Pre-lecture brain teaser

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

$L = \{0^n 1^n \mid n \geq 0\}$

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

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

$L = \{0^n 1^n \mid n \geq 0\}$

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

Closure

$L = L' \cap L(0^*1^*)$

If L' was regular, then L would have to be regular

Since L' is not regular

Non-regularity via closure properties

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

$L = \{0^n 1^n \mid n \geq 0\}$

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.

Outside of course scope

Can help

Pre-lecture brain teaser

We have a language $L = \{0^n 1^n | n \geq 0\}$

Prove that L is non-regular.

Not all languages are regular

Regular Languages, DFAs, NFAs

Theorem
Languages accepted by DFAs, NFAs, and regular expressions are the same.

Question: Is every language a regular language? **No.**

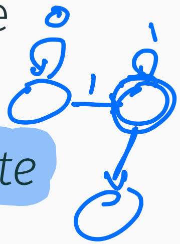
Theorem
Languages accepted by DFAs, NFAs, and regular expressions are the same.

have a finite # of states

Question: Is every language a regular language? **No.**

- Each DFA M can be represented as a string over a finite alphabet Σ 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!

0*1*



A Simple and Canonical Non-regular Language

$L = \{0^n 1^n \mid n \geq 0\} = \{\epsilon, 01, 0011, 000111, \cdots, \}$

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

$L = \{0^n 1^n \mid n \geq 0\} = \{\epsilon, 01, 0011, 000111, \cdots, \}$

Theorem
L is not regular.

Question: Proof?

A Simple and Canonical Non-regular Language

$L = \{0^n 1^n \mid n \geq 0\} = \{\epsilon, 01, 0011, 000111, \cdots, \}$

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

$L = \{0^n 1^n \mid n \geq 0\} = \{\epsilon, 01, 0011, 000111, \cdots, \}$

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 $|Q| = n$.

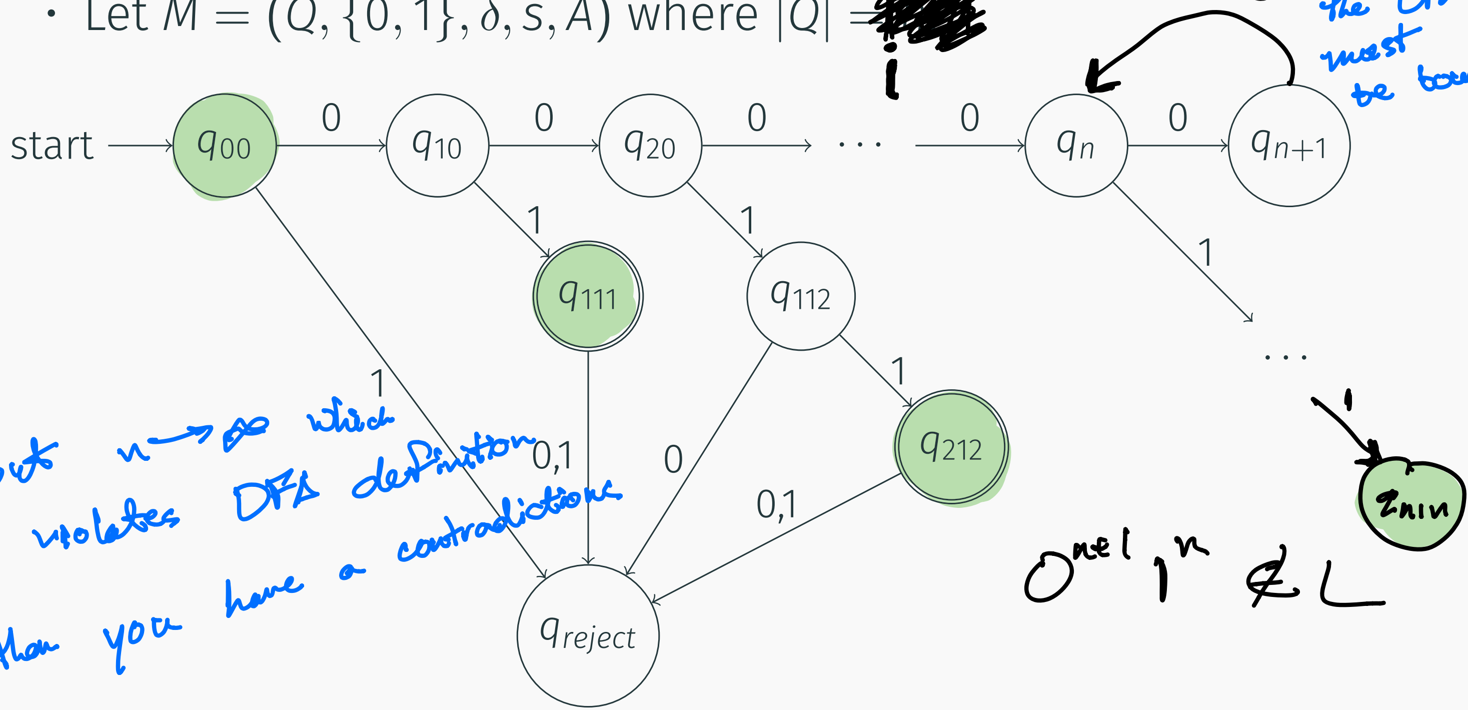
each substring 0^i must have a separate state

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

$L = 0^n1^n := \{\varepsilon, 01, 0011, 000111, \ldots\}$

- Let $M = (Q, \{0,1\}, \delta, s, A)$ where $|Q| =$ [redacted] i

Because the # of states in the DFA must be bound



but $n \to \infty$ which violates DFA definition

then you have a contradiction

$0^{n+1}1^n \notin L$

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

For each of these strings we need to reach a different state

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

if $n > m$ you have n items and m spaces then >0 spaces must have more than one item

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 $i \neq j$.

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.

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 $\epsilon, 0, 00, 000, \cdots, 0^n$ 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 $0^i 1^i$ but then it will also accept $0^j 1^i$ where $i \neq j$.

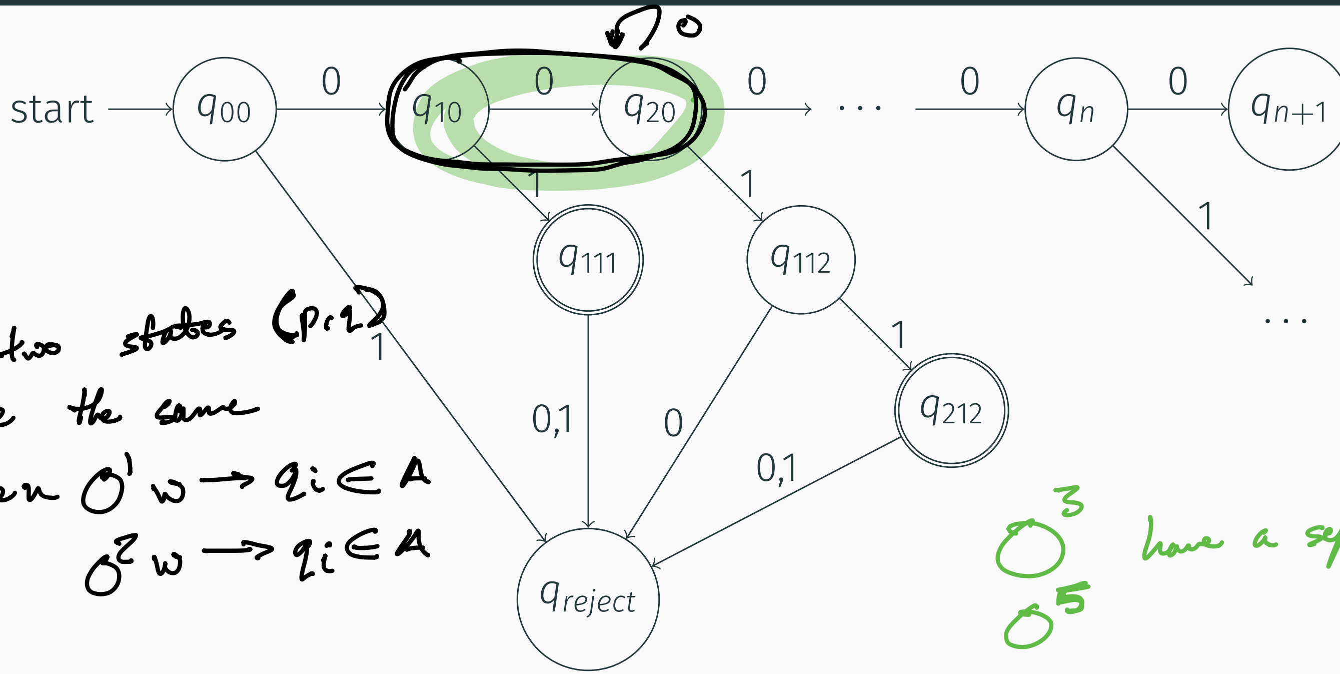
This contradicts the fact that M accepts L. Thus, there is no DFA

When two states are equivalent?

O^i & O^j must have seperate

states

States that cannot be combined?



If two states (p, q) are the same then $0^1 w \to q_i \in A$, $0^2 w \to q_i \in A$

0^3, 0^5 have a sep state

We concluded that because each 0^i prefix has a unique state.

Are there states that aren't unique? Lets combine 0^1 & 0^2

Can states be combined?

Equivalence between states

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

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

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

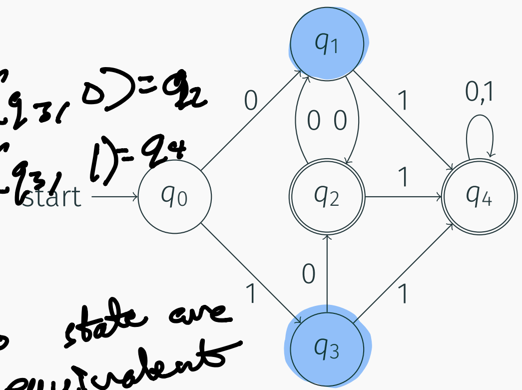
← two state are equivalent

if they result in the same accept behavior for all input strings

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

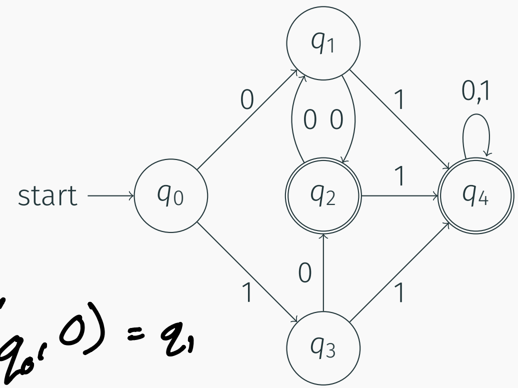
$\delta(q_1, 0) = q_2$
$\delta(q_1, 1) = q_4$

$\delta(q_3, 0) = q_2$
$\delta(q_3, 1) = q_4$



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



$$\delta^*(p, w) \in A \qquad \text{and} \qquad \delta^*(q, w) \notin A.$$

$\delta(q_0, 0) = q_1$
$\delta(q_0, 1) = q_3$

or

$$\delta^*(p, w) \notin A \qquad \text{and} \qquad \delta^*(q, w) \in A.$$

$\delta(q_3, 0) = q_2$ (A)
$\delta(q_3, 1) = q_4$ (A)

Distinguishable prefixes

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

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

$= q_w$

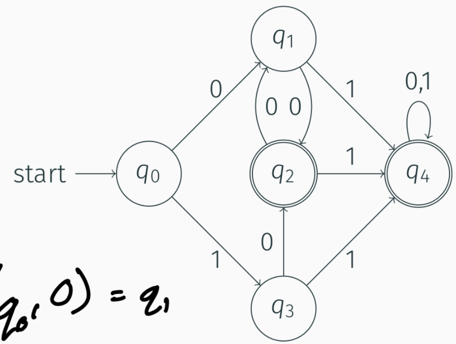
$q_i = \delta^*(s, w)$

$\nabla(0) \equiv \nabla(1)$

$\nabla(\varepsilon) \not\equiv \nabla(1)$

$\nabla(00) = q_2$

$\nabla(10) = q_2$



$\delta(q_0, 0) = q_1$

$\nabla(0001) = q_4$

$\delta(q_0, 1) = q_3$

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 ∇u and ∇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 ∇u and ∇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 $wx \in L$).

A $\quad$ A

If $\nabla u \equiv \nabla w \Rightarrow q_i$ $\quad$ $\delta^*(q_i, x) \Rightarrow q_x \in A$ / $\notin A$

equivalent

If $\delta^*(s, ux) \in A$ $\quad$ $\delta^*(s, ux) = \delta^*(s, wx) \Rightarrow q_x$

$\delta^*(s, wx) \notin A$

Distinguishable means different states

Lemma
L: regular language.

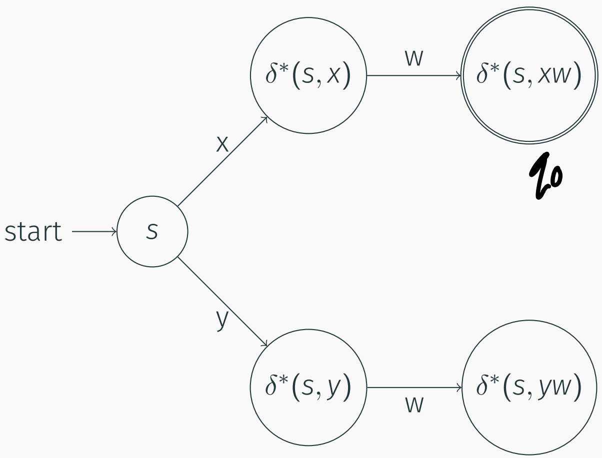
$M = (Q, \Sigma, \delta, s, A)$: DFA for L.

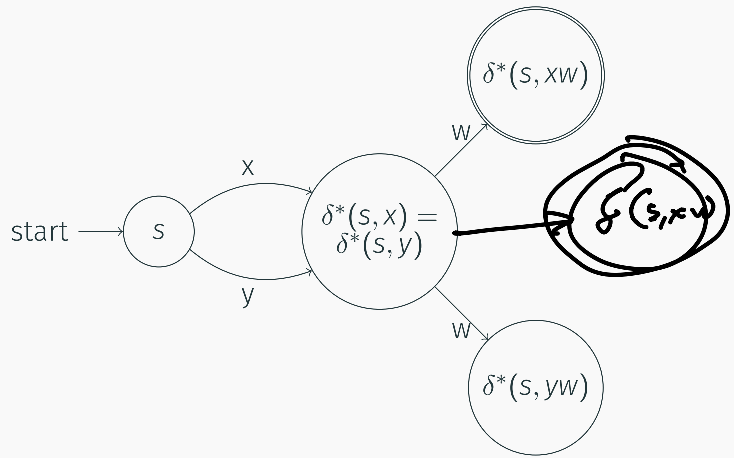
If $x, y \in \Sigma^*$ are distinguishable, then $\nabla x \neq \nabla y$.

Reminder: $\nabla x = \delta^*(s, x) \in Q$ and $\nabla y = \delta^*(s, y) \in Q$

Proof by a figure

Possible	Not possible





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 $\nabla yw \notin A$.

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 $\nabla yw \notin A$.

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

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 $\nabla yw \notin A$.

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

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 $\nabla yw \notin A$.

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

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 $\nabla yw \notin A$.

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

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

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 $\nabla yw \notin A$.

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

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

Assumption that $\nabla x = \nabla y$ is false. □

- Prove for any $i \neq j$ then 0^i and 0^j are distinguishable for the language $\{0^n 1^n \mid n \geq 0\}$.

$uw \in A$ u, v are distinguishable

$vw \notin A$ $u = 0^i$ $v = 0^j$ $w = 1^i$

$\forall w$ $0^i 1^i \in A$ thus 0^i & 0^j are distinguishable

$0^j 1^i \notin A$

Review questions...

- Prove for any $i \neq j$ then 0^i and 0^j are distinguishable for the language $\{0^n 1^n \mid n \geq 0\}$.
- 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.

$\nabla w_i = q_i \quad Q = \{q_1 \ldots \ldots q_k\} \quad |Q| = k$ or more

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.

use

0^i & 0^j are distinguishable

For every string $\nabla 0^n = q$ in therefore

DFA must have atleast n states

Since $n \to \infty$ DFA not possible

L not regular

Fooling sets: Proving non-regularity

Fooling Sets

Definition
For a language L over Σ 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.

Definition
For a language L over Σ 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 $L = \{0^n 1^n \mid n \geq 0\}$.

Fooling Sets

Definition
For a language L over Σ 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 $q_i = \nabla w_i = \delta^*(s, w_i)$.

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

Infinite Fooling Sets

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.

Infinite Fooling Sets

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.

Infinite Fooling Sets

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.

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

Examples

- $\{0^n1^n \mid n \geq 0\}$ $F = \{0^i \mid i > 0\}$
  0^i & 0^j are distinguishable because $w = 1^i$
- {bitstrings with equal number of 0s and 1s} $\supset \{0^n1^i \mid i > 0\}$
  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.
  $u = 0^i$ $w = 1^i$ $uw \notin L$
  $v = 0^j$ $vw \in L$

Examples

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

$L = \{\text{palindromes over the binary alphabet} \Sigma = \{0, 1\}\}$

A palindrome is a string that is equal to its reversal, e.g. 10001 or 0110.

$F = \{(01)^i \mid i > 0\}$ → hence all prefixes are distinguishable

$x \neq y$

$u = (01)^x$

$v = (01)^y$

$w = (10)^x$

$uw \in L$

$vw \notin L$

$|F| \to \infty$

Thus M cannot exist

L(M) not regular

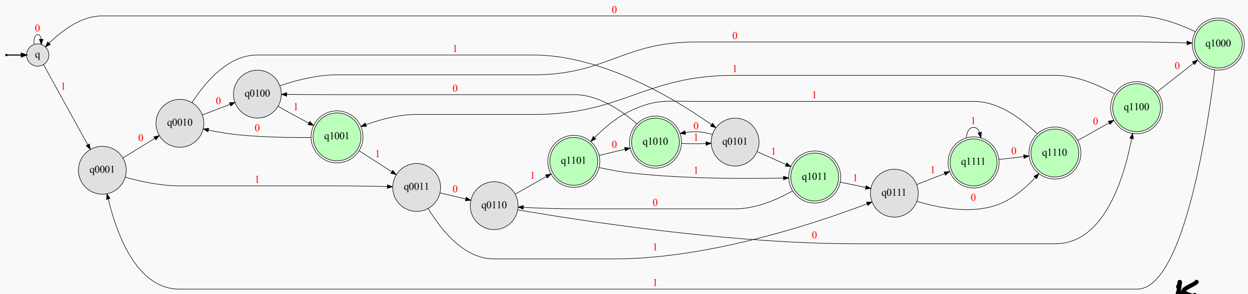
Exponential gap in number of states between DFA and NFA sizes

Exponential gap between NFA and DFA size

$= k$

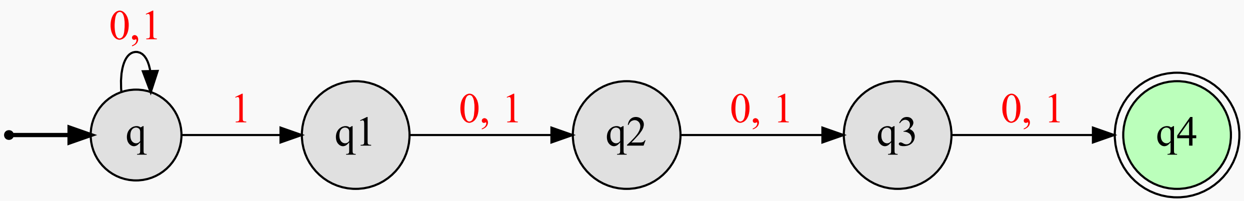
$L_4 = \{w \in \{0,1\}^* \mid w$ has a 1 located 4 positions from the end$\}$

DFA:



2^k

NFA:



k+1 states

Exponential gap between NFA and DFA size

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

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

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

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

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

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

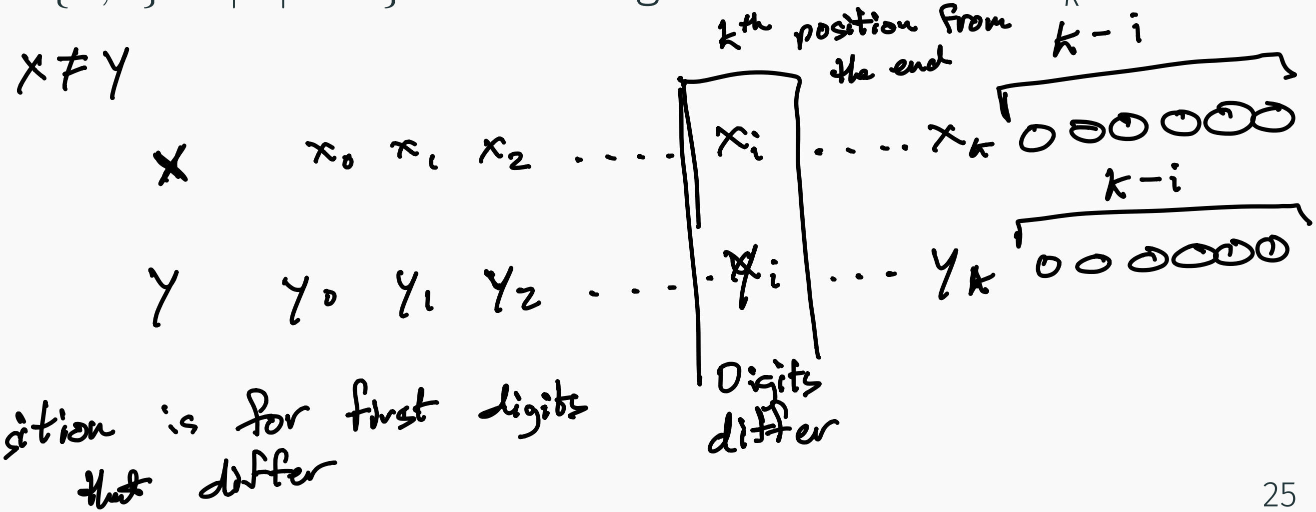
Theorem

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

Claim

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

Why?



How do pick a fooling set

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 $L = \{0^k 1^k \mid k \geq 0\}$ do not pick 1 and 10 (say).
  Why?

Myhill-Nerode Theorem

One automata to rule them all

"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.

Indistinguishably

Recall:

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

Indistinguishably

Recall:

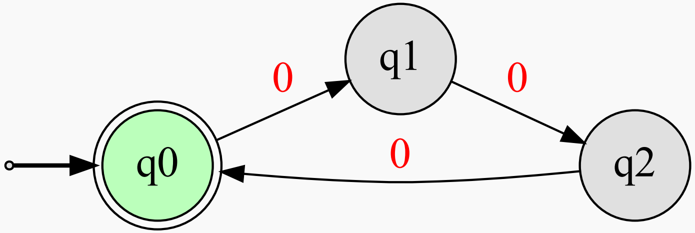
Definition
For a language L over Σ and two strings $x, y \in \Sigma^*$ we say that x and y are distinguishable with respect to L if there is a string $w \in \Sigma^*$ such that exactly one of xw, yw is in L. x, y are indistinguishable with respect to L if there is no such w.

Given language L over Σ define a relation $\equiv_L$ over strings in Σ^* 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



Indistinguishability

Claim

$\equiv_L$ is an equivalence relation over Σ^*.

Proof.

- Reflexive: $\forall x \in \Sigma^*\colon \forall w \in \Sigma^*\colon xw \in L \iff xw \in L$.
  $\implies x \equiv_L x$.
- Symmetry: $x \equiv_L y$ then $\forall w \in \Sigma^*\colon xw \in L \iff yw \in L$
  $\forall w \in \Sigma^*\colon yw \in L \iff xw \in L \implies y \equiv_L x$.
- Transitivity: $x \equiv_L y$ and $y \equiv_L z$
  $\forall w \in \Sigma^*\colon xw \in L \iff yw \in L$ and $\forall w \in \Sigma^*\colon yw \in L \iff zw \in L$
  $\implies \forall w \in \Sigma^*\colon xw \in L \iff zw \in L$
  $\implies x \equiv_L z$.

□

Equivalences over automatas...

Claim

$\equiv_L$ is an equivalence relation over Σ^*.
Therefore, $\equiv_L$ partitions Σ^* 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

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.

Strings in the same equivalence class are indistinguishable

Lemma
Let x, y be two distinct strings.

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

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

x and y are indistinguishable for L.

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

$\implies$ x and y are distinguishable for L.

□

All strings arriving at a state are in the same class

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 $L_q \subseteq [x]_L$.

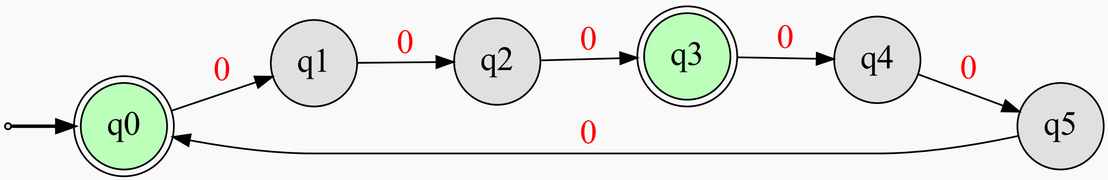
An inefficient automata

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 $p' \not\equiv q'$ then $p \not\equiv q$

An inefficient automata



	q_0	q_1	q_2	q_3	q_4	q_5
q_1						
q_2						
q_3						
q_4						
q_5						