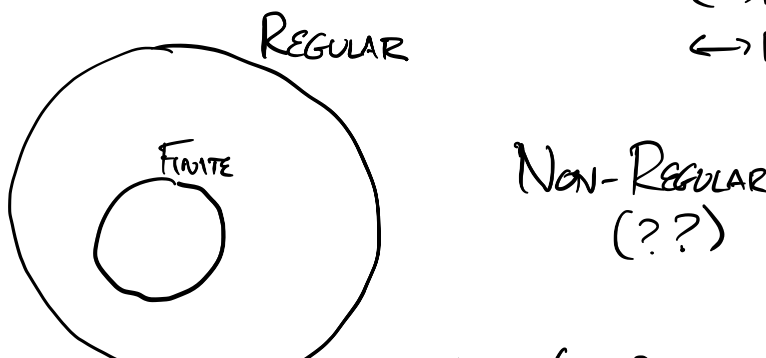
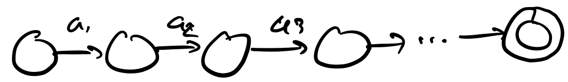
COMS 3261 - Lecture 4, Part 2: Non regular languages and the pumping lemma.

(Regular $\leftrightarrow$ DFA recognizes
$\leftrightarrow$ NFA recognizes
$\leftrightarrow$ Regular Expression)



<u>Fact</u>: All finite languages are regular. (Why? Take the union over all languages with one string in L.)



<u>Example</u>: Consider

$$B = \{0^k 1^k \mid k \in \mathbb{N}_{\geq 0}\}.$$

How could we recognize this? Seems like we need to 'count' or store an arbitrarily large number in our automaton's "memory."

<u>Ex. 2.</u> Consider

$$D = \{w \mid w \text{ has an equal number of 'ox' and 'xo' substrings.}\}.$$

OXXOOXOXXXOXOOO

4. The "Pumping Lemma"

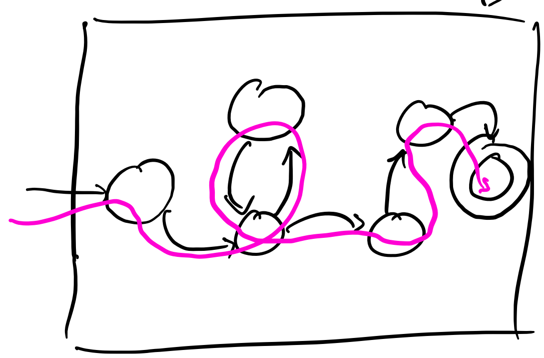
math speak for unintuitive

<u>Idea</u>: Come up with a "technical" property that all regular

languages have. ∴ If a language doesn't have that property, it's not regular.

Picture of a DFA.

D



Fact from earlier: Finite languages are regular.

So: nonregular languages must be infinite.

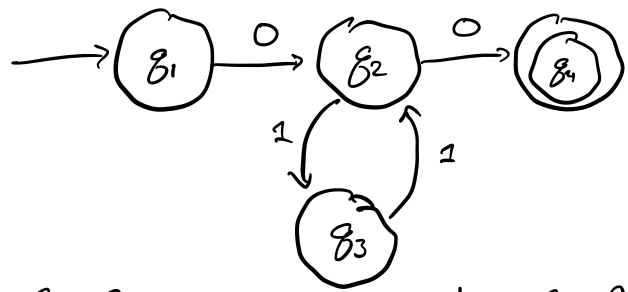
So: nonregular languages must accept long strings

Find a property of regular languages that accept long strings — loopiness.

More formally: Consider any DFA D that recognizes an infinite regular language. D has $|Q| = k$ states. Thus, any string $w \in L(D)$ with $|w| > k$ symbols must visit the same state twice and contain a loop.

Example: Language $0(11)^*0$. $= \{00, 0110, 011110 \ldots\}$

Any string $w \in L$ with $|w| \geq |Q| = 4$ visits the same state twice and contains a loop. Example: 0110 visits



0110 ↴

$q_1, \underline{q_2, q_3, q_2}, q_4$ on an accepting computation. Moreover, any <u>loop</u> can be repeated an arbitrary number of times to create an accepting computation for a new string in the language.

$q_1 \xrightarrow{0} q_2 \xrightarrow{1} q_3 \xrightarrow{1} q_2 \xrightarrow{11} q_3, q_2 \xrightarrow{0} q_4 \Rightarrow 011110$

$q_1 \xrightarrow{0} q_2 \xrightarrow{11} q_3, q_2 \xrightarrow{11} q_3, q_2 \xrightarrow{11} q_3, q_2 \xrightarrow{0} q_4 \Rightarrow 01111110$

<u>Lemma (Pumping Lemma.)</u> If A is a regular language, then there is some number p (the pumping length) such that, for any string $s \in A$ with length $|s| \geq p$, s can be divided into three substrings $s = x\underline{y}z$ satisfying:

1. For each $i \geq 0$, $xy^iz \in A$, // going around the loop
2. $|y| > 0$,
3. $|xy| \leq p$.

Proof idea: Consider the DFA of a regular language, and show that all sufficiently long strings have a loop.

<u>Proof:</u> Let A be a regular language, and let $D = (Q, \Sigma, \delta, q_1, F)$ be a DFA recognizing A. Let $p = |Q|$, the number of states in our DFA.

Now, let s be any string of length $n \ge p$ that is accepted by D. By the definition of DFA acceptance, if we write $s = s_1 s_2 s_3 \cdots s_n$, where $s_i \in \Sigma$, for all $i \in [n]$, there exists a sequence of states $r_1 r_2 \cdots, r_{n+1}$ that D enters while processing s. We have

$$r_{i+1} = \delta(r_i, s_i) \text{ for } 1 \le i \le n.$$

Because we have p states in our DFA, two of the first $p+1$ states in our sequence must be the same. Call the first duplicated state r_j and the first duplicate state r_ℓ

$$r_1 \xrightarrow{s_1} r_2 \xrightarrow{s_2} \cdots, r_{j-1} \xrightarrow{s_{j-1}} \Big| \, r_j \xrightarrow{s_j} \cdots, r_{\ell-1} \xrightarrow{s_{\ell-1}} \Big| \, r_\ell \xrightarrow{s_\ell} r_{\ell+1}, \cdots, r_n \xrightarrow{s_n} r_{n+1}.$$

Let $x = s_1 s_2 \cdots s_{j-1}$, $y = s_j s_{j+1} \cdots s_{\ell-1}$, and $z = s_\ell \cdots s_n$.
On the accepting computation for s, x takes us from r_1 to r_j, y takes us from r_j to r_j, and z takes us from r_j to r_n.

Thus: 1. xy^iz accepts for all $i \ge 0$ (going around the loop i times.)

2. $|y| \ge 0$, as $j < \ell$ by definition.

3. $|xy| \le p$. $|xy| = \ell - 1$, which is $\le p$ because we must find a duplicate in the first $p+1$ states. ∎

Zooming out: Want to show some languages aren't regular

We proved all regular languages satisfy the pumping lemma.
Thus if we show any language "can't be pumped," it is not regular.

Example. (Using the pumping lemma.)

Goal: show $B = \{0^n 1^n \mid n \geq 0\}$ is not regular.

Strategy: 1. Assume (for contradiction) that B is regular, so the pumping lemma holds.

"Assume for contradiction that B is regular, so the P.L. holds."

2. If the language is regular, it has a pumping length.

"Thus there exists some number p such that all strings $w \in B$ with length $|w| \geq p$ satisfy the conditions of the pumping lemma."

3. Find some string $s \in B$ such that $|s| \geq p$, and show that s cannot be pumped: no matter how we divide s into substrings $s = xyz$ such that $|y| > 0$, $|xy| \leq p$, there exists some i such that $xy^iz \notin B$.

"Select $s = 0^p 1^p$. How can we divide s into xyz with $|y| > 0$, $|xy| \leq p$?

Case 1. y is all zeroes. If $|y| > 0$, then $xyyz = xy^2z$, has more zeroes than ones and is not in the language.

$$S = \underbrace{0}_{x}\overbrace{000}^{y}\underbrace{1111}_{z}.$$

Now $xyyz =$

$$\underbrace{00}_{x}\underbrace{00}_{y}\underbrace{00}_{y}\underbrace{1111}_{z} \notin B.$$

Case 2. y is all ones.

$\therefore$ $xyyz$ is not in the language for the same reason.

Case 3. y contains zeroes and ones.

$$S = \underbrace{00}_{x}\underbrace{001}_{y}111.$$

Now: $xyyz =$

$$\underbrace{00}_{x}\underbrace{001}_{y}\underbrace{001}_{y}\underbrace{111}_{z} \notin B.$$

$xyyz \notin B$ because it contains 0's and 1's out of order.

So: $s \in B$, $|s| \geq p$, and there is no way to decompose s into substrings x, y, z that satisfy the pumping lemma.

Therefore our assumption that B is regular leads to contradiction, and B must be nonregular. ∎

Reminders: HW 2 due 7/12, HW 3 due 7/19 at ~~midnight~~ 11:59

Readings: end of 1.3, 1.4.

Next time: More PL examples

On to bigger & better language classes!