Non-regularity and fooling sets

Sides based on material by Profs. Kani, Erickson, Chekuri, et. al.

All mistakes are my own! - Ivan Abraham (Fall 2024)

Image by ChatGPT (probably collaborated with DALL-E)

Goal of lecture

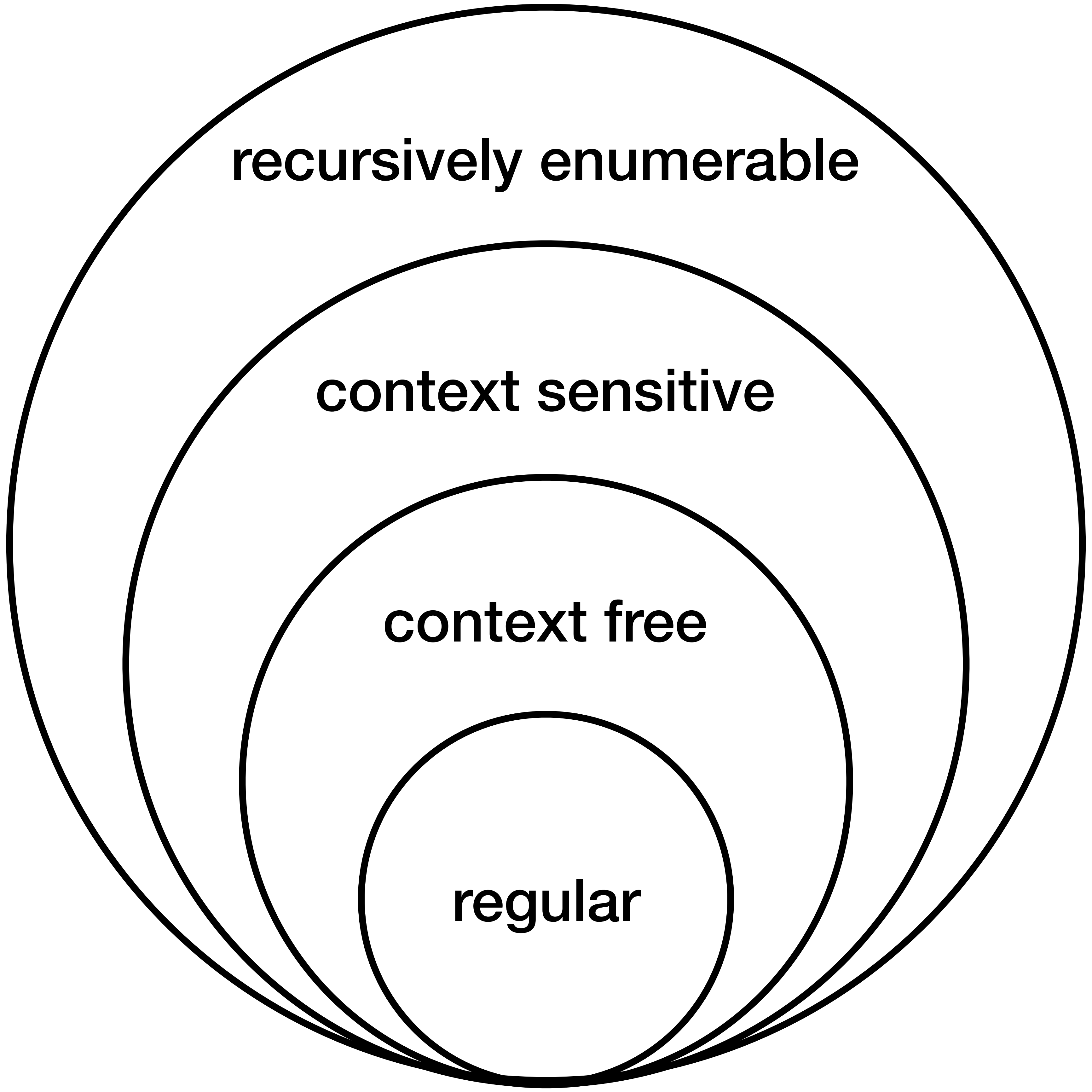
Introduce the next computability class

- So far, we have dealt with regular languages - if we bothered to name some as **regular**, are there some that *aren't regular*?
  - Irregular? Non-regular?
  - Indeed, one goal of the first part of 374 is to introduce the computability classes - ***Chomsky's Hierarchy***

Goal of lecture

Introduce the next computability class

- So far, we have dealt with regular languages - if we bothered to name some as **regular**, are there some that *aren't regular*?
    - Irregular? Non-regular?
    - Indeed, one goal of the first part of 374 is to introduce the computability classes - ***Chomsky's Hierarchy***

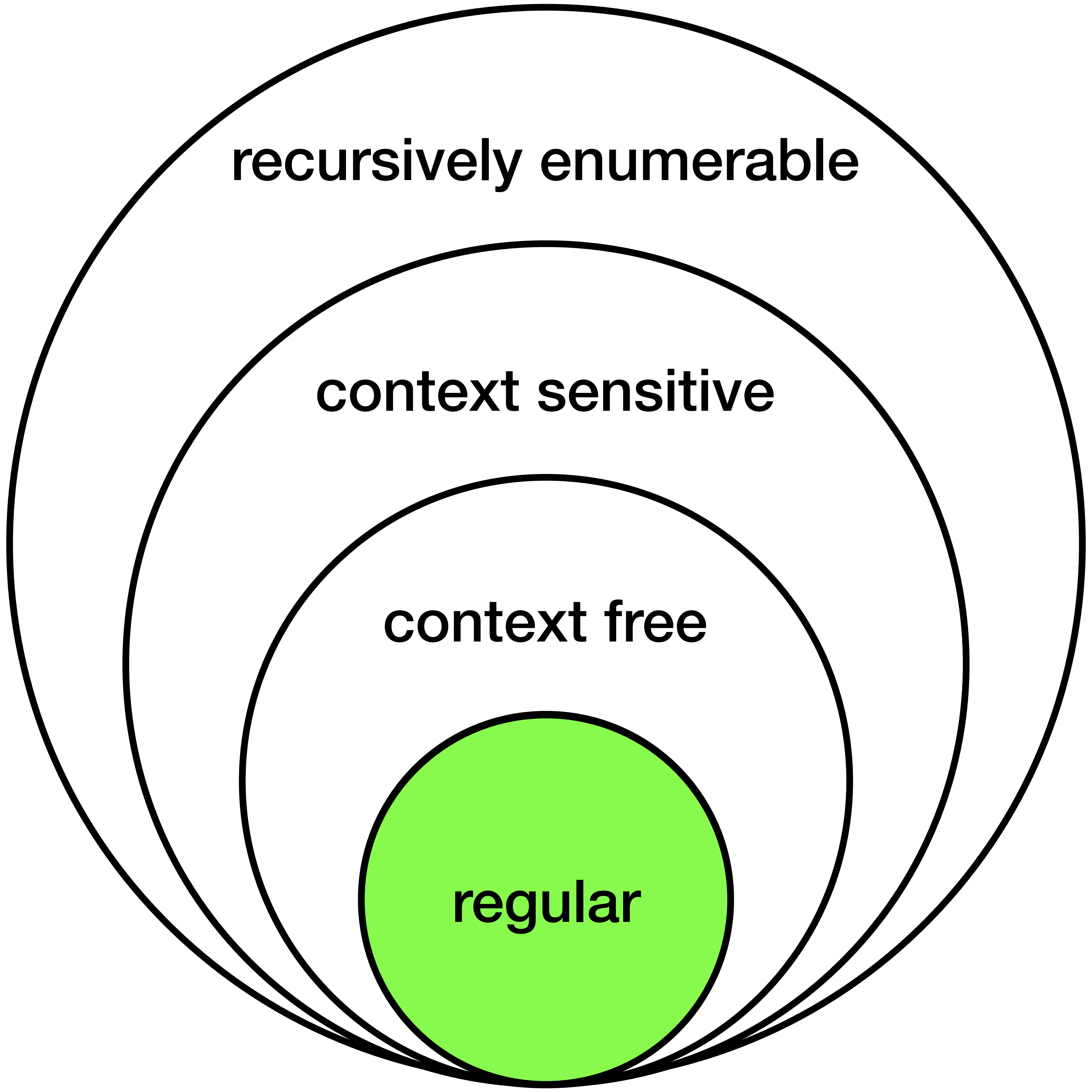


Source: Kani Archive

Goal of lecture

Introduce the next computability class

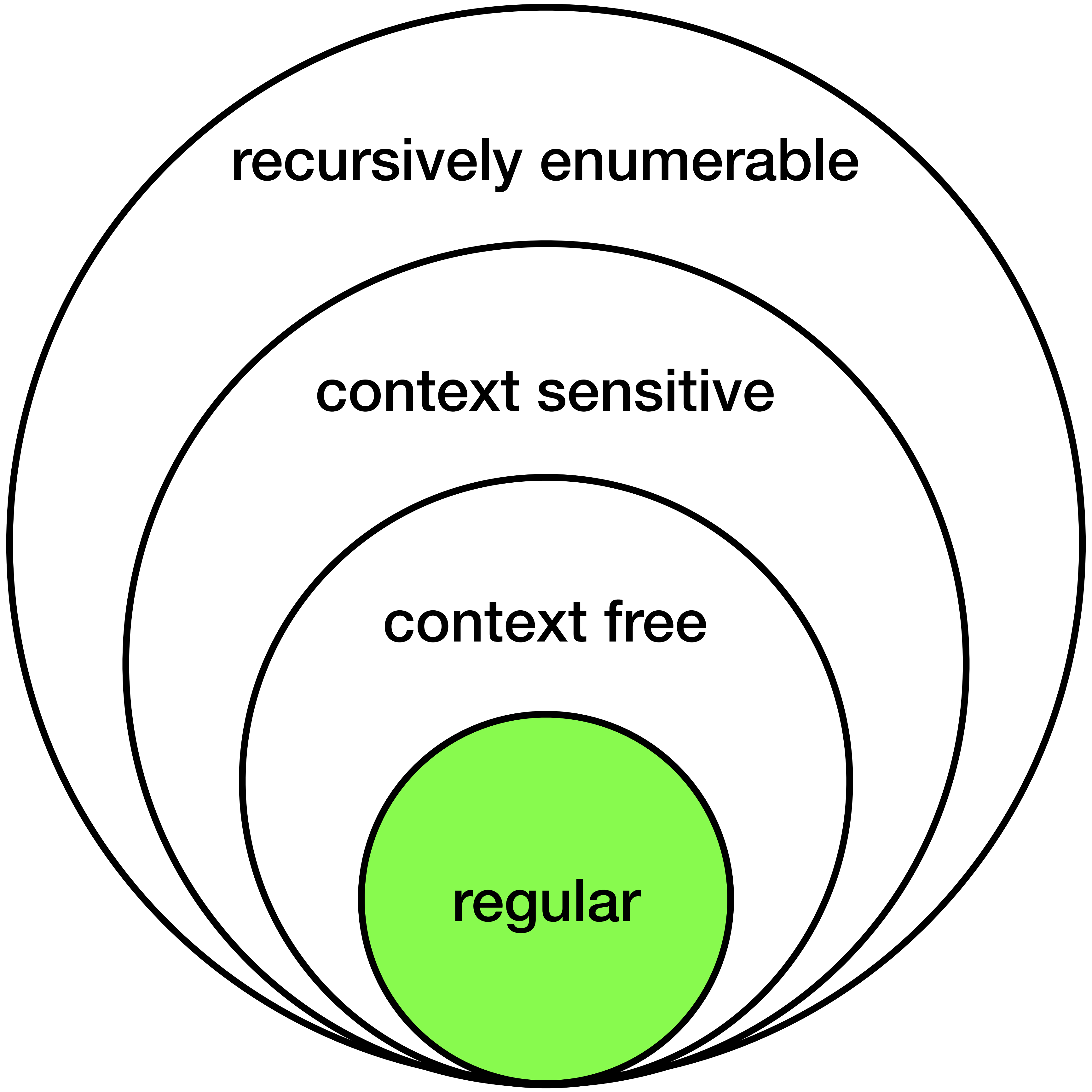
- So far, we have dealt with regular languages - if we bothered to name some as **regular**, are there some that *aren't regular*?
    - Irregular? Non-regular?
    - Indeed, one goal of the first part of 374 is to introduce the computability classes - ***Chomsky's Hierarchy***



Source: Kani Archive

Lecture outline

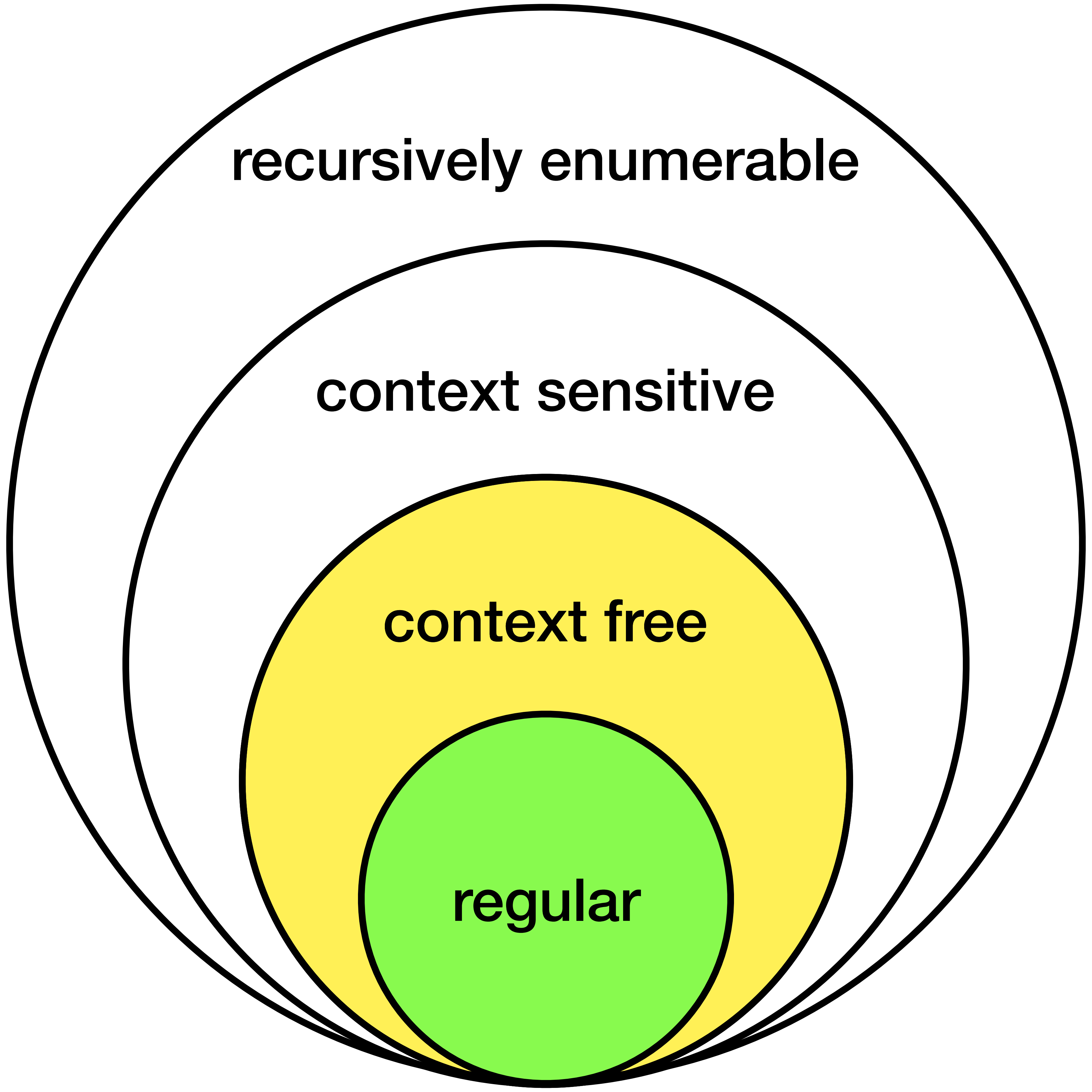
- Introduce non-regular languages
  - An argument for existence



Source: Kani Archive

Lecture outline

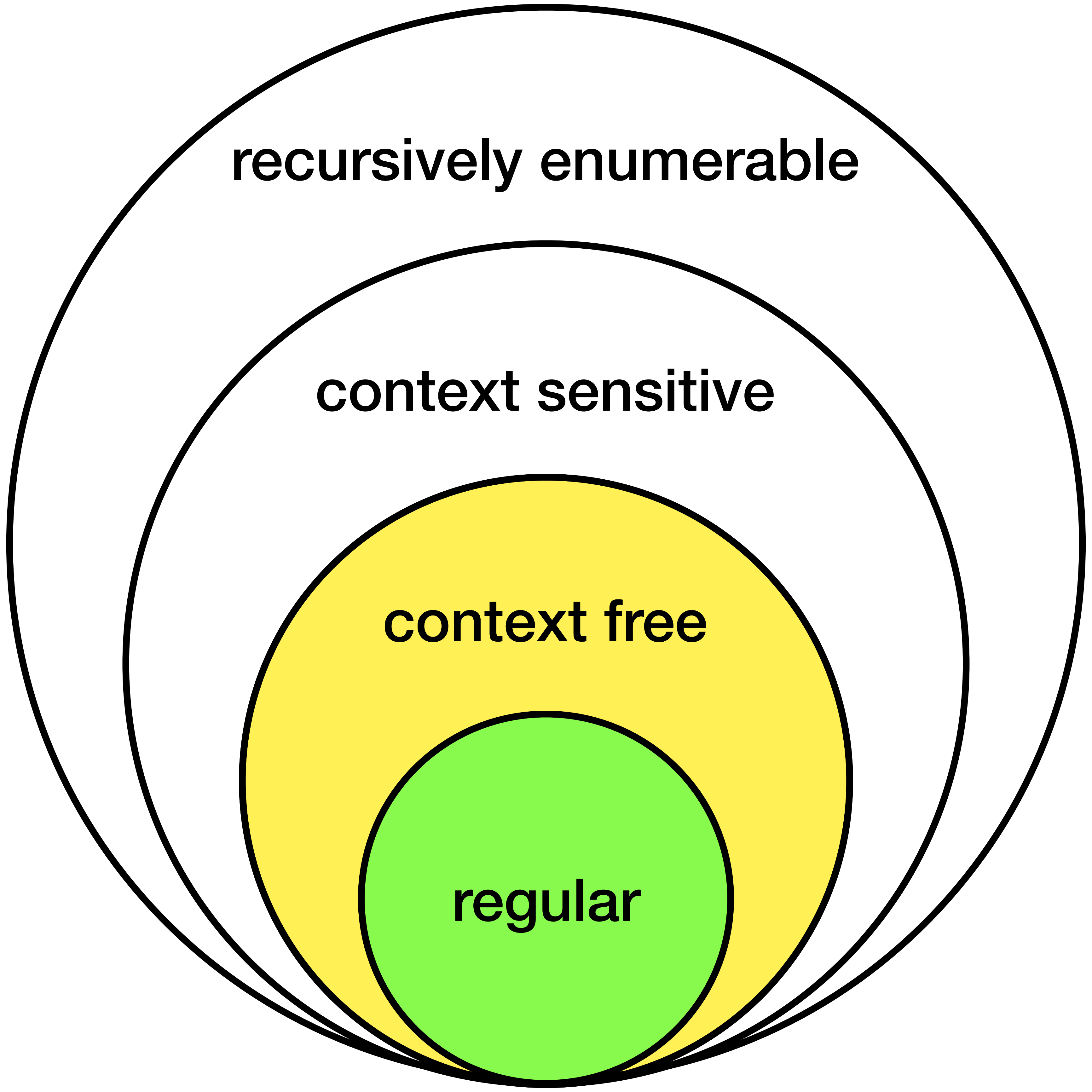
- Introduce non-regular languages
  - An argument for existence
  - A classic example of a non-regular language - a context-free language



Source: Kani Archive

Lecture outline

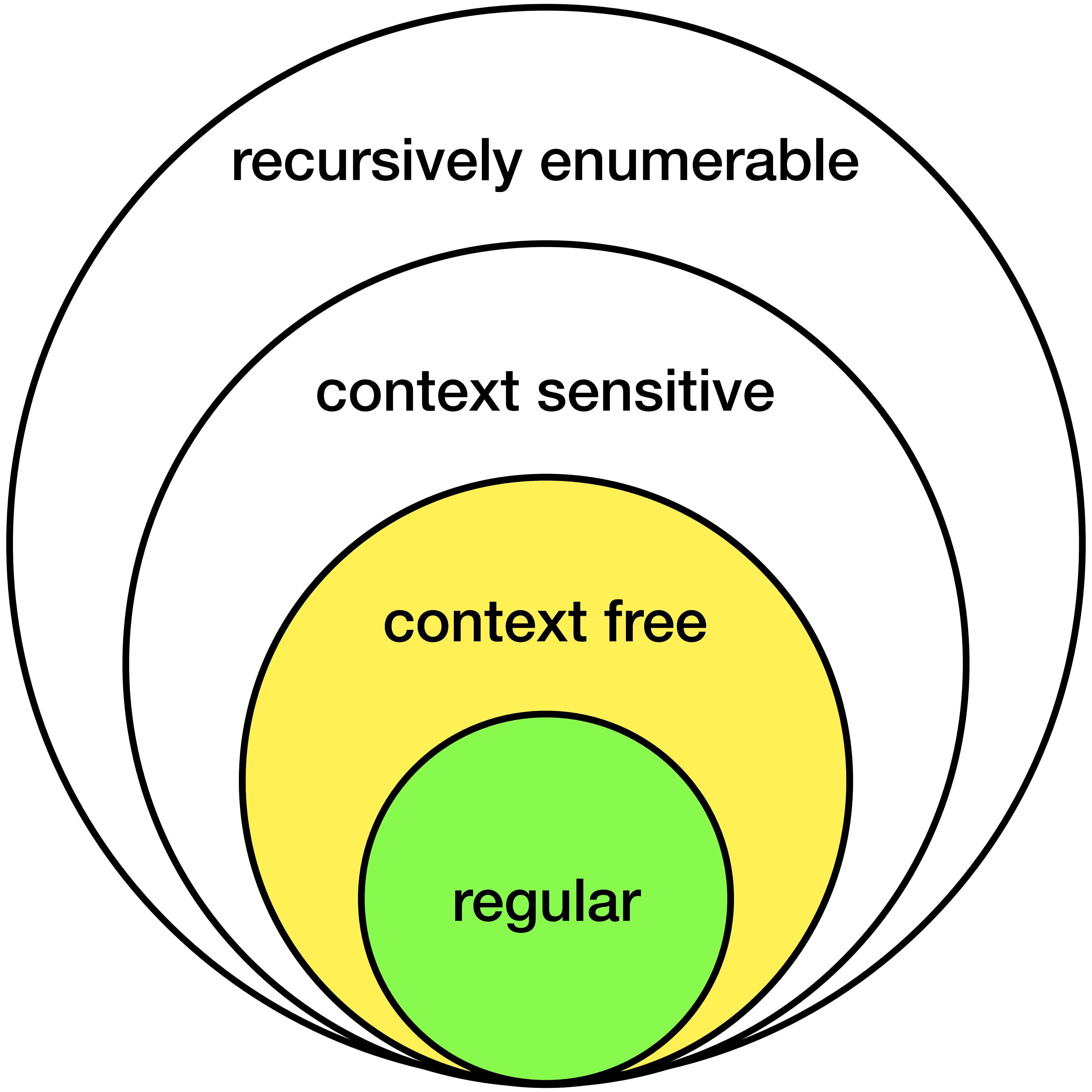
- Introduce non-regular languages
  - An argument for existence
  - A classic example of a non-regular language - a context-free language
  - Methods for showing when a language is non-regular



Source: Kani Archive

Lecture outline

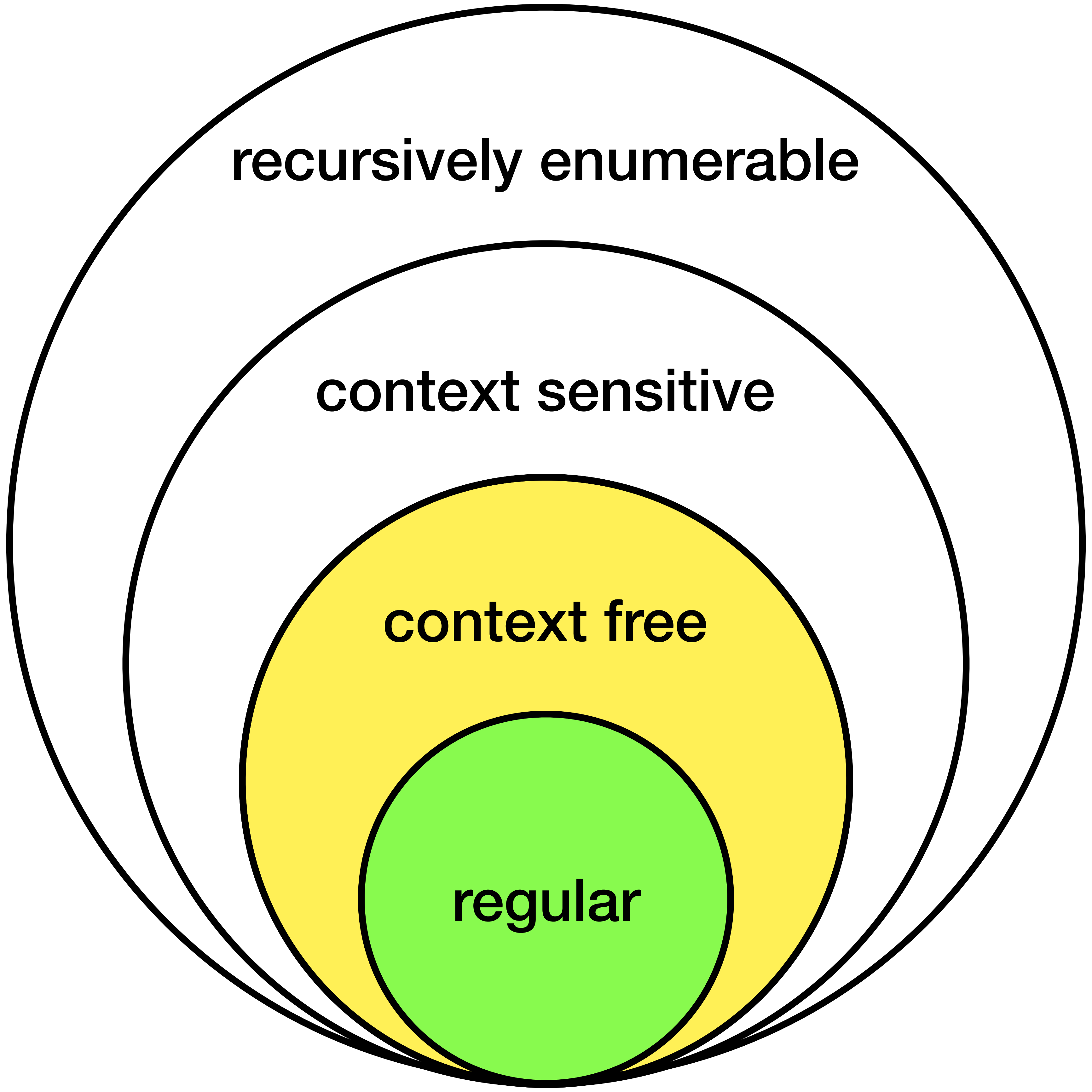
- Introduce non-regular languages
  - An argument for existence
  - A classic example of a non-regular language - a context-free language
  - Methods for showing when a language is non-regular
    - Fooling sets & closure properties



Source: Kani Archive

Lecture outline

- Introduce non-regular languages
  - An argument for existence
  - A classic example of a non-regular language - a context-free language
  - Methods for showing when a language is non-regular
    - Fooling sets & closure properties
    - Myhill-Nerode Theorem



Source: Kani Archive

What languages are non-regular?

Are there non-regular languages to begin with?

- Recall Kleene's theorem:

The classes of languages accepted by DFAs, NFAs, and regular expressions are the same.

represented by

What languages are non-regular?

Are there non-regular languages to begin with?

- Recall Kleene's theorem:

The classes of languages accepted by DFAs, NFAs, and regular expressions are the same.

- **Question**: Why should non-regular language exist? What if the above class (regular languages) are the *only* kind of languages?

Basic question: What is the cardinality / size of an infinite set and how does it compare to the cardinality of its power set?

$L = \{0^n 1^p \mid p < n\}$

$\Sigma = \{0, 1\}$, Σ^*

Non-regular languages

Existence of non-regular languages

- Integers can be counted (or put in 1-1 correspondence) - called ***countably infinite.***

Non-regular languages

Existence of non-regular languages

- Integers can be counted (or put in 1-1 correspondence) - called ***countably infinite.***

- The real numbers are uncountable (c.f. *Cantor's diagonalization argument*) — called ***uncountably infinite.***

Non-regular languages

Existence of non-regular languages

watch youtube.

- Integers can be counted (or put in 1-1 correspondence) - called ***countably infinite.***
- The real numbers are uncountable (c.f. *Cantor's diagonalization argument*) — called ***uncountably infinite.***
- Similarly, while the class of regular languages is countably infinite, the set of all languages is uncountably infinite.

Non-regular languages

Existence of non-regular languages

- Integers can be counted (or put in 1-1 correspondence) - called ***countably infinite.***
- The real numbers are uncountable (c.f. *Cantor's diagonalization argument*) — called ***uncountably infinite.***
- Similarly, while the class of regular languages is countably infinite, the set of all languages is uncountably infinite.
  - In other words, there must exist languages that are not regular.

Non-regular languages

Existence of non-regular languages

- Integers can be counted (or put in 1-1 correspondence) - called ***countably infinite.***
- The real numbers are uncountable (c.f. *Cantor's diagonalization argument*) — called ***uncountably infinite.***
- Similarly, while the class of regular languages is countably infinite, the set of all languages is uncountably infinite.
    - In other words, there must exist languages that are not regular.
    - This isn't a "proof," but we can readily provide an example of a non-regular language

A simple and canonical non-regular language

$$L_1 = \{0^n 1^n \mid n \geq 0\} = \{\epsilon, 01, 0011, 000111, \ldots\}$$

Lemma: L_1 is not regular.

Question: Proof?

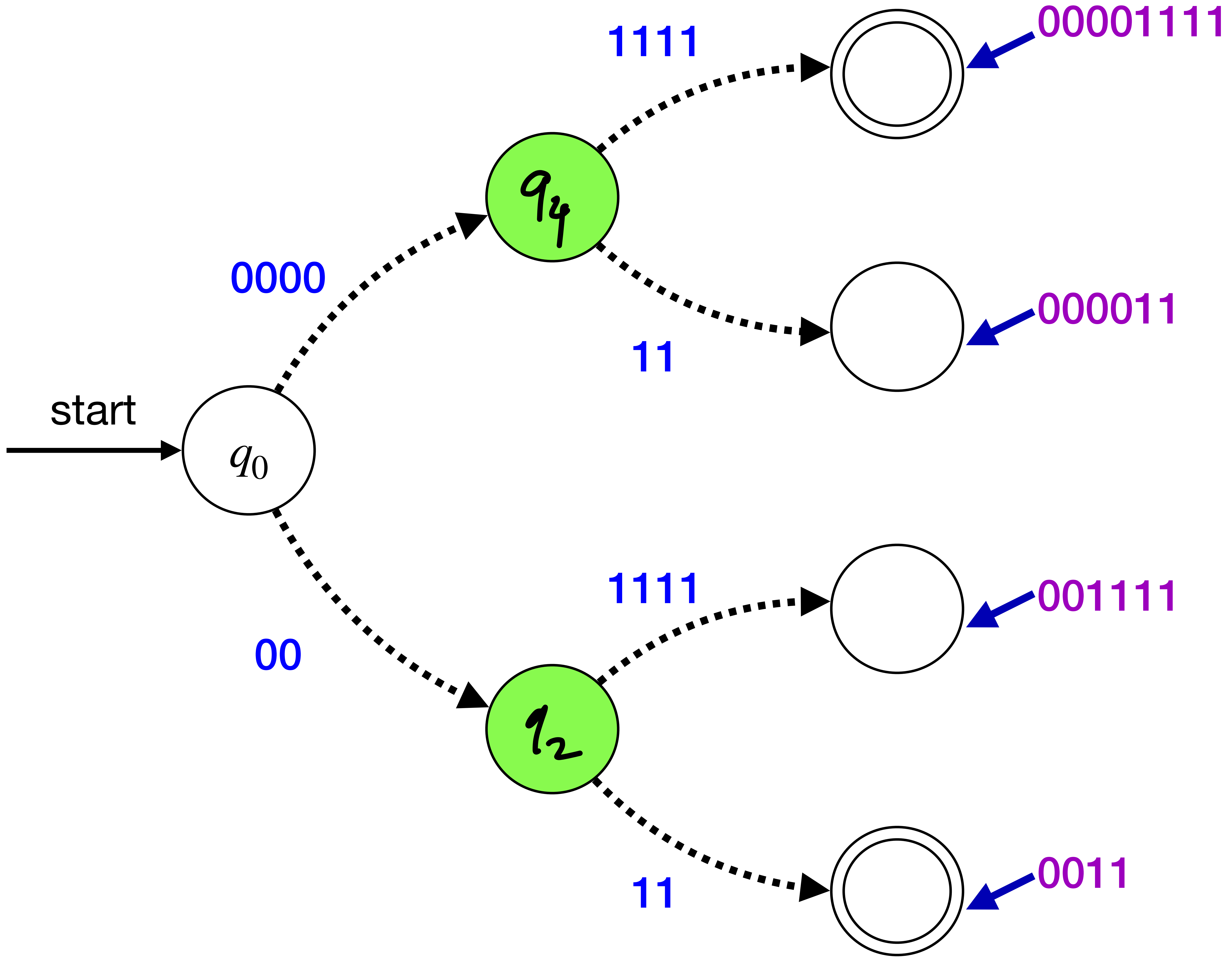
Intuition: Any program that recognizes L seems to require counting the number of zeros in the input so that it can then compare it to the number of ones—*this cannot be done with fixed memory for all n.*

How do we formalize intuition and come up with a proof?

A simple and canonical non-regular language

Building intuition

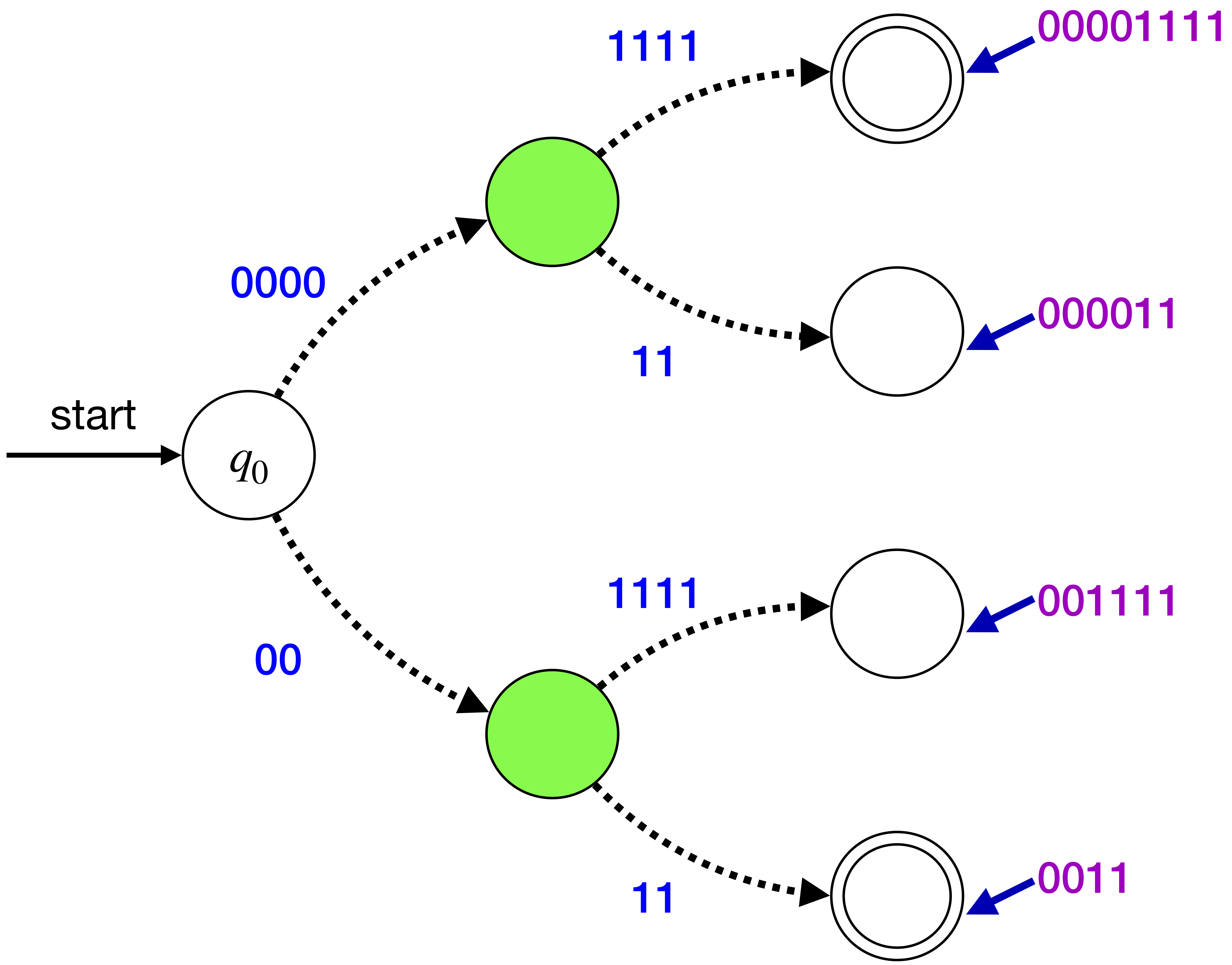
- Can the two green colored states be the same?



A simple and canonical non-regular language

Building intuition

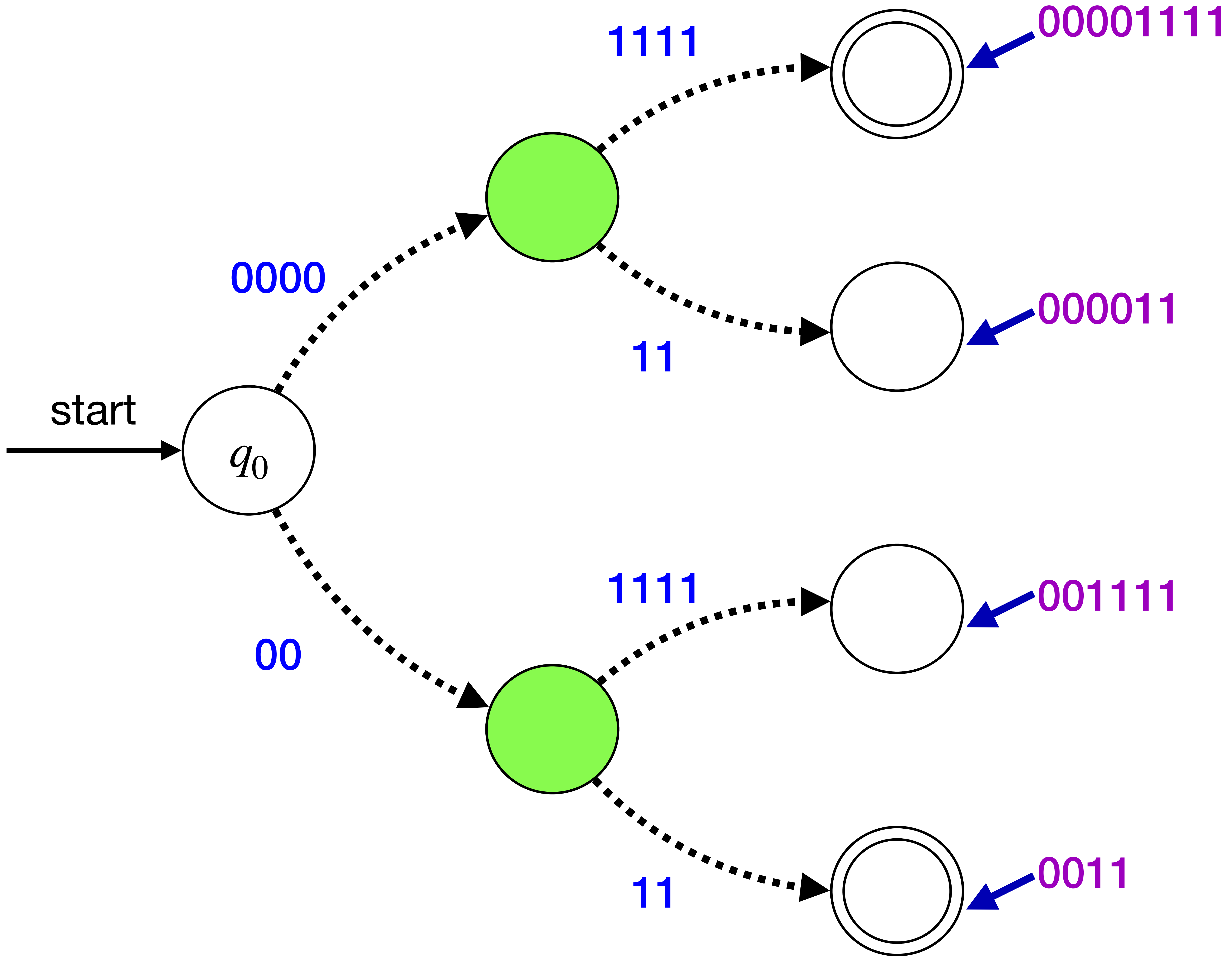
- Can the two green colored states be the same?
  - What happens if they are?



A simple and canonical non-regular language

Building intuition

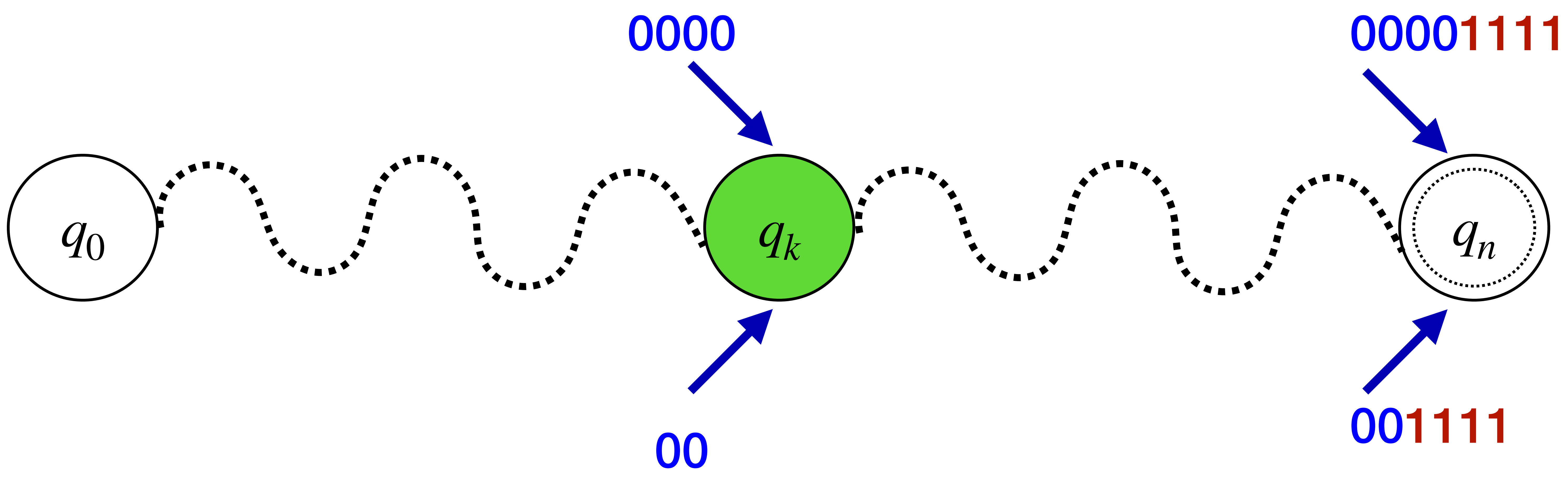
- Can the two green colored states be the same?
    - What happens if they are?
    - Suppose they are the same …



A simple and canonical non-regular language

Building intuition

- Can the two green colored states be the same?
  - What happens if they are?
  - Suppose they are the same …



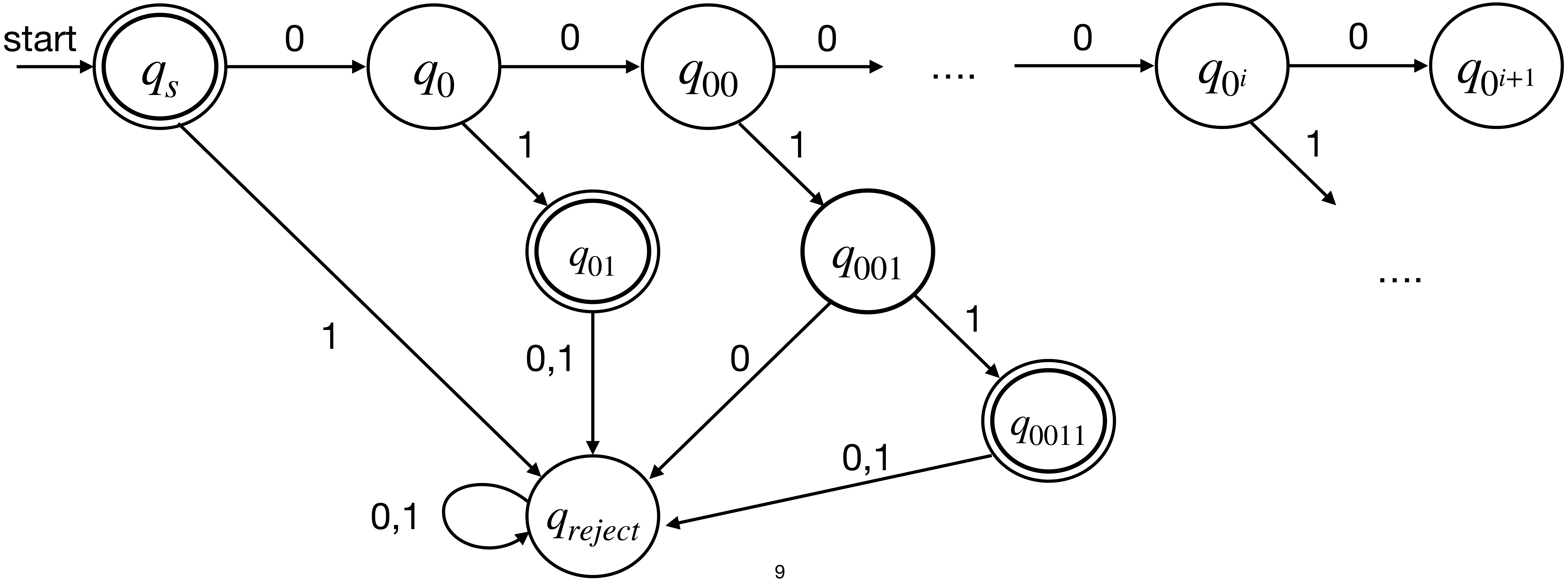
What state should DFA be in after reading the suffix 1111 ?

Proof by contradiction

- Suppose L is regular. Then there is a DFA M which recognizes L .

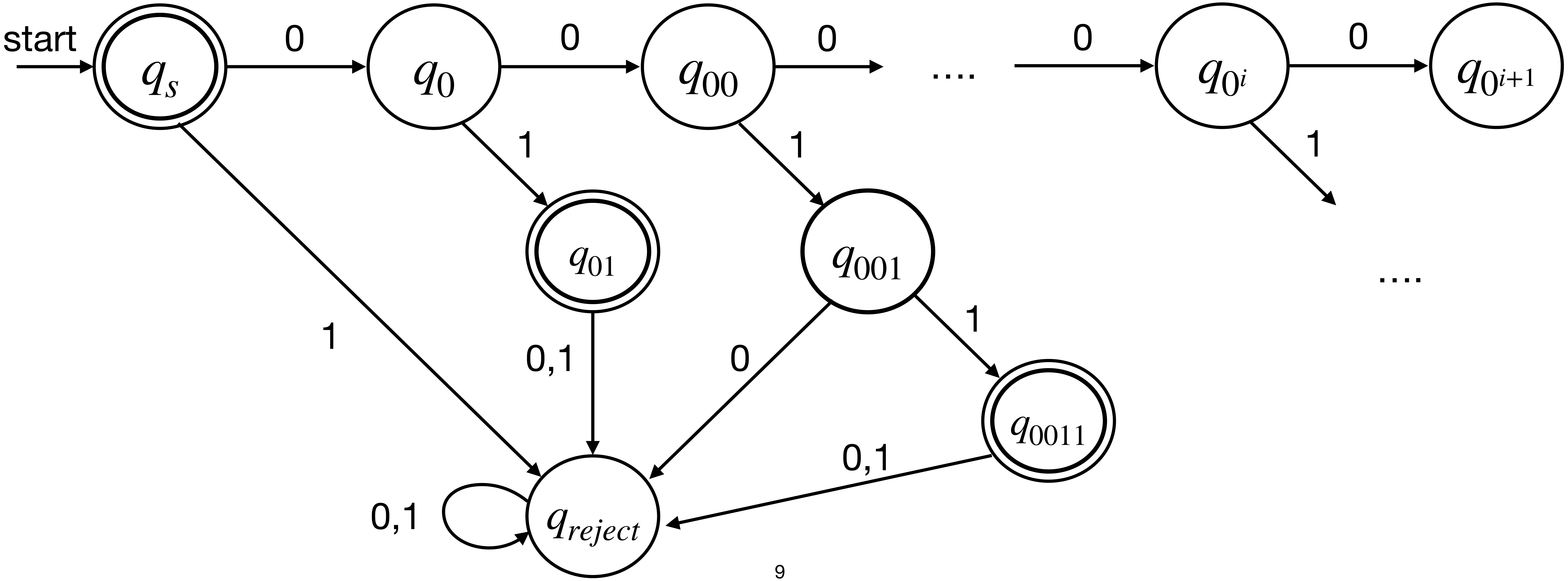
Proof by contradiction

- Suppose L is regular. Then there is a DFA M which recognizes L .



Proof by contradiction

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



Proof by contradiction

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

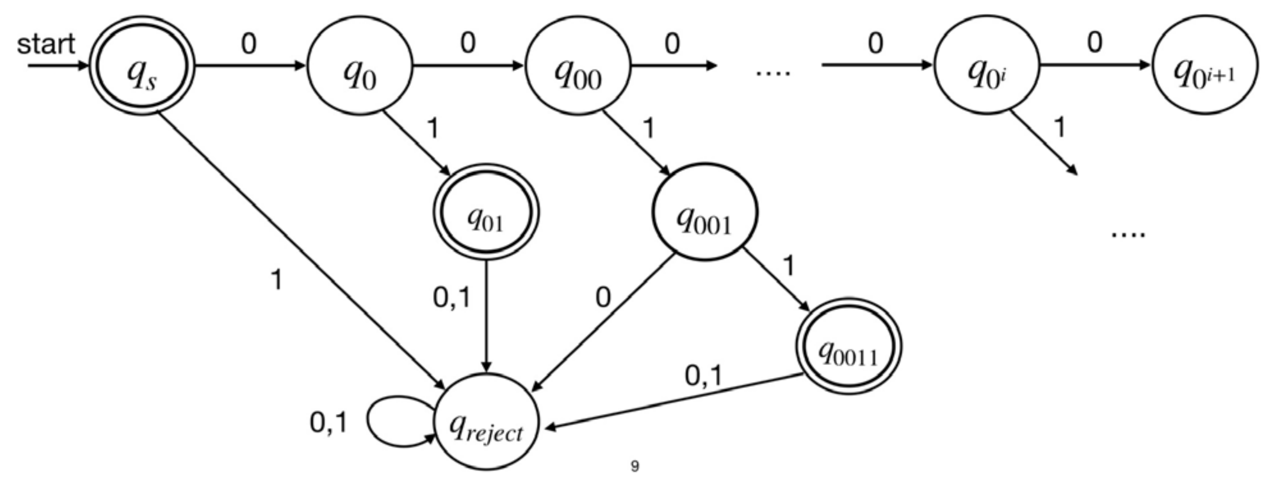
Proof by contradiction

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

$$\epsilon,0,00,000,...0^n$$

for a total of $n + 1$ strings. What states does M reach on the above strings?

Proof by contradiction



- Suppose L is regular. Then there is a DFA M which recognizes L .

- Let $M = (Q, \{0,1\}, \delta, s, A)$) where $|Q| = n$ is finite. Consider strings

$$\epsilon, 0, 00, 000, \ldots 0^n$$

for a total of $n + 1$ strings. What states does M reach on the above strings?

- Let $q_{0^i} = \hat{\delta}(s, 0^i)$. By pigeon-hole principle $q_{0^i} = q_{0^j}$ for some $0 \leq i < j \leq n$.

Proof by contradiction

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

$$\epsilon,0,00,000,...0^n$$

for a total of $n + 1$ strings. What states does M reach on the above strings?

- Let $q_{0^i} = \hat{\delta}(s,0^i)$. By pigeon-hole principle $q_{0^i} = q_{0^j}$ for some $0 \leq i < j \leq n$.
- That is, M is in the same state after reading 0^i and 0^j where $i \neq j$. Then M should accept 0^i1^i but then it will also accept 0^j1^i where $i \neq j$. $\Rightarrow$ M doesn't work for L.

Proof by contradiction

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

$$\epsilon,0,00,000,...0^n$$

for a total of $n + 1$ strings. What states does M reach on the above strings?

- Let $q_{0^i} = \hat{\delta}(s,0^i)$. By pigeon-hole principle $q_{0^i} = q_{0^j}$ for some $0 \leq i < j \leq n$.
- That is, M is in the same state after reading 0^i and 0^j where $i \neq j$. Then 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 is a DFA for L. Thus, there is no DFA for L.

Proving non-regularity: Methods

- Fooling sets: Also called the method of distinguishing suffixes. To prove that L it is non-regular, find an infinite fooling set.

Proving non-regularity: Methods

- Fooling sets: Also called the method of distinguishing suffixes. To prove that L it is non-regular, find an infinite fooling set.
- Closure properties: Use existing non-regular languages and regular languages to prove that some new language is non-regular.

Proving non-regularity: Methods

- Fooling sets: Also called the method of distinguishing suffixes. To prove that L it is non-regular, find an infinite fooling set.
- Closure properties: Use existing non-regular languages and regular languages to prove that some new language is non-regular.
- 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 - there are many different pumping lemmas for different classes of languages.

- Myhill-Nerode Theorem ⟹ requires exhaustive case analysis.

Proving non-regularity: Fooling sets

Fooling set method

Definitions: what is meant by distinguishable?

- Given a DFA M recognizing a language $L(M)$ defined over Σ, we say two **states** $p, q \in Q$ are **equivalent** if, for *all* $w \in \Sigma*$

$$\hat{\delta}\left(p, w\right) \in A \;\Leftrightarrow\; \hat{\delta}\left(q, w\right) \in A$$

Fooling set method

Definitions: what is meant by distinguishable?

q1, q0 are distinguishable because q1 0 ∈ A, q0 0 = q1 ∉ A

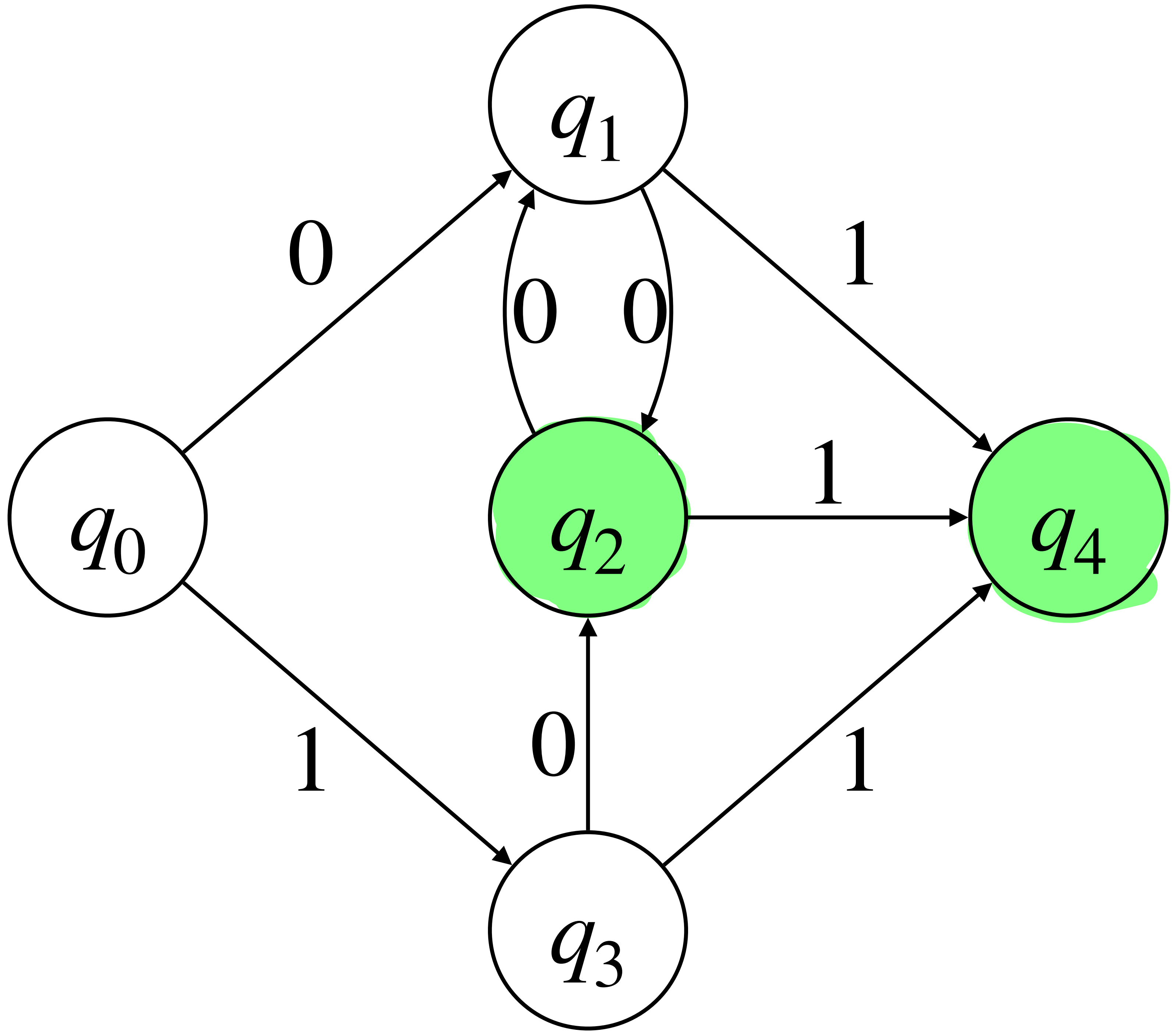
- Given a DFA M recognizing a language $L(M)$ defined over Σ, we say two **states** $p, q \in Q$ are **equivalent** if, for *all* $w \in \Sigma^*$

$$\hat{\delta}\left(p, w\right) \in A \iff \hat{\delta}\left(q, w\right) \in A$$

- We say two states $p, q \in Q$ are **distinguishable** if $\exists w \in \Sigma^*$ such that *exactly one* of $\hat{\delta}(p, w)$ or $\hat{\delta}(q, w)$ is in A.

extended transition functions.

are states.



Source: Kani Archive

Fooling set method

Definitions: what is meant by distinguishable?

- In light of the previous definitions, denote

$$\Omega_w := \hat{\delta}\left(q_0, w\right)$$

Fooling set method

Definitions: what is meant by distinguishable?

- In light of the previous definitions, denote

$$\Omega_w := \hat{\delta}\left(q_0, w\right)$$

- We say two strings $x, y \in \Sigma^*$ are **distinguishable** relative to $L(M)$ if Ω_x and Ω_y are distinguishable.

Fooling set method

Definitions: what is meant by distinguishable?

- In light of the previous definitions, denote

$$\Omega_w := \hat{\delta}\left(q_0, w\right)$$

- We say two strings $x, y \in \Sigma^*$ are **distinguishable** relative to $L(M)$ if Ω_x and Ω_y are distinguishable.

- In other words, two strings $x, y \in \Sigma^*$ are **distinguishable** relative to $L(M)$ if $\exists w \in \Sigma^*$ such that precisely one of xw or yw is in $L(M)$.

either $xw \in L(M)$ and $yw \notin L(M)$
or $xw \notin L(M)$ and $yw \in L(M)$

Fooling sets

F is a set of strings from Σ^* such that they are pairwise distinguishable for L (or M).

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:

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.

$\varepsilon, 0, 00, \ldots, 0^n$

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

Theorem:

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

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.

Formalize our work so far …

We have already saw the essence of the following lemma:

Lemma

Let L be a regular language over Σ and M be a DFA $(Q, \Sigma, \delta, q_0, A)$ such that M recognizes L. If $x, y \in \Sigma^*$ are distinguishable, then $\Omega_x \neq \Omega_y$ where $\Omega_w := \hat{\delta}(q_0, w)$.

Formalize our work so far …

We have already saw the essence of the following lemma:

Lemma

Let L be a regular language over Σ and M be a DFA $(Q, \Sigma, \delta, q_0, A)$ such that M recognizes L. If $x, y \in \Sigma^*$ are distinguishable, then $\Omega_x \neq \Omega_y$ where $\Omega_w := \hat{\delta}(q_0, w)$.

Let use this lemma to prove the theorem on the previous slide.

Proof of 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.

Proof:

Proof of 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.

Proof:

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

Proof of 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.

Proof:

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

$$M = (Q, \Sigma, \delta, q_0, A)$$

Proof of 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.

Proof:

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

$$M = (Q, \Sigma, \delta, q_0, A)$$

be any DFA that accepts L . Also let $q_i = \Omega_{w_i} = \hat{\delta}(q_0, w_i)$. Then by lemma $q_i \neq q_j$ for all $i \neq j$. As such,

Proof of 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.

Proof:

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

$$M = (Q, \Sigma, \delta, q_0, A)$$

be any DFA that accepts L . Also let $q_i = \Omega_{w_i} = \hat{\delta}(q_0, x_i)$. Then by lemma $q_i \neq q_j$ for all $i \neq j$. As such,

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

Proof of 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.

Proof:

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

$$M = (Q, \Sigma, \delta, q_0, A)$$

be any DFA that accepts L . Also let $q_i = \Omega_{w_i} = \hat{\delta}(q_0, x_i)$. Then by lemma $q_i \neq q_j$ for all $i \neq j$. As such,

$$|Q| \geq |\{q_1, \ldots, q_m\}| = |\{w_1, \ldots, w_m\}| = |A| \qquad \blacksquare$$

Infinite Fooling Sets

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

Infinite Fooling Sets

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

Proof by contradiction

Infinite Fooling Sets

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

Proof by contradiction

Let $w_1, w_2, \ldots \subseteq F$ be an infinite sequence of strings that are *pairwise distinguishable* and define $F_k := \{w_1, w_2, \ldots, w_k\}$ for $i \geq 1$.

Infinite Fooling Sets

$F_1 = \{w_1\}$
$F_2 = \{w_1, w_2\}$
$F_3 = \{w_1, w_2, w_3\}$
$\vdots$

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

Proof by contradiction

Let $w_1, w_2, \ldots \subseteq F$ be an infinite sequence of strings that are *pairwise distinguishable* and define $F_k := \{w_1, w_2, \ldots, w_k\}$ for $i \geq 1$.

Assume $\exists\ M = (Q, \Sigma, \delta, q_0, A)$ a DFA for L. Then *by the previous theorem*, $|Q| > |F_k|$ for all k.

Infinite Fooling Sets

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

<u>Proof by contradiction</u>

Let $w_1, w_2, \ldots \subseteq F$ be an infinite sequence of strings that are *pairwise distinguishable* and define $F_k := \{w_1, w_2, \ldots, w_k\}$ for $i \geq 1$.

Assume $\exists\ M = (Q, \Sigma, \delta, q_0, A)$ a DFA for L . Then *by the previous theorem*, $|Q| > |F_k|$ for all k.

But k is not bounded above. As such $|Q|$ *cannot* be bounded above.

Infinite Fooling Sets

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

Proof by contradiction

Let $w_1, w_2, \ldots \subseteq F$ be an infinite sequence of strings that are *pairwise distinguishable* and define $F_k := \{w_1, w_2, \ldots, w_k\}$ for $i \geq 1$.

Assume $\exists\ M = (Q, \Sigma, \delta, q_0, A)$ a DFA for L . Then *by the previous theorem*, $|Q| > |F_k|$ for all k.

But k is not bounded above. As such $|Q|$ *cannot* be bounded above.

Therefore M cannot be a D**F(inite)A** $\Longrightarrow$ contradiction. ∎

Examples

Exercises with fooling sets

Example 1 - $\Sigma = \{0,1\}$

- $L_1 = \{0^n1^n \mid n \geq 0\}$

F is infinite in size

$F = \{0^i \mid i \geq 0\}$, is a fooling set.

0^i and 0^j should be pairwise distinguishable

$0^i1^i \in L$ $\quad 0^j1^i \notin L$, $j \neq i$

Exercises with fooling sets

Example 2 - $\Sigma = \{0,1\}$

- $L_2 = \{w \in \Sigma^* \mid \#_0(w) = \#_1(w)\}$

$F = \{0^i \mid i \geq 0\}$.

Show that this works

(have to finish argument precisely)

Exercises with fooling sets

Example 3 - $\Sigma = \{0,1\}$

- $L_3 = \{w \in \Sigma^* \mid w = \text{rev}(w)\}$

$F = \{ 0^i \mid i \geq 0 \}$.

What is a distinguishing suffix for a pair in F?

$\Rightarrow \exists x$ such that $0^i x \in L$ and $0^j x \notin L$.

set $x = 10^i$ $i \neq j$.

Proving non-regularity: Closure properties

Closure properties & non-regularity

Thought exercise

- We know that *regular* languages are **closed** under concatenation, union and Kleene star.

Closure properties & non-regularity

Thought exercise

- We know that *regular* languages are **closed** under concatenation, union and Kleene star.
  - **Fact:** They are also closed under complementation and intersection.

Closure properties & non-regularity

Thought exercise

- We know that *regular* languages are **closed** under concatenation, union and Kleene star.
  - **Fact:** They are also closed under complementation and intersection.
- Suppose:

unknown

regular language

$$L_n = L_u \square L_r \quad \text{where} \quad \square \in \{ \cap , \cup , \circ \} \textbf{ or}$$

non regular

Closure properties & non-regularity

Thought exercise

- We know that *regular* languages are **closed** under concatenation, union and Kleene star.
  - **Fact:** They are also closed under complementation and intersection.
- Suppose:

$$L_n = L_u \Box L_r \quad \text{where} \quad \Box \in \{\cap, \cup, \circ\} \text{ or}$$

$$L_n = \widetilde{L_u} \quad \text{where} \quad \widetilde{()} \in \{()^*, \overline{()}\}$$

Closure properties & non-regularity

Thought exercise

- We know that *regular* languages are **closed** under concatenation, union and Kleene star.
  - **Fact:** They are also closed under complementation and intersection.
- Suppose:

$$L_n = L_u \,\square\, L_r \quad \text{where} \quad \square \in \{\, \cap \,,\, \cup \,,\, \circ \,\} \textbf{ or}$$

$$L_n = \widetilde{L_u} \quad \text{where} \quad \widetilde{()} \in \{()^*, \overline{()}\}$$

- What can we say about L_u?

Closure properties & non-regularity

Example 1

- Recall

$$L_1 = \{0^n 1^n \mid n \geq 0\} \text{ and } L_2 = \{w \in \Sigma^* \mid \#_0(w) = \#_1(w)\}$$

Closure properties & non-regularity

Example 1

- Recall

canonical example

$$L_1 = \{0^n 1^n \mid n \geq 0\} \text{ and } L_2 = \{w \in \Sigma^* \mid \#_0(w) = \#_1(w)\}$$

- By now we know L_1 is non-regular. What about L_2?

$L_1 = L_2 \cap \{0^* 1^*\}$

non regular

must be non regular.

regular language.

Closure properties & non-regularity

Example 1

- Recall

$$L_1 = \{0^n1^n \mid n \geq 0\} \text{ and } L_2 = \{w \in \Sigma^* \mid \#_0(w) = \#_1(w)\}$$

- By now we know L_1 is non-regular. What about L_2?
- Which set is larger? Can we get L_1 from L_2 using a regular operation?

Prove by contradiction

Closure properties & non-regularity

Example 2

$L_1 = \{a^m b^n \mid m = n\}$ — Note $\overline{L_3} \neq L_1$

↳ has order → a before n.

↳ includes b before a as well.

- Let

$$L_3 := \{a^m b^n \mid m \geq 0, n \geq 0, m \neq n\}$$

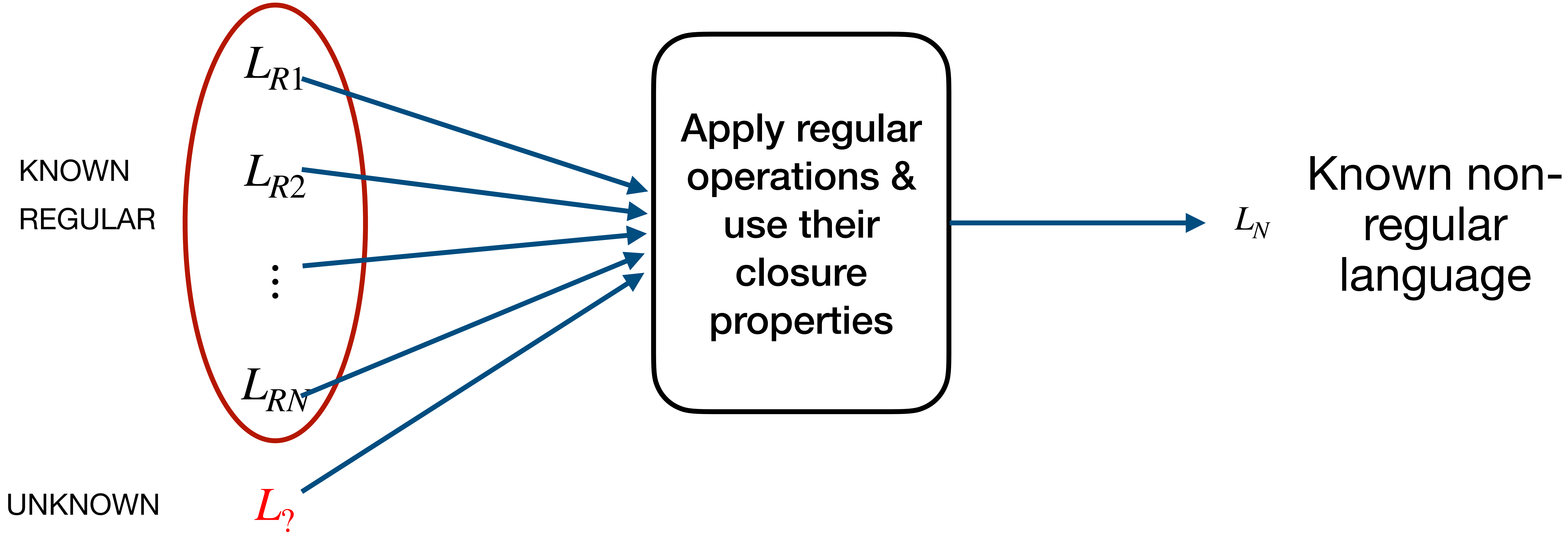
Suppose L_3 is regular. Then $\overline{L_3}$ is regular.

$$L_1 = \overline{L_3} \cap \{a^* b^*\}$$

↳ leads to contradiction

Closure properties & non-regularity

General recipe



Myhill-Nerode Theorem

Towards the statement

- Recall that two strings x, y are *distinguishable relative to* $L = L(M)$ provided there exists a *distinguishing suffix* $w \in \Sigma^*$ where the DFA M recognizes L and Σ is the alphabet of M.

Myhill-Nerode Theorem

Towards the statement

mathematicians like to be precise, distinguishability is always with respect to a M/L

- Recall that two strings x, y are *distinguishable* ~~*relative*~~ *to* $L = L(M)$ provided there exists a *distinguishing suffix* $w \in \Sigma^*$ where the DFA M recognizes L and Σ is the alphabet of M.

okay to read as if would not there.

- Define x, y to be *equivalent* ~~*relative*~~ *to* L (denoted $x \sim_L y$) if there is no distinguishing suffix for x and y. In other words, $x \sim_L y$ means that

Myhill-Nerode Theorem

Towards the statement

- Recall that two strings x, y are *distinguishable relative to* $L = L(M)$ provided there exists a *distinguishing suffix* $w \in \Sigma^*$ where the DFA M recognizes L and Σ is the alphabet of M.

- Define x, y to be *equivalent relative to* L (denoted $x \sim_L y$) if there is no distinguishing suffix for x and y. In other words, $x \sim_L y$ means that

$$\forall w \in \Sigma^* : xw \in L \iff yw \in L$$

Myhill-Nerode Theorem

Towards the statement

easily digestible

- Recall that two strings x, y are *distinguishable relative* to $L = L(M)$ provided there exists a *distinguishing suffix* $w \in \Sigma^*$ where the DFA M recognizes L and Σ is the alphabet of M.

- Define x, y to be *equivalent relative* to L (denoted $x \sim_L y$) if there is no distinguishing suffix for x and y. In other words, $x \sim_L y$ means that

$$\forall w \in \Sigma^* : xw \in L \iff yw \in L$$

- Then $\sim_L$ partitions $L = L(M)$ into *equivalence classes*.

have to unpack this.

Myhill-Nerode Theorem

Quick review - definitions

- What is an equivalence class?

Myhill-Nerode Theorem

Quick review - definitions

- What is an equivalence class?
  - Let $\sim$ be an ***equivalence relation*** on a nonempty set A. For each $a \in A$, the equivalence class $[a]$ of a is the subset of A consisting of all elements that are equivalent to a

$$[a] := \{x \in A \ \mid x \sim a\}$$

Myhill-Nerode Theorem

Quick review - definitions

- What is an equivalence class?
  - Let $\sim$ be an ***equivalence relation*** on a nonempty set A. For each $a \in A$, the equivalence class $[a]$ of a is the subset of A consisting of all elements that are equivalent to a

$$[a] := \left\{x \in A \ \mid x \sim a\right\}$$

- What is an equivalence relation?

Myhill-Nerode Theorem

Quick review - definitions

- What is an equivalence class?
  - Let $\sim$ be an ***equivalence relation*** on a nonempty set A. For each $a \in A$, the equivalence class $[a]$ of a is the subset of A consisting of all elements that are equivalent to a

$$[a] := \{x \in A \mid x \sim a\}$$

$x \sim x$ $\quad$ $x \sim y \Rightarrow y \sim x$

- What is an equivalence relation?
  - An ***equivalence relation*** is a binary relation that is reflexive, symmetric & transitive.

$a \sim b$ and $b \sim c \Rightarrow a \sim c$

Myhill-Nerode Theorem

Quick review - definitions

- Recall that given sets X and Y,

$$X \times Y := \left\{ (x, y) \ \mid x \in X, y \in Y \right\}$$

Myhill-Nerode Theorem

Quick review - definitions

- Recall that given sets X and Y,

$$X \times Y := \left\{(x, y) \ \mid x \in X, y \in Y\right\}$$

- A ***binary relation*** *over* sets X and Y is a subset of $X \times Y$. A binary relation *on* X is a subset of $X \times X$.

Myhill-Nerode Theorem

Quick review - definitions

- Recall that given sets X and Y,

$$X \times Y := \{(x, y) \mid x \in X, y \in Y\}$$

- A ***binary relation*** *over* sets X and Y is a subset of $X \times Y$. A binary relation *on* X is a subset of $X \times X$.

- An ***equivalence relation*** on X is a binary relation that is reflexive, symmetric & transitive.

Myhill-Nerode Theorem

Quick review - definitions

Example 1: Modulo arithmetic

- Recall that given sets X and Y,

$$X \times Y := \{(x, y) \mid x \in X, y \in Y\}$$

- A ***binary relation*** *over* sets X and Y is a subset of $X \times Y$. A binary relation *on* X is a subset of $X \times X$.

- An ***equivalence relation*** on X is a binary relation that is reflexive, symmetric & transitive.

Myhill-Nerode Theorem

Quick review - definitions

- Recall that given sets X and Y,

$$X \times Y := \left\{ (x, y) \ \mid x \in X, y \in Y \right\}$$

- A ***binary relation*** *over* sets X and Y is a subset of $X \times Y$. A binary relation *on* X is a subset of $X \times X$.

- An ***equivalence relation*** on X is a binary relation that is reflexive, symmetric & transitive.

Example 1: Modulo arithmetic

We denote by $\mathbb{Z}_n$ (for positive n) the integers modulo n.

Myhill-Nerode Theorem

Quick review - definitions

$2 \equiv_3 5$

$5 \equiv_3 8$

- Recall that given sets X and Y,

$$X \times Y := \left\{(x, y) \ \mid x \in X, y \in Y\right\}$$

- A ***binary relation*** *over* sets X and Y is a subset of $X \times Y$. A binary relation *on* X is a subset of $X \times X$.

- An ***equivalence relation*** on X is a binary relation that is reflexive, symmetric & transitive.

Example 1: Modulo arithmetic

We denote by $\mathbb{Z}_n$ (for positive n) the integers modulo n.

Thus in $\mathbb{Z}_3$, we have $1 \equiv_3 4$, $4 \equiv_3 7$, and so on.

Myhill-Nerode Theorem

Quick review - definitions

- Recall that given sets X and Y,

$$X \times Y := \{(x, y) \mid x \in X, y \in Y\}$$

- A ***binary relation*** *over* sets X and Y is a subset of $X \times Y$. A binary relation *on* X is a subset of $X \times X$.

- An ***equivalence relation*** on X is a binary relation that is reflexive, symmetric & transitive.

reflexive $1 \equiv_3 1$

symmetric $1 \equiv_3 4 \Rightarrow 4 \equiv_3 1$

Example 1: Modulo arithmetic

We denote by $\mathbb{Z}_n$ (for positive n) the integers modulo n.

Thus in $\mathbb{Z}_3$, we have $1 \equiv_3 4$, $4 \equiv_3 7$, and so on.

Then $\equiv_3$ is an equivalence relation.

transitive

$1 \equiv_3 4,\ 4 \equiv_3 7 \Rightarrow 1 \equiv_3 7$

Myhill-Nerode Theorem

Quick review - definitions

- Recall that given sets X and Y,

$$X \times Y := \left\{(x, y) \;\mid x \in X, y \in Y\right\}$$

- A ***binary relation*** *over* sets X and Y is a subset of $X \times Y$. A binary relation *on* X is a subset of $X \times X$.

- An ***equivalence relation*** on X is a binary relation that is reflexive, symmetric & transitive.

Example 2:

$$X = \{a, b, c\}$$

$$R = \left\{ \begin{array}{l} (a, a), \\ (b, b), \\ (c, c), \\ (b, c), \\ (c, b) \end{array} \right\} \subseteq X \times X$$

Myhill-Nerode Theorem

Necessary and sufficient condition for regularity

- If two strings $x \sim_L y$ then x is indistinguishable from y in L. The equivalence relation $\sim_L$ partitions $L(M)$ into equivalence classes.

"Divide into non-Intersecting subsets such that their union comprises the whole"

Myhill-Nerode Theorem

Necessary and sufficient condition for regularity

- If two strings $x \sim_L y$ then x is indistinguishable from y in L. The equivalence relation $\sim_L$ partitions $L(M)$ into equivalence classes.

A language $L = L(M)$ is regular if and only if $\sim_L$ has a finite number of equivalence classes. Furthermore, this number is equal to the number of states in the minimal DFA M accepting L.

Myhill-Nerode Theorem

Necessary and sufficient condition for regularity

- If two strings $x \sim_L y$ then x is indistinguishable from y in L. The equivalence relation $\sim_L$ partitions $L(M)$ into equivalence classes.

A language $L = L(M)$ is regular if and only if $\sim_L$ has a finite number of equivalence classes. Furthermore, this number is equal to the number of states in the minimal DFA M accepting L .

Example: Let L be the set of binary strings divisible by 3. Show that L is regular.

Myhill-Nerode Theorem

Example

Let L be the set of binary strings divisible by 3. Show that L is regular.

Hint: A binary string is divisible by 3 if the **sum of the odd bits** equal the **sum of the even bits**.

Myhill-Nerode Theorem

Example

Let L be the set of binary strings divisible by 3. Show that L is regular.

Hint: A binary string is divisible by 3 if the **sum of the odd bits** equal the **sum of the even bits**.

- ε and 0 are indistinguishable: Both $\varepsilon w, 0w \in L$ **or** $\varepsilon w, 0w \notin L$ for all w.

Myhill-Nerode Theorem

Example

Let L be the set of binary strings divisible by 3. Show that L is regular.

Hint: A binary string is divisible by 3 if the **sum of the odd bits** equal the **sum of the even bits**.

- ε and 0 are indistinguishable: Both $\varepsilon w, 0w \in L$ **or** $\varepsilon w, 0w \notin L$ for all w.
    - By the same argument 11 is indistinguishable from $\varepsilon, 0$.

Myhill-Nerode Theorem

Example

Let L be the set of binary strings divisible by 3. Show that L is regular.

Hint: A binary string is divisible by 3 if the **sum of the odd bits** equal the **sum of the even bits.**

- ε and 0 are indistinguishable: Both $\varepsilon w, 0w \in L$ **or** $\varepsilon w, 0w \notin L$ for all w.
  - By the same argument 11 is indistinguishable from $\varepsilon, 0$.
  - Thus $[0] = \{\varepsilon, 0, 11, 110, 1001, 1100, 1111, \ldots\}$ → equivalence class of numbers exactly divisible by three.

Myhill-Nerode Theorem
Example

Let L be the set of binary strings divisible by 3. Show that L is regular.

Hint: A binary string is divisible by 3 if the **sum of the odd bits** equal the **sum of the even bits.**

Myhill-Nerode Theorem

Example

Let L be the set of binary strings divisible by 3. Show that L is regular.

Hint: A binary string is divisible by 3 if the **sum of the odd bits** equal the **sum of the even bits.**

- 1 is distinguishable from $[0]$ since for any $x \in [0]$ we have $x \cdot 1 \notin L$ but $1 \cdot 1 \in L$.

Myhill-Nerode Theorem

Example

Let L be the set of binary strings divisible by 3. Show that L is regular.

Hint: A binary string is divisible by 3 if the **sum of the odd bits** equal the **sum of the even bits.**

- 1 is distinguishable from $[0]$ since for any $x \in [0]$ we have $x \cdot 1 \notin L$ but $1 \cdot 1 \in L$.
  - Same holds true for 100 — why?

Myhill-Nerode Theorem

Example

Let L be the set of binary strings divisible by 3. Show that L is regular.

Hint: A binary string is divisible by 3 if the **sum of the odd bits** equal the **sum of the even bits.**

- 1 is distinguishable from $[0]$ since for any $x \in [0]$ we have $x \cdot 1 \notin L$ but $1 \cdot 1 \in L$.
  - Same holds true for 100 — why?
  - Thus $[1] = \{1,100,111,1010,\ldots\}$

equivalence class of numbers leaving remainder one on division by 3.

Myhill-Nerode Theorem

Example

Let L be the set of binary strings divisible by 3. Show that L is regular.

Hint: A binary string is divisible by 3 if the **sum of the odd bits** equal the **sum of the even bits.**

Myhill-Nerode Theorem

Example

Let L be the set of binary strings divisible by 3. Show that L is regular.

Hint: A binary string is divisible by 3 if the **sum of the odd bits** equal the **sum of the even bits.**

- 10 is distinguishable from $[0]$ *and* $[1]$. For any $x \in [0]$ we have $x \cdot 0 \in L$ but $10 \cdot 0 \notin L$. For any $y \in [1]$ we have $y \cdot 1 \in L$ but $10 \cdot 1 \notin L$.

Myhill-Nerode Theorem

Example

Let L be the set of binary strings divisible by 3. Show that L is regular.

Hint: A binary string is divisible by 3 if the **sum of the odd bits** equal the **sum of the even bits.**

- 10 is distinguishable from $[0]$ *and* $[1]$. For any $x \in [0]$ we have $x \cdot 0 \in L$ but $10 \cdot 0 \notin L$. For any $y \in [1]$ we have $y \cdot 1 \in L$ but $10 \cdot 1 \notin L$.
  - Same holds true for 101 — why?

Myhill-Nerode Theorem

Example

Let L be the set of binary strings divisible by 3. Show that L is regular.

Hint: A binary string is divisible by 3 if the **sum of the odd bits** equal the **sum of the even bits.**

- 10 is distinguishable from $[0]$ *and* $[1]$. For any $x \in [0]$ we have $x \cdot 0 \in L$ but $10 \cdot 0 \notin L$. For any $y \in [1]$ we have $y \cdot 1 \in L$ but $10 \cdot 1 \notin L$.
  - Same holds true for 101 — why?
  - Thus $[10] = \{10, 101, \ldots\}$

Equivalence class of numbers leaving remainder 2 on division by 3.

Myhill-Nerode Theorem

Example

Let L be the set of binary strings divisible by 3. Show that L is regular.

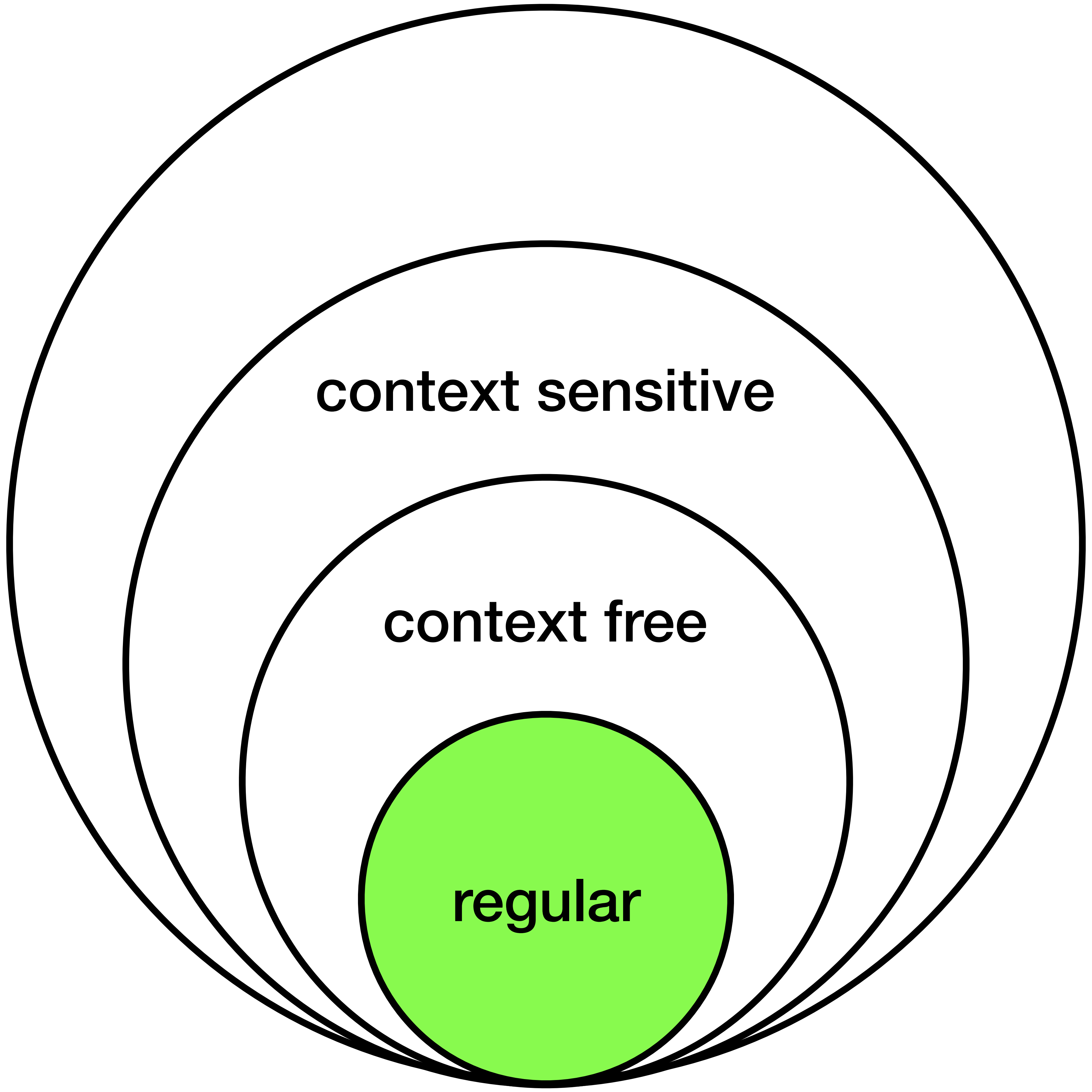
Hint: A binary string is divisible by 3 if the **sum of the odd bits** equal the **sum of the even bits.**

- 10 is distinguishable from $[0]$ *and* $[1]$. For any $x \in [0]$ we have $x \cdot 0 \in L$ but $10 \cdot 0 \notin L$. For any $y \in [1]$ we have $y \cdot 1 \in L$ but $10 \cdot 1 \notin L$.
  - Same holds true for 101 — why?
  - Thus $[10] = \{10, 101, \ldots\}$
  - $[0], [1], [10]$ form a partition of Σ^* under $\sim_L$. **Thus L is regular.**

There are no more classes to consider!! Remainder is 0, 1 or 2.

Next time

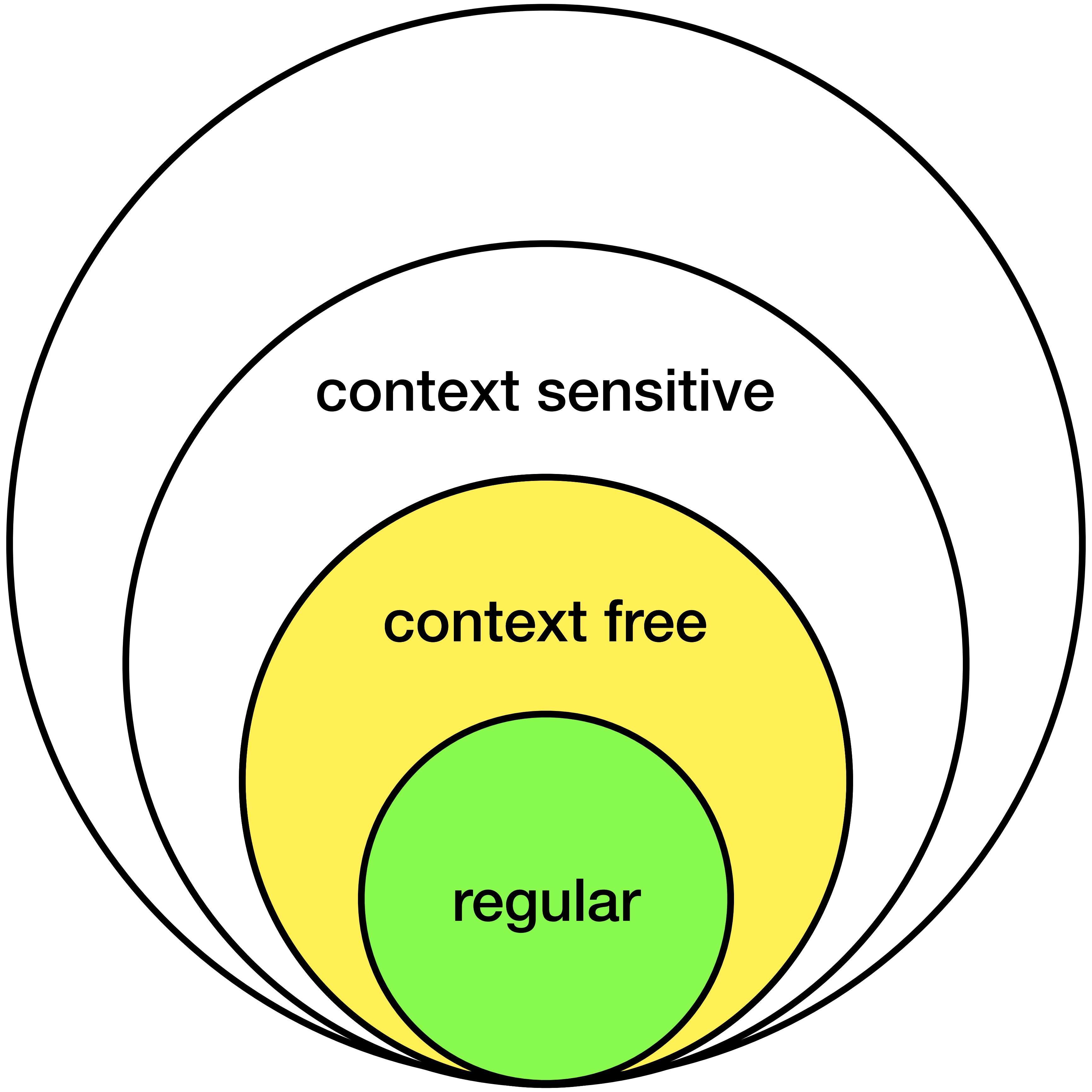
- This lecture was about some tools for recognizing non-regular lanaguages
- Next week we will see the equivalent of DFAs for *context-free* languages.
  - Called ***Pushdown Automata***
  - Context sensitive languages & Linear Bounded Automata (LBAs) will not be covered
    - See Sipser's book



Source: Kani Archive

Next time

- This lecture was about some tools for recognizing non-regular lanaguages
- Next week we will see the equivalent of DFAs for *context-free* languages.
  - Called ***Pushdown Automata***
  - Context sensitive languages & Linear Bounded Automata (LBAs) will not be covered
    - See Sipser's book



Source: Kani Archive