Pre-lecture brain teaser

\[ L' = \{ \text{bitstrings with equal number of 0s and 1s} \} \]

\[ L = \{0^n1^n \mid n \geq 0\} \]

Suppose we have already shown that \( L' \) is non-regular. Can we show \( L \) is regular via closure.
CS/ECE-374: Lecture 7 - Non-regularity and fooling sets

Lecturer: Nickvash Kani
Chat moderator: Samir Khan
February 16, 2021

University of Illinois at Urbana-Champaign
Non-regularity via closure properties

$L’ = \{\text{bitstrings with equal number of 0s and 1s}\}$

$L = \{0^n1^n \mid n \geq 0\}$

Suppose we have already shown that $L’$ is non-regular. Can we show $L$ is regular via closure.
Non-regularity via closure properties

$L' = \{\text{bitstrings with equal number of 0s and 1s}\}$

$L = \{0^n1^n \mid n \geq 0\}$

Suppose we have already shown that $L'$ is non-regular. Can we show $L$ is regular via closure.

Can we show that $L$ is non-regular from scratch?
Proving non-regularity: Methods

- **Pumping lemma.** We will not cover it but it is *sometimes* an easier proof technique to apply, but not as general as the fooling set technique.
- **Closure properties.** Use existing non-regular languages and regular languages to prove that some new language is non-regular.
- **Fooling sets**—Method of distinguishing suffixes. To prove that $L$ is non-regular find an infinite fooling set.
We have a language $L = \{0^n1^n | n \geq 0\}$
Prove that $L$ is non-regular.
Not all languages are regular
Theorem
Languages accepted by DFAs, NFAs, and regular expressions are the same.

Question: Is every language a regular language? No.
Theorem
Languages accepted by DFAs, NFAs, and regular expressions are the same.

Question: Is every language a regular language? No.

- Each DFA $M$ can be represented as a string over a finite alphabet $\Sigma$ by appropriate encoding.
- Hence number of regular languages is countably infinite.
- Number of languages is uncountably infinite.
- Hence there must be a non-regular language!
A Simple and Canonical Non-regular Language

\[ L = \{0^n1^n \mid n \geq 0\} = \{\epsilon, 01, 0011, 000111, \ldots, \} \]
A Simple and Canonical Non-regular Language

\[ L = \{0^n1^n \mid n \geq 0\} = \{\epsilon, 01, 0011, 000111, \cdots\} \]

**Theorem**

\( L \) is not regular.
A Simple and Canonical Non-regular Language

\[ L = \{0^n1^n \mid n \geq 0\} = \{\epsilon, 01, 0011, 000111, \cdots\} \]

**Theorem**

\( L \) is not regular.

**Question:** Proof?
A Simple and Canonical Non-regular Language

\[ L = \{0^n1^n \mid n \geq 0\} = \{\epsilon, 01, 0011, 000111, \cdots\} \]

**Theorem**

\( L \) is not regular.

**Question:** Proof?

**Intuition:** Any program to recognize \( L \) seems to require counting number of zeros in input which cannot be done with fixed memory.
A Simple and Canonical Non-regular Language

$L = \{0^n1^n \mid n \geq 0\} = \{\epsilon, 01, 0011, 000111, \cdots\}$

**Theorem**

$L$ is not regular.

**Question:** Proof?

**Intuition:** Any program to recognize $L$ seems to require counting number of zeros in input which cannot be done with fixed memory.

How do we formalize intuition and come up with a formal proof?
Proof by contradiction

• Suppose \( L \) is regular. Then there is a DFA \( M \) such that \( L(M) = L \).
• Let \( M = (Q, \{0, 1\}, \delta, s, A) \) where \(|Q| = n\).
Proof by contradiction

• Suppose $L$ is regular. Then there is a DFA $M$ such that $L(M) = L$.

• Let $M = (Q, \{0, 1\}, \delta, s, A)$ where $|Q| = n$. 
Proof by Contradiction

- Suppose $L$ is regular. Then there is a DFA $M$ such that $L(M) = L$.
- Let $M = (Q, \{0, 1\}, \delta, s, A)$ where $|Q| = n$. 
Proof by Contradiction

- Suppose \( L \) is regular. Then there is a DFA \( M \) such that \( L(M) = L \).
- Let \( M = (Q, \{0, 1\}, \delta, s, A) \) where \(|Q| = n\).

Consider strings \( \epsilon, 0, 00, 000, \cdots, 0^n \) total of \( n + 1 \) strings.
Proof by Contradiction

- Suppose \( L \) is regular. Then there is a DFA \( M \) such that \( L(M) = L \).
- Let \( M = (Q, \{0, 1\}, \delta, s, A) \) where \(|Q| = n\).

Consider strings \( \epsilon, 0, 00, 000, \ldots, 0^n \) total of \( n + 1 \) strings.

What states does \( M \) reach on the above strings? Let \( q_i = \delta^*(s, 0^i) \).

By pigeon hole principle \( q_i = q_j \) for some \( 0 \leq i < j \leq n \).
That is, \( M \) is in the same state after reading \( 0^i \) and \( 0^j \) where \( i \neq j \).
• Suppose \( L \) is regular. Then there is a **DFA** \( M \) such that \( L(M) = L \).

• Let \( M = (Q, \{0, 1\}, \delta, s, A) \) where \( |Q| = n \).

Consider strings \( \epsilon, 0, 00, 000, \cdots, 0^n \) total of \( n + 1 \) strings.

What states does \( M \) reach on the above strings? Let \( q_i = \delta^*(s, 0^i) \).

By pigeon hole principle \( q_i = q_j \) for some \( 0 \leq i < j \leq n \).

That is, \( M \) is in the same state after reading \( 0^i \) and \( 0^j \) where \( i \neq j \).

\( M \) should accept \( 0^i1^i \) but then it will also accept \( 0^j1^i \) where \( i \neq j \).
Proof by Contradiction

- Suppose $L$ is regular. Then there is a DFA $M$ such that $L(M) = L$.
- Let $M = (Q, \{0, 1\}, \delta, s, A)$ where $|Q| = n$.

Consider strings $\epsilon, 0, 00, 000, \ldots, 0^n$ total of $n + 1$ strings.

What states does $M$ reach on the above strings? Let $q_i = \delta^*(s, 0^i)$.

By pigeon hole principle $q_i = q_j$ for some $0 \leq i < j \leq n$. That is, $M$ is in the same state after reading $0^i$ and $0^j$ where $i \neq j$.

$M$ should accept $0^i1^i$ but then it will also accept $0^j1^i$ where $i \neq j$. This contradicts the fact that $M$ accepts $L$. Thus, there is no DFA for $L$. 
When two states are equivalent?
States that cannot be combined?

We concluded that because each $0^i$ prefix has a unique state. Are there states that aren’t unique? Can states be combined?
Equivalence between states

Definition
\( M = (Q, \Sigma, \delta, s, A): \text{DFA}. \)

Two states \( p, q \in Q \) are equivalent if for all strings \( w \in \Sigma^* \), we have that

\[
\delta^*(p, w) \in A \iff \delta^*(q, w) \in A.
\]

One can merge any two states that are equivalent into a single state.
Distinguishing between states

**Definition**

\( M = (Q, \Sigma, \delta, s, A) \): DFA.

Two states \( p, q \in Q \) are **distinguishable** if there exists a string \( w \in \Sigma^* \), such that

\[ \delta^*(p, w) \in A \quad \text{and} \quad \delta^*(q, w) \notin A. \]

or

\[ \delta^*(p, w) \notin A \quad \text{and} \quad \delta^*(q, w) \in A. \]
Distinguishable prefixes

\[ M = (Q, \Sigma, \delta, s, A): \text{DFA} \]

**Idea:** Every string \( w \in \Sigma^* \) defines a state \( \nabla w = \delta^*(s, w) \).
Distinguishable prefixes

\( M = (Q, \Sigma, \delta, s, A) \): DFA

**Idea:** Every string \( w \in \Sigma^* \) defines a state \( \nabla w = \delta^*(s, w) \).

**Definition**
Two strings \( u, w \in \Sigma^* \) are **distinguishable** for \( M \) (or \( L(M) \)) if \( \nabla u \) and \( \nabla w \) are distinguishable.
Distinguishable prefixes

$M = (Q, \Sigma, \delta, s, A)$: DFA

Idea: Every string $w \in \Sigma^*$ defines a state $\nabla w = \delta^*(s, w)$.

Definition
Two strings $u, w \in \Sigma^*$ are distinguishable for $M$ (or $L(M)$) if $\nabla u$ and $\nabla w$ are distinguishable.

Definition (Direct restatement)
Two prefixes $u, w \in \Sigma^*$ are distinguishable for a language $L$ if there exists a string $x$, such that $ux \in L$ and $wx \notin L$ (or $ux \notin L$ and $wx \in L$).
Lemma

\( L: \text{regular language.} \)

\( M = (Q, \Sigma, \delta, s, A): \text{DFA for } L. \)

If \( x, y \in \Sigma^* \) are distinguishable, then \( \nabla x \neq \nabla y. \)

Reminder: \( \nabla x = \delta^*(s, x) \in Q \) and \( \nabla y = \delta^*(s, y) \in Q \)
Proof by a figure

Possible

Not possible

\[ \delta^*(s, x) \xrightarrow{w} \delta^*(s, xw) \]

\[ \delta^*(s, y) \xrightarrow{w} \delta^*(s, yw) \]

\[ \delta^*(s, x) = \delta^*(s, y) \]

\[ \delta^*(s, xw) \]

\[ \delta^*(s, yw) \]
Distinguishable strings means different states: Proof

**Lemma**

$L$: regular language.

$M = (Q, \Sigma, \delta, s, A)$: DFA for $L$.

If $x, y \in \Sigma^*$ are distinguishable, then $\nabla x \neq \nabla y$.

**Proof.**

Assume for the sake of contradiction that $\nabla x = \nabla y$.
Distinguishable strings means different states: Proof

**Lemma**

$L$: regular language.

$M = (Q, \Sigma, \delta, s, A)$: DFA for $L$.

If $x, y \in \Sigma^*$ are distinguishable, then $\nabla x \neq \nabla y$.

**Proof.**

Assume for the sake of contradiction that $\nabla x = \nabla y$.

By assumption $\exists w \in \Sigma^*$ such that $\nabla xw \in A$ and $\nabla yw \notin A$. 

15
Lemma
$L$: regular language.

\[ M = (Q, \Sigma, \delta, s, A): \text{DFA for } L. \]

If \( x, y \in \Sigma^* \) are distinguishable, then \( \nabla x \neq \nabla y. \)

Proof.
Assume for the sake of contradiction that \( \nabla x = \nabla y. \)

By assumption \( \exists w \in \Sigma^* \) such that \( \nabla xw \in A \) and \( \nabla yw \notin A. \)

\[ \implies A \ni \nabla xw = \delta^*(s, xw) = \delta^*(\nabla x, w) \]
Lemma

$L$: regular language.

$M = (Q, \Sigma, \delta, s, A)$: DFA for $L$.

If $x, y \in \Sigma^*$ are distinguishable, then $\nabla x \neq \nabla y$.

Proof.

Assume for the sake of contradiction that $\nabla x = \nabla y$.

By assumption $\exists w \in \Sigma^*$ such that $\nabla x w \in A$ and $\nabla y w \not\in A$.

$\implies A \ni \nabla x w = \delta^*(s, x w) = \delta^*(\nabla x, w) = \delta^*(\nabla y, w)$
Distinguishable strings means different states: Proof

**Lemma**

$L$: regular language.

$M = (Q, \Sigma, \delta, s, A)$: DFA for $L$.

If $x, y \in \Sigma^*$ are distinguishable, then $\nabla x \neq \nabla y$.

**Proof.**

Assume for the sake of contradiction that $\nabla x = \nabla y$.

By assumption $\exists w \in \Sigma^*$ such that $\nabla xw \in A$ and $\nabla yw \notin A$.

$$\Rightarrow A \ni \nabla xw = \delta^*(s, xw) = \delta^*(\nabla x, w) = \delta^*(\nabla y, w) = \delta^*(s, yw) = \nabla yw \notin A.$$
Lemma
$L$: regular language.

$M = (Q, \Sigma, \delta, s, A)$: DFA for $L$.

If $x, y \in \Sigma^*$ are distinguishable, then $\nabla x \neq \nabla y$.

Proof.
Assume for the sake of contradiction that $\nabla x = \nabla y$.

By assumption $\exists w \in \Sigma^*$ such that $\nabla xw \in A$ and $\nabla yw \notin A$.

$$\Rightarrow A \ni \nabla xw = \delta^*(s, xw) = \delta^*(\nabla x, w) = \delta^*(\nabla y, w) = \delta^*(s, yw) = \nabla yw \notin A.$$

$$\Rightarrow A \ni \nabla yw \notin A. \text{ Impossible!}$$
Lemma

$L$: regular language.

$M = (Q, \Sigma, \delta, s, A)$: DFA for $L$.

If $x, y \in \Sigma^*$ are distinguishable, then $\nabla x \neq \nabla y$.

Proof.

Assume for the sake of contradiction that $\nabla x = \nabla y$.

By assumption $\exists w \in \Sigma^*$ such that $\nabla xw \in A$ and $\nabla yw \notin A$.

$\implies A \ni \nabla xw = \delta^*(s, xw) = \delta^*(\nabla x, w) = \delta^*(\nabla y, w) = \delta^*(s, yw) = \nabla yw \notin A.$

$\implies A \ni \nabla yw \notin A$. Impossible!

Assumption that $\nabla x = \nabla y$ is false. \qed
• Prove for any $i \neq j$ then $0^i$ and $0^j$ are distinguishable for the language $\{0^n1^n \mid n \geq 0\}$. 
• Prove for any $i \neq j$ then $0^i$ and $0^j$ are distinguishable for the language $\{0^n1^n \mid n \geq 0\}$.

• Let $L$ be a regular language, and let $w_1, \ldots, w_k$ be strings that are all pairwise distinguishable for $L$. Prove any DFA for $L$ must have at least $k$ states.
Review questions...

• Prove for any $i \neq j$ then $0^i$ and $0^j$ are distinguishable for the language $\{0^n1^n \mid n \geq 0\}$.

• Let $L$ be a regular language, and let $w_1, \ldots, w_k$ be strings that are all pairwise distinguishable for $L$. Prove any DFA for $L$ must have at least $k$ states.

• Prove that $\{0^n1^n \mid n \geq 0\}$ is not regular.
Fooling sets: Proving non-regularity
Definition
For a language $L$ over $\Sigma$ a set of strings $F$ (could be infinite) is a fooling set or distinguishing set for $L$ if every two distinct strings $x, y \in F$ are distinguishable.

Example: $F = \{0^i | i \geq 0\}$ is a fooling set for the language $L = \{0^n1^n | n \geq 0\}$.
**Definition**
For a language $L$ over $\Sigma$ a set of strings $F$ (could be infinite) is a **fooling set** or **distinguishing set** for $L$ if every two distinct strings $x, y \in F$ are distinguishable.

**Example:** $F = \{0^i \mid i \geq 0\}$ is a fooling set for the language $L = \{0^n1^n \mid n \geq 0\}$.
Fooling Sets

Definition
For a language $L$ over $\Sigma$ a set of strings $F$ (could be infinite) is a fooling set or distinguishing set for $L$ if every two distinct strings $x, y \in F$ are distinguishable.

Example: $F = \{0^i \mid i \geq 0\}$ is a fooling set for the language $L = \{0^n1^n \mid n \geq 0\}$.

Theorem
Suppose $F$ is a fooling set for $L$. If $F$ is finite then there is no DFA $M$ that accepts $L$ with less than $|F|$ states.
Recall

Already proved the following lemma:

**Lemma**

$L$: regular language.

$M = (Q, \Sigma, \delta, s, A)$: **DFA** for $L$.

If $x, y \in \Sigma^*$ are distinguishable, then $\nabla x \neq \nabla y$.

Reminder: $\nabla x = \delta^*(s, x)$. 
Proof of theorem

_Theorem (Reworded.)_

\( L: \) A language

\( F: \) a fooling set for \( L. \)

If \( F \) is finite then any **DFA** \( M \) that accepts \( L \) has at least \( |F| \) states.

**Proof.**

Let \( F = \{w_1, w_2, \ldots, w_m\} \) be the fooling set.

Let \( M = (Q, \Sigma, \delta, s, A) \) be any **DFA** that accepts \( L. \)
Proof of theorem

Theorem (Reworded.)

$L$: A language

$F$: a fooling set for $L$.

If $F$ is finite then any DFA $M$ that accepts $L$ has at least $|F|$ states.

Proof.

Let $F = \{w_1, w_2, \ldots, w_m\}$ be the fooling set.

Let $M = (Q, \Sigma, \delta, s, A)$ be any DFA that accepts $L$.

Let $q_i = \nabla w_i = \delta^*(s, x_i)$.
Proof of theorem

**Theorem (Reworded.)**

$L$: A language

$F$: a fooling set for $L$.

*If $F$ is finite then any DFA $M$ that accepts $L$ has at least $|F|$ states.*

**Proof.**

Let $F = \{w_1, w_2, \ldots, w_m\}$ be the fooling set.

Let $M = (Q, \Sigma, \delta, s, A)$ be any DFA that accepts $L$.

Let $q_i = \nabla w_i = \delta^*(s, x_i)$.

By lemma $q_i \neq q_j$ for all $i \neq j$.

As such, $|Q| \geq |\{q_1, \ldots, q_m\}| = |\{w_1, \ldots, w_m\}| = |A|$.  

$\square$
Corollary
If \( L \) has an infinite fooling set \( F \) then \( L \) is not regular.

Proof.
Let \( w_1, w_2, \ldots \subseteq F \) be an infinite sequence of strings such that every pair of them are distinguishable.

Assume for contradiction that \( \exists M \) a DFA for \( L \).
Corollary
If L has an infinite fooling set F then L is not regular.

Proof.
Let \( w_1, w_2, \ldots \subseteq F \) be an infinite sequence of strings such that every pair of them are distinguishable.

Assume for contradiction that \( \exists M \) a DFA for L.

Let \( F_i = \{w_1, \ldots, w_i\} \).

By theorem, \# states of M \( \geq |F_i| = i \), for all i.

As such, number of states in M is infinite.
Corollary

If $L$ has an infinite fooling set $F$ then $L$ is not regular.

Proof.
Let $w_1, w_2, \ldots \subseteq F$ be an infinite sequence of strings such that every pair of them are distinguishable.

Assume for contradiction that $\exists M$ a DFA for $L$.

Let $F_i = \{w_1, \ldots, w_i\}$.

By theorem, number of states of $M \geq |F_i| = i$, for all $i$.

As such, number of states in $M$ is infinite.

Contradiction: DFA = deterministic finite automata. But $M$ not finite.
Examples

• \( \{0^n1^n \mid n \geq 0\} \)

• \{bitstrings with equal number of 0s and 1s\}
  Can use the same fooling set as before: Same logic.
  \(0^i1^i \in L\) and \(0^j1^i \notin L\) so \(\nabla 0^i\) and \(\nabla 0^j\) are distinguishable
  and so \(L\) is not regular.

• \( \{0^k1^\ell \mid k \neq \ell\} \)
  Similar logic. \(0^i1^i \notin L\) and \(0^j1^i \in L\) so \(\nabla 0^i\) and \(\nabla 0^j\) are distinguishable
  and so \(L\) is not regular.
Examples

$L = \{\text{strings of properly matched open and closing parentheses}\}$
Examples

$L = \{\text{palindromes over the binary alphabet}\Sigma = \{0, 1\}\}$
A palindrome is a string that is equal to its reversal, e.g. 10001 or 0110.
Exponential gap in number of states between DFA and NFA sizes
Exponential gap between NFA and DFA size

\[ L_4 = \{w \in \{0, 1\}^* \mid w \text{ has a 1 located 4 positions from the end} \} \]
Exponential gap between NFA and DFA size

\[ L_k = \{ w \in \{0, 1\}^* | w \text{ has a } 1 \text{ } k \text{ positions from the end} \} \]
Exponential gap between NFA and DFA size

\[ L_k = \{ w \in \{0, 1\}^* \mid w \text{ has a 1 } k \text{ positions from the end} \} \]

Recall that \( L_k \) is accepted by a NFA \( N \) with \( k + 1 \) states.
Exponential gap between NFA and DFA size

\( L_k = \{ w \in \{0, 1\}^* \mid w \text{ has a } 1 \text{ } k \text{ positions from the end} \} \)

Recall that \( L_k \) is accepted by a NFA \( N \) with \( k + 1 \) states.

**Theorem**

*Every DFA that accepts \( L_k \) has at least \( 2^k \) states.*
Exponential gap between NFA and DFA size

$L_k = \{ w \in \{0, 1\}^* \mid w \text{ has a } 1 \text{ \textit{k} positions from the end} \}$

Recall that $L_k$ is accepted by a NFA $N$ with $k + 1$ states.

**Theorem**
Every DFA that accepts $L_k$ has at least $2^k$ states.

**Claim**
$F = \{ w \in \{0, 1\}^* : |w| = k \}$ is a fooling set of size $2^k$ for $L_k$.

Why?
How do we pick a fooling set $F$?

- If $x, y$ are in $F$ and $x \neq y$ they should be distinguishable! Of course.
- All strings in $F$ except maybe one should be prefixes of strings in the language $L$.
  For example if $L = \{0^k1^k \mid k \geq 0\}$ do not pick 1 and 10 (say). Why?
Myhill–Nerode Theorem
“Myhill-Nerode Theorem”: A regular language $L$ has a unique (up to naming) minimal automata, and it can be computed efficiently once any DFA is given for $L$. 
Recall:

**Definition**
For a language $L$ over $\Sigma$ and two strings $x, y \in \Sigma^*$ we say that $x$ and $y$ are *distinguishable* with respect to $L$ if there is a string $w \in \Sigma^*$ such that exactly one of $xw, yw$ is in $L$. $x, y$ are *indistinguishable* with respect to $L$ if there is no such $w$.

Given language $L$ over $\Sigma$ define a relation $\equiv_L$ over strings in $\Sigma^*$ as follows: $x \equiv_L y$ iff $x$ and $y$ are indistinguishable with respect to $L$. 
Recall:

**Definition**
For a language $L$ over $\Sigma$ and two strings $x, y \in \Sigma^*$ we say that $x$ and $y$ are distinguishable with respect to $L$ if there is a string $w \in \Sigma^*$ such that exactly one of $xw, yw$ is in $L$. $x, y$ are indistinguishable with respect to $L$ if there is no such $w$.

Given language $L$ over $\Sigma$ define a relation $\equiv_L$ over strings in $\Sigma^*$ as follows: $x \equiv_L y$ iff $x$ and $y$ are indistinguishable with respect to $L$.

**Definition**
$x \equiv_L y$ means that $\forall w \in \Sigma^*: xw \in L \iff yw \in L$.

In words: $x$ is equivalent to $y$ under $L$. 
Example: Equivalence classes
Claim

\(\equiv_L\) is an equivalence relation over \(\Sigma^*\).

Proof.

• Reflexive: \(\forall x \in \Sigma^*: \forall w \in \Sigma^*: xw \in L \iff xw \in L\).
  \[\implies x \equiv_L x.\]

• Symmetry: \(x \equiv_L y\) then \(\forall w \in \Sigma^*: xw \in L \iff yw \in L\)
  \(\forall w \in \Sigma^*: yw \in L \iff xw \in L \implies y \equiv_L x.\)

• Transitivity: \(x \equiv_L y\) and \(y \equiv_L z\)
  \(\forall w \in \Sigma^*: xw \in L \iff yw \in L\) and \(\forall w \in \Sigma^*: yw \in L \iff zw \in L\)
  \[\implies \forall w \in \Sigma^*: xw \in L \iff zw \in L\]
  \[\implies x \equiv_L z.\]
Claim
≡_L is an equivalence relation over \( \Sigma^* \).
Therefore, \( \equiv_L \) partitions \( \Sigma^* \) into a collection of equivalence classes.

Definition
\( L \): A language
For a string \( x \in \Sigma^* \), let
\[
[x] = [x]_L = \{ y \in \Sigma^* \mid x \equiv_L y \}
\]
be the equivalence class of \( x \) according to \( L \).

Definition
\( [L] = \{ [x]_L \mid x \in \Sigma^* \} \) is the set of equivalence classes of \( L \).
Claim

Let $x, y$ be two distinct strings. If $x, y$ belong to the same equivalence class of $\equiv_L$ then $x, y$ are indistinguishable. Otherwise they are distinguishable.
Lemma

Let $x, y$ be two distinct strings.

$x \equiv_L y \iff x, y$ are indistinguishable for $L$.

Proof.

$x \equiv_L y \implies \forall w \in \Sigma^*: xw \in L \iff yw \in L$

$x$ and $y$ are indistinguishable for $L$.

$x \not\equiv_L y \implies \exists w \in \Sigma^*: xw \in L$ and $yw \not\in L$

$\implies x$ and $y$ are distinguishable for $L$. 
Lemma
$M = (Q, \Sigma, \delta, s, A)$ a DFA for a language $L$.

For any $q \in A$, let $L_q = \{w \in \Sigma^* \mid \nabla w = \delta^*(s, w) = q\}$.

Then, there exists a string $x$, such that $L_q \subseteq [x]_L$. 
General idea behind algorithm:

**Base case:** Given two states, if $p$ and $q$, if one accepts and the other rejects, then they are not equivalent.

**Recursion:** Assuming $p \xrightarrow{a} p'$ and $q \xrightarrow{a} q'$, if $p' \neq q'$ then $p \neq q$
An inefficient automata

The diagram shows a state transition graph with states labeled $q_0, q_1, q_2, q_3, q_4, q_5$. The transitions are labeled with inputs of 0, indicating movement from one state to another. The table below represents the transition probabilities:

<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>
<th>$q_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q_0$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$q_1$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$q_2$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$q_3$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$q_4$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$q_5$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>