Non-regular Languages

Arjun Chandrasekhar

Regular languages

Show that the following language is regular

$$\{0^n1^n | n \in \mathbb{N}\} = \{\epsilon, 01, 0011, 000111, \ldots\}$$

- You may construct a DFA, NFA, or a regex to recognize the language

Non-regular languages

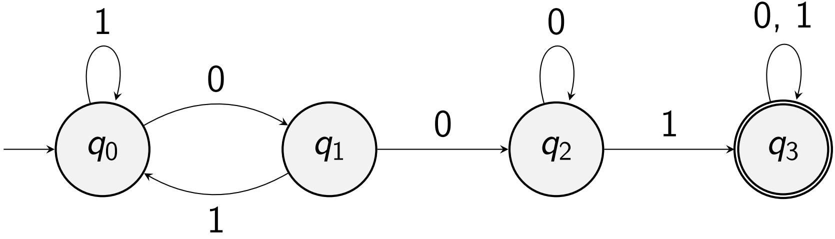
- Are DFAs an all-purpose computing device?
  Can DFAs be used to recognize *every* language?
- How do we show that a particular langauge cannot be recognized by *any* DFA?
  - We certainly can't exhaustively check every possible DFA...

Non-regul

- Are D device?
  Can D language?
- How ngauge
  canno
  - W every
    p

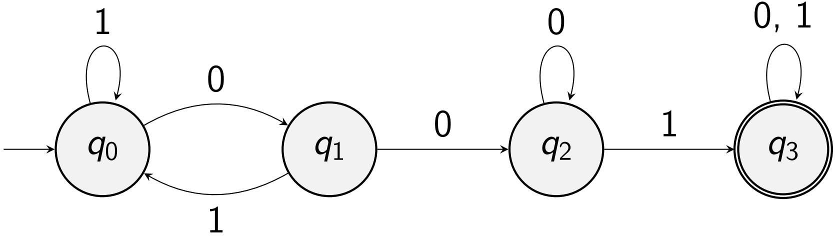
Repeated states

How many symbols can we read before we are forced to repeat a state?



Repeated states

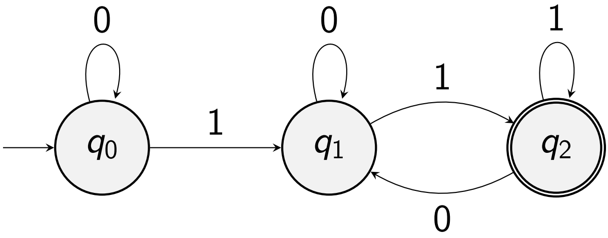
How many symbols can we read before we are forced to repeat a state?



4 symbols

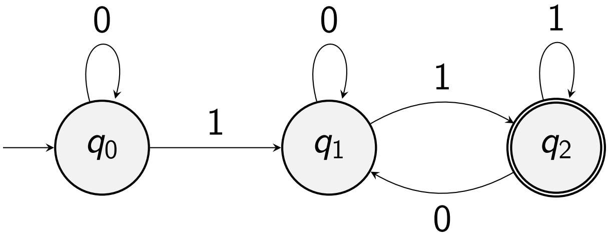
Repeated states

How many symbols can we read before we are forced to repeat a state?



Repeated states

How many symbols can we read before we are forced to repeat a state?



3 symbols

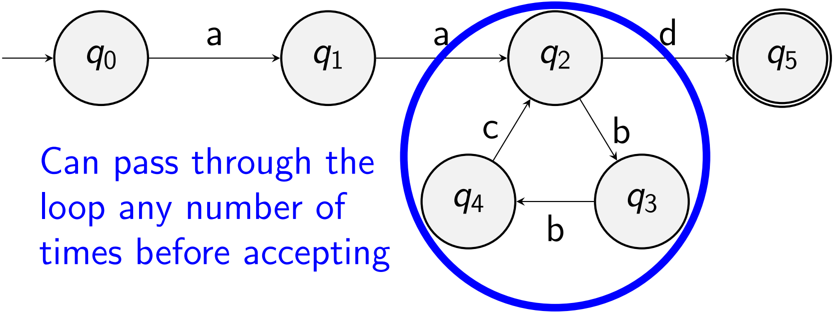
Repeated states

How many characters can we read before we are forced to repeat a state?

- Let D be a DFA with n states
- Let w be a string with at least n characters
- **Proposition:** When we read w, we will repeat a state
- **Proof:**
  - We visit starting state without reading a character
  - After reading n characters, we visit n more states
  - n states in the DFA, $n + 1$ total states visited
  - At least one visited state is a repeat (pigeonhole principle)

Pumping lemma illustration

What strings are accepted by the following DFA?
(Ignore missing transitions)



Can pass through the loop any number of times before accepting

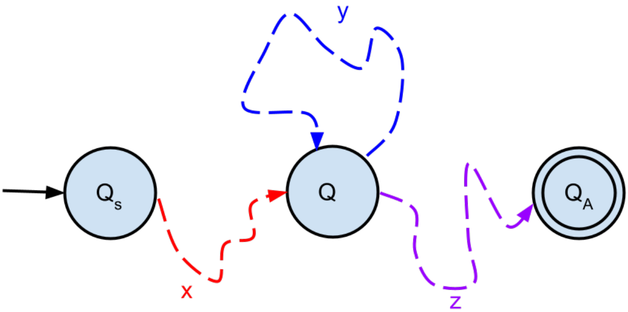
- aad
- $aabbcbbcd$
- $aabbcd$
- $aa\underbrace{bbcbbc\ldots bbc}_{(bbc)^n}d$

The Pumping Lemma

- Let $D = (Q, \Sigma, \delta, q_s, \epsilon)$ be a DFA that recognizes language L
- Given a string $s \in L$:
  - If $|s| >= |Q|$, then when D processes s, it must visit one (or more) state(s) more than once
- Let's consider the implications of one state being visited twice.

Pumping Lemma Idea

What strings are accepted by this DFA?



- xz
- xyz
- $xyyz$
- $x\underbrace{y\ldots y}_{y^n}z$

The Pumping Lemma (informal statement

- Let L be a regular language.
- Let $s \in L$ be a string that is "sufficiently long"
- Can split s into three parts $s = xyz$ such that:
  - The prefix xy is not "too long"
  - The middle part y is not empty
  - We can add (or remove) any number of copies of y, and the new string will still be in the language

The Pumping Lemma

Lemma: Let L be a regular language. There exists a pumping length p such that for all $w \in L$, if $|w| \geq p$, then w may be divided up into three parts $w = xyz$ such that:

1. $|xy| \leq p$
2. $|y| > 0$
3. $xy^iz \in L$ for all $i \in \mathbb{N}$

The Pumping Lemma

Proof idea:

- If L is regular, it is recognized by a DFA.
- If we take a sufficiently long string in the language, the DFA will visit the same state twice (thus forming a loop) before accepting.
- In principle, we don't need to go around the loop just once; we could 'pump' around the loop any number of times (or not at all) before going to the accept state.
- Thus, the "middle" (i.e. loop) part of the string can be pumped to make other strings that are accepted

The Pumping Lemma

Proof:

- Let M be a DFA recognizing L. Let $p = |Q|$ i.e. number of states
- Let $s = s_1 s_2 ... s_n \in L$ have length $n \geq p$
- Let $r_1 r_2 \ldots r_{n+1}$ be the states visited when processing s. Note that $n + 1 \geq p + 1$
- In the first $p + 1$ states, there must be a repeated state $r_j = r_l$ with $j < l \leq p + 1$ (pigeonhole principle)
- Let $x = s_1 \ldots s_{j-1}, y = s_j \ldots s_{l-1}, z = s_l \ldots s_n$. Then $|y| > 0$, $|xy| \leq p$, and $xy^i z$ will be accepted for all i

Pumping Lemma and Finite Languages

- All finite languages are regular.
- But the pumping lemma says every string in a regular language should be *infinitely* pumpable.
- Is this a contradiction?

Pumping Lemma and Finite Languages

- The pumping lemma states that every string *of length* $\geq p$ should be infinitely pumpable.
  - We are not on the hook for any strings of length $< p$
- For any finite language L, let N be the length of the longest string. Pick $p = N + 1$
  - L does not contain a string of length $\geq p$ that is not pumpable.
  - Thus the pumping lemma is satisfied.

Using the Pumping Lemma

Why is the pumping lemma useful?

- The pumping lemma says that if L is regular, then L has a pumping length p
- **The contrapositive is that if L does not have a valid pumping length, it cannot possibly regular.**

Pumping Lemma Example

Proposition: Let $L = \{0^n 1^n | n \geq 0\}$. L is not a regular language.

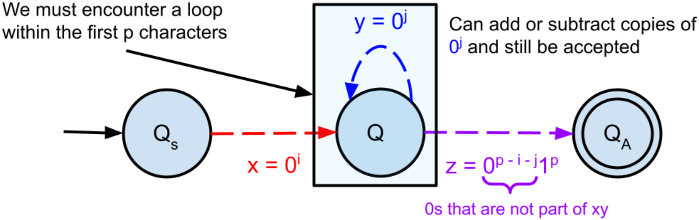
- AFSOC L is regular with pumping length p
- Take $w = 0^p 1^p = \underbrace{0...0}_{p}\underbrace{1...1}_{p} \in L$.
  - Clearly $|w| \geq p$, so it *should* be pumpable
- Let $w = xyz$ with $|xy| \leq p$ and $|y| > 0$
  - $w = \underbrace{0..0}_{x}\underbrace{00...0}_{y}\underbrace{00...1...1}_{z}$
  - y is not empty, xy only contains 0s
- If we pump y, the 0s and 1s won't be equal
- $w \in L$ is not pumpable, which contradicts the pumping lemma. We conclude L is not regular

The Pumping Lemma Example

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

AFSOC L can be decided by a DFA with p states

$s = 0^p 1^p = \underbrace{0 \ldots 0}_{p}\underbrace{1 \ldots 1}_{p} \in L$



Any DFA that accepts $0^p 1^p$ would accept $0^i 0^j 0^j 0^{p-i-j} 1^p = 0^{p+j} 1^p \notin L$

The Pumping Lemma as a 2-player game

Another way to interpret the pumping lemma is as a two player game

1. Player 1 claims L is regular, and declares a pumping length p
2. Player 2 picks a string $w \in L$ with length at least p
3. Player 1 splits up $w = xyz$ such that $|xy| \leq p$ and $|y| > 0$
4. Player 2 tries to create a string $w' \notin L$ by pumping y up or down
5. If Player 2 wins if s/he can pump to create a string $w' \notin L$. Otherwise, Player 1 wins

The Pumping Lemma as a 2-player game

Another way to interpret the pumping lemma is as a two player game

- If L is regular, there exists a pumping length p such that Player 1 can always win
  - There is always a valid way to split up *any* string of length $\geq p$ such that pumping won't cause a problem
- If L is not regular, Player 2 can win for *every* possible p
  - For any p, there exists a string $w \in L$ with length $\geq p$ that cannot be pumped no matter how it is split up

The pumping lemma

Proposition: Let $L = \{0^n 1^n | n \geq 0\}$. L is not a regular language.

1. Player 1 claims L is regular and declares a pumping length p
2. Player 2 picks $w = 0^p 1^p$
3. Player 1 splits up $w = xyz$ according to the pumping lemma rules
4. No matter how Player 1 splits up w, Player 2 can pump up or down to make the 0s and 1s unequal
5. Player 2 will win for every p, so we conclude L is not regular.

Pumping Lemma Example

Let $\Sigma = \{0, 1\}$. Let's prove that L $=$ $\{ww^R | w \in \Sigma^*\}$ (i.e. a string followed by the reverse of that string) is not regular

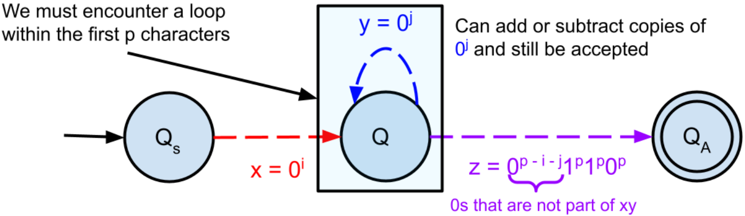
- AFSOC L is regular with pumping length p
- Let $w = 0^p 1^p 1^p 0^p = \underbrace{0...0}_{p}\underbrace{1...1}_{p}\underbrace{1...1}_{p}\underbrace{0...0}_{p} \in L$
- Split up $w = xyz$ such that $|y| > 0$ and $|xy| \leq p$
  - $w = \underbrace{0...0}_{x}\underbrace{0...0}_{y}\underbrace{0..01...11...10...0}_{z}$
- If we pump y, the leading 0s won't equal the trailing 0s
- $w \in L$ is not pumpable; conclude L is not regular

Pumping Lemma Example

$L = \{ww^R \mid n \geq 0\}$

AFSOC L can be decided by a DFA with p states

$$s = 0^p1^p1^p0^p = \overbrace{\underbrace{0\ldots0}_{p}\underbrace{1\ldots1}_{p}}^{w}\overbrace{\underbrace{1\ldots1}_{p}\underbrace{0\ldots0}_{p}}^{w^R} \in L$$



Any DFA that accepts $0^p1^p1^p0^p$ would accept

$$0^i0^j0^j0^{p-i-j}1^p1^p0^p = 0^{p+j}1^p1^p0^p \notin L$$

Pumping Lemma Example

Let $\Sigma = \{a, b\}$. Let's prove that $L = \{a^i b^j | i \geq j\}$ is not regular

- AFSOC L is regular with pumping length p
- Let $w = a^p b^p = \underbrace{a...a}_{p}\underbrace{b...b}_{p} \in L$
- Split up $w = xyz$ such that $|y| > 0$ and $|xy| \leq p$
  - $w = \underbrace{a...a}_{x}\underbrace{a...a}_{y}\underbrace{a..ab...b}_{z}$
  - y is not empty, xy contains all a's
- If we pump up, we...still have more a's than b's
  - $xy^2z = \underbrace{a...a}_{x}\underbrace{a...aa...a}_{y^2}\underbrace{a..ab...b}_{z}$

Pumpi

Let Σ … $j\}$ is not re

- … p
-
-
- … n b's

Pumping Lemma Example

Let $\Sigma = \{a, b\}$. Let's prove that $L = \{a^i b^j | i \geq j\}$ is not regular

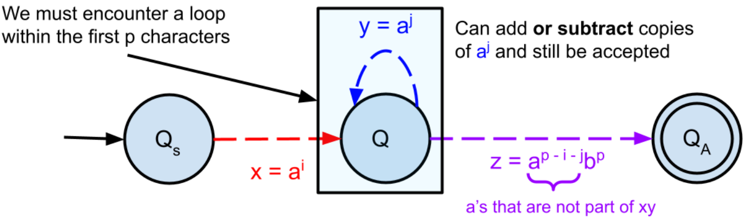
- AFSOC L is regular with pumping length p
- Let $w = a^p b^p = \underbrace{a\ldots a}_{p}\underbrace{b\ldots b}_{p} \in L$
- Split up $w = xyz$ such that $|y| > 0$ and $|xy| \leq p$
  - $w = \underbrace{a\ldots a}_{x}\underbrace{a\ldots a}_{y}\underbrace{a..ab\ldots b}_{z}$
  - y is not empty, xy contains all a's
- If we pump down, we have fewer a's than b's
  - $xy^0z = \underbrace{a\ldots a}_{x}\underbrace{a..ab\ldots b}_{z}$
- $w \in L$ is not pumpable, conclude L is not regular

Pumping Lemma Example

$L = \{a^ib^j \mid i \geq j\}$

AFSOC L can be decided by a DFA with p states

$$s = a^pb^p = \underbrace{a\ldots a}_{p}\underbrace{b\ldots b}_{p} \in L$$



Any DFA that accepts a^pb^p would accept

$a^ia^ja^{p-i-j}b^p = a^ia^{p-i-j}b^p = a^{p-j}b^p \notin L$

Pumping Lemma Example

Let $\Sigma = \{0, 1, +, =\}$. Consider the following language:

$\mathrm{ADD} = \{$i+j=k | i, j, k are binary numbers, equation is valid$\}$

Which of the following strings are in the language?

A) 1100+11=1111

B) 1001+0000=0110

C) 1111+0=1111

D) 001=111+111=000

Pumping Lemma Example

Let $\Sigma = \{0, 1, +, =\}$. Consider the following language:

$\mathrm{ADD} = \{$i+j=k $|$ i, j, k are binary numbers, equation is valid$\}$

Which of the following strings are in the language?

A) 1100+11=1111 ✓

B) 1001+0000=0110
Numbers don't add up

C) 1111+0=1111 ✓

D) 001=111+111=000
Wrong format

Pumping Lemma Example

Let's prove that ADD is not regular

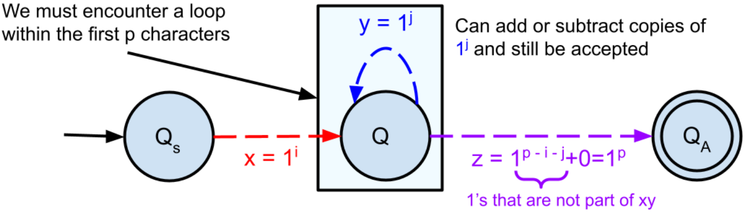
- AFSOC ADD is regular with pumping length p
- Choose $\underbrace{1...1}_{p}+0=\underbrace{1...1}_{p}$
- Split w into xyz such that $|y| > 0$ and $|xy| \leq p$
  - $\underbrace{1...1}_{x}\underbrace{1...1}_{y}\underbrace{1...1 + 0 = 1...1}_{z}$
  - y is not empty, xy contains all 1's
- When we pump y, the 1's are unequal, and the numbers won't add up
- $w \in L$ is not pumpable, conclude L is not regular

Pumping Lemma Example

L = {i+j=k | i, j, k are binary numbers, equation is valid}
AFSOC L can be decided by a DFA with p states

$$s: 1^p+0=1^p \in L$$

$$\underbrace{1\ldots1}_{p}+0=\underbrace{1\ldots1}_{p} \in L$$



Any DFA that accepts $1^p+0=1^p$ would accept
$1^i1^j1^j1^{p-i-j}+0=1^p \rightarrow 1^{p+j}+0=1^p \notin L$