regular Expressions
From NFAs to Regular Expressions
From NFAs to Regular Expressions
From NFAs to Regular Expressions

![Diagram of an NFA](image)

- **Start state**: $q_s$
- **States**: $q_0$, $q_1$, $q_f$
- **Transitions**:
  - $q_s$ on $\varepsilon$ to $q_0$
  - $q_0$ on $0$, $1$ to $q_1$
  - $q_1$ on $\varepsilon$ to $q_f$

**Regular Expression** derived from the NFA:

$$(\varepsilon + 0 + 1)^*$$
From NFAs to Regular Expressions
From NFAs to Regular Expressions

The diagram shows a non-deterministic finite automaton (NFA) with states $q_0$, $q_1$, and $q_f$. The transitions include:

- From $q_s$ (start state) to $q_0$ on $\varepsilon$.
- From $q_0$ to $q_1$ on $0 \mid 1$.
- From $q_1$ to $q_0$ and $q_f$ on $\varepsilon$.
- From $q_0$ to $q_f$ on $\varepsilon$.
From NFAs to Regular Expressions
From NFAs to Regular Expressions
From NFAs to Regular Expressions
From NFAs to Regular Expressions
From NFAs to Regular Expressions

- Start state: $q_s$
- States: $q_0$, $q_1$, $q_f$
- Transitions:
  - $q_0$ to $q_s$: $\varepsilon$
  - $q_0$ to $q_1$: $0(0 \mid 1)$
  - $q_0$ to $q_f$: $0\varepsilon$
  - $q_1$ to $q_f$: $\varepsilon$
  - $q_1$ to $q_1$: $\varepsilon$

Regular Expression: $0(0 \mid 1)$
From NFAs to Regular Expressions

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

\[ q_0 \xrightarrow{0(0 | 1)} q_0 \]

\[ q_0 \xrightarrow{0} q_1 \]

\[ q_1 \xrightarrow{\varepsilon} q_f \]

\[ q_1 \xrightarrow{|1|} q_1 \]

\[ q_f \xrightarrow{\varepsilon} q_f \]
From NFAs to Regular Expressions

```
0(0 | 1)
```

The diagram shows a nondeterministic finite automaton (NFA) with states $q_s$, $q_0$, $q_f$, and transitions labeled with $0$, $0(0 | 1)$, $\varepsilon$, and $\varepsilon$. The start state is $q_s$, and the accepting state is $q_f$. The regular expression is $0(0 | 1)$.
From NFAs to Regular Expressions

\[
\begin{align*}
q_s & \xrightarrow{\varepsilon} q_f \\
q_0 & \xrightarrow{0(0 | 1)} q_0 \\
q_0 & \xrightarrow{\varepsilon} q_f \\
q_0 & \xrightarrow{0} q_f
\end{align*}
\]
From NFAs to Regular Expressions

- Start state $q_s$
- Transition on $\varepsilon$ from $q_s$ to $q_0$
- Transition on $0(0 | 1)$ from $q_0$ to $q_0$
- Transition on $0 | \varepsilon$ from $q_0$ to $q_f$
From NFAs to Regular Expressions

\[ q_s \]

\[ q_0 \] (0 | 1)

\[ q_f \]

\[ ε \]
From NFAs to Regular Expressions

\[
\begin{align*}
q_0 &\rightarrow 0(0 \mid 1) \\
q_f &\rightarrow 0 \mid \varepsilon
\end{align*}
\]
From NFAs to Regular Expressions

\[
\begin{align*}
q_s & \quad \text{start} \\
\epsilon & \\
q_0 & \quad 0(0 \mid 1) \\
0 \mid \epsilon & \\
q_f & 
\end{align*}
\]
From NFAs to Regular Expressions

\[ q_0 \xrightarrow{0(0 \mid 1)} q_0 \]

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

\[ q_s \xrightarrow{\text{start}} q_s \]

\[ q_f \]

\[ q_f \xrightarrow{0 \mid \varepsilon} q_f \]
From NFAs to Regular Expressions

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

\[ q_s \xrightarrow{\varepsilon} q_f \]

\[ q_s \xrightarrow{\varepsilon(0(0 | 1))*(0 | \varepsilon)} q_f \]

\[ q_0 \xrightarrow{0(0 | 1)} q_0 \]

\[ q_0 \xrightarrow{0 | \varepsilon} q_f \]

\[ 0 | \varepsilon \]
From NFAs to Regular Expressions

\[ q_s \] start

\[ \varepsilon \]

\[ q_0 \]

\[ 0(0 \mid 1) \]

\[ \varepsilon(0(0 \mid 1)) \ast (0 \mid \varepsilon) \]

\[ q_f \]
From NFAs to Regular Expressions

\[ \varepsilon(0(0 \mid 1))^*(0 \mid \varepsilon) \]
From NFAs to Regular Expressions

\[ (0(0 \mid 1))^* (0 \mid \varepsilon) \]
From NFAs to Regular Expressions

\[(0(0 \mid 1))^{*}(0 \mid \varepsilon)\]

Diagram:
- Start state: \(q_0\)
- States: \(q_0, q_1, q_2\)
- Transitions:
  - From \(q_0\) on 0 to \(q_1\)
  - From \(q_0\) on 1 to \(q_2\)
  - From \(q_1\) on 0 to \(q_1\)
  - From \(q_1\) on 1 to \(q_2\)
  - From \(q_2\) on 0, 1 to \(q_2\)
  - From \(q_2\) on 0, 1 to \(q_0\)
Appendix: Homomorphisms of Regular Languages
Homomorphisms of Regular Languages

• Consider the language defined by the regular expression \((0120)^*\) and the function
  
  • \(h(0) = n\)
  • \(h(1) = y\)
  • \(h(2) = a\)

• Then \(h^*((0120)^*) = ((n) (y) (a) (n))^*\)
Homomorphisms of Regular Languages

- Consider the language $011^*$ and the function
  - $h(0) = \text{Here}$
  - $h(1) = \text{Kitty}$
- Then $h^*(011^*) = (\text{Here})(\text{Kitty})(\text{Kitty})^*$