$L = \{ a^n \mid n \text{ is prime} \}$  $\Sigma = \{a\}$

By P.L. $\exists\, p$ s.t. ...

let $q$ be some prime $> p$

$\exists\, x\, y\, z$ s.t.

$a^q = xyz$

$|y| \neq 0$

$|xy| \leq p$

$xy^i z \in L \;\; \forall i \geq 0$

$0 \leq |x| < p$  $|x| = k$

$1 \leq |y| \leq p$  $|y| = j$

$|z| = \ell$

$xyz = a^k \cdot a^j a^\ell$

$\ell = q - k - j$

**Try 1:** $i = q + k$

$$xy^iz = a^k a^{j(q+k)} a^l$$

$$\cancel{= a^k a^{jq + jk + l}}$$

**try 2**

$$\cancel{i = 2q \cdot k - l}$$

**try 3**

$i = k + l$

$$\underset{a^k}{x} \quad \underset{a^{j(k+l)}}{y^i} \quad \underset{a^l}{z}$$

$$= a^{(j+1)(k+l)}$$

promising, but if $k+l = 1$, this is original string. (could be fixed by choosing $q > p+2$)

**Try 4**

$i = q + 1$

$$\underbrace{a^k}_{x} \; \underbrace{a^{j(q+1)}}_{y^i} \; \underbrace{a^l}_{z}$$

$$a^k a^j a^{qj} a^l$$

$$= a^q a^{qj}$$

$$= a^{q(j+1)}$$

$j \neq 0$ ($|y| > 0$), $j+1 \neq 1$, $q(j+1)$ is composite

Bingo!

$\Sigma = \{ (, ) \}$

$L = \{ w \mid$ parens are balanced $\}$

$\varepsilon$ , $()$ , $()()$ , $(()())$

not $)($

~~$L_2 = \{$~~

if $L$ is regular, so is

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

$L' = \{ (^n )^n \mid n \geq 0 \}$

$\equiv \{ a^n b^n \mid n \geq 0 \}$

C – (the programming language)

C satisfies the pumping lemma

main() { return ((((0)))); }

if regular $\exists p\ \forall x$ program $\exists xy \ldots$

$\rightarrow$ $x = \varepsilon$, $y = m$ pumps nicely – get new fn. names

But C not regular

$L = C \cap \{ \overbrace{\text{main()\{return}}^{x \text{ anything}} (^* 0)^*; \} \}$

reg exp.

L is not regular $\exists p \ldots$

$s =$ main() { return $(^p 0)^p$; }

Then if $y \notin (^*$, $i \neq 1$ gives invalid prefix

$y \in \Sigma^*$, $i \neq 1$ gives unbalanced parens.

Given a regular language L

& a string x how hard is it to decide

- $x \in L$ ?
- $L = \phi$ ?
- $L = \Sigma^*$ ?

A key issue: how is L (in general an infinite thing) "given" as input to our program? Some options

1. DFA
2. NFA
3. Reg. Exp.
4. A Java program

⋮

Does it matter?