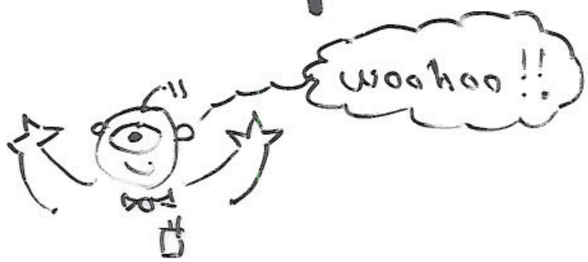
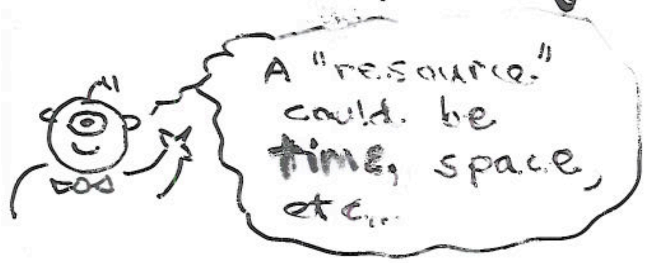
Complexity Theory!



- Restrict attention to recursive languages.

- How much of my favorite resource does it take a TM to decide (accept/reject) its input.

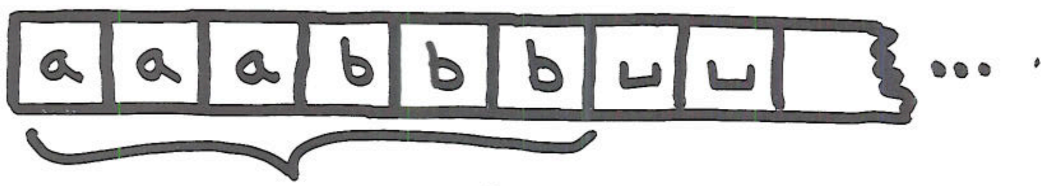


- We'll concentrate on time-complexity and then examine space-complexity.

Time Complexity

Example

$$L = \{ a^i b^i \mid i \geq 0 \}$$



length of input denoted by n

Definition

$$f(n) \in O(g(n))$$

if $\exists$ positive integers c and n_0 s.t. $f(n) \leq c \cdot g(n)$ $\forall$ $n \geq n_0$.

Definition:

regular o!e 1-tape TM

The <u>running time</u> of an algorithm on a TM is a function $f : \mathbb{N} \to \mathbb{N}$ s.t. $f(n)$ is the maximum number of steps taken by the TM over all inputs of length n.

Definition:

A set of languages!

$TIME(t(n)) = \{ L \mid L$ is a language decided by a TM whose running time is $O(t(n)) \}$

still a regular o!e 1-tape TM

Example:

$\{a^i b^i \mid i \geq 0\} \in TIME(\ ?\ )$

Question:

Is $\{a^i b^i \mid i \geq 0\} \in \mathrm{TIME}(n)$?

Using just one tape!

Polynomial Time and the Class $\mathcal{P}$

Definition:

$$\mathcal{P} = \bigcup_{k \geq 1} \text{TIME}(n^k)$$

$k \in \mathbb{Z}^+$

<u>Example</u>

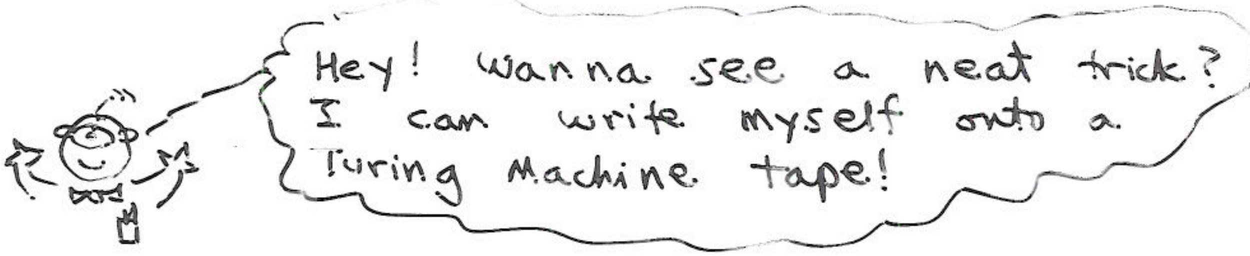
$$L = \{a^i b^i \mid i \geq 0\} \in \text{TIME}(n^2)$$

$$\Rightarrow L \in \mathcal{P}$$

<u>Note</u>

$$L \in \mathcal{P} \Rightarrow L \in \text{TIME}(n^k)$$

for some positive integer k.

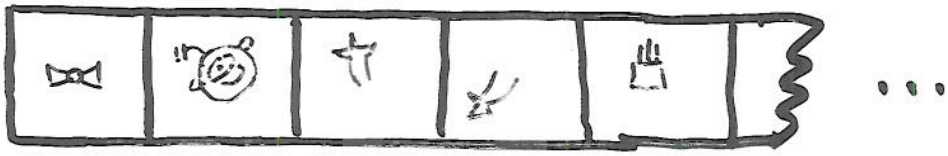


- Is $\mathcal{P}$ really of any practical consequence?

  - "$\mathcal{P}$ is all about deciding languages, not about solving real problems."

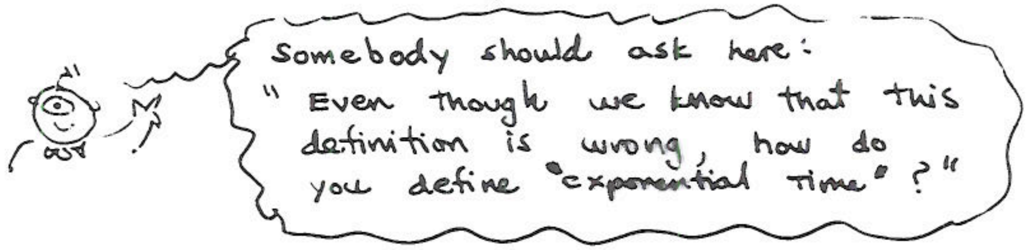
  - "$\mathcal{P}$ measures running time on a single tape! TM, not on a real computer."

  - "An n^{100} algorithm is in $\mathcal{P}$, but it's not very efficient!"



What is $\mathcal{NP}$?

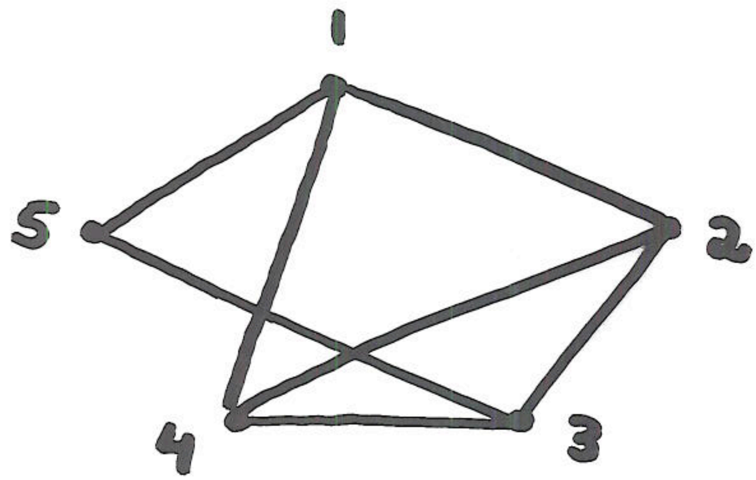
Attempt #1: $\mathcal{NP}$ is the set of all languages (decision problems) that can be solved in exponential time.



informal version! (we'll formalize it next.)

Attempt #2: $\mathcal{NP}$ is the set of all languages (decision problems) for which a solution can be verified in polynomial time.

The Hamiltonian Cycle Problem...



Decision Problem:

Given a graph G, does G contain a Hamiltonian cycle.

Language Problem:

$$L = \{ \langle G \rangle \mid G \text{ contains a Hamiltonian cycle} \}$$

verification:

$$\underbrace{\#\#1\#10\#11\#\ \ldots\ \#\#}_{\substack{\text{Graph encoding} \\ \text{length} = n}}\ \underbrace{1\#100\#10\#11\#101}_{\text{"certificate"}}$$

Defining $\mathcal{NP}$ Formally...

deterministic !!!

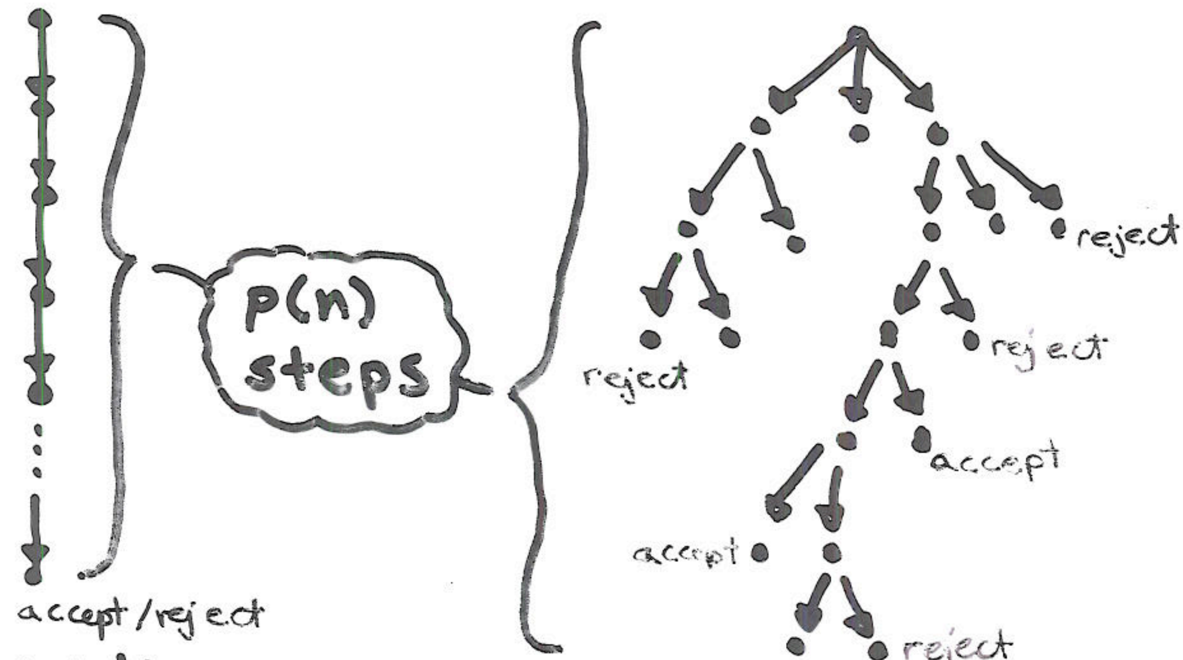
Definition: Let L be a language. A TM V is a polynomial-time verifier for L if $\exists$ a polynomial $p(n)$ s.t.

Such as Hamiltonian cycle or vertex cover, for example

0. For any input (w,c), V halts in time $p(|w|)$
1. For each $w \in L$ $\exists$ $c \in \Sigma^*$ s.t. when (w,c) is written on the tape, V accepts in time $p(\underbrace{|w|}_{\text{we call this "n" usually.}})$.
2. For each $w \notin L$ V rejects (w,c) in $p(|w|)$ time $\forall\, c \in \Sigma^*$.

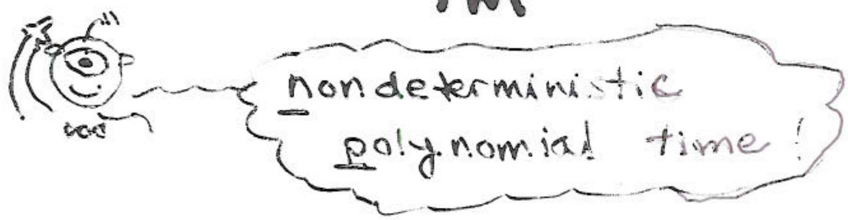
Definition: A language L is in $\mathcal{NP}$ if $\exists$ a polynomial-time verifier for L.

Another View of $\mathcal{NP}$



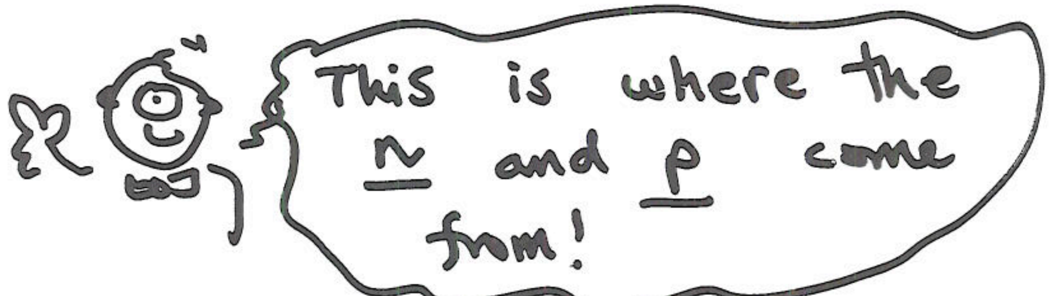
Deterministic TM

Nondeterministic TM



Theorem:

A language L is in $\mathcal{NP}$ iff it is decided by some nondeterministic polynomial-time TM.



repeated from previous slide :)

Theorem :

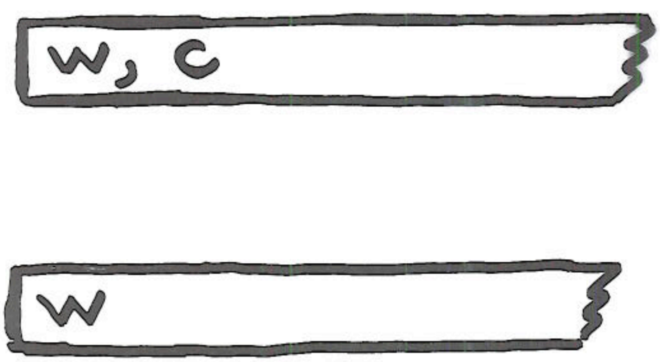
A language L is in $\mathcal{NP}$ iff it is decided by some nondeterministic polynomial-time TM

Proof:

$\Rightarrow$ Assume L is in $\mathcal{NP}$. Then by definition, $\exists$ a polynomial-time verifier V for L.

deterministic verifier V

nondeterministic decider M



runs in $p(n)$ time, where $n = |w|$.

$\Leftarrow$ Assume that L is accepted in $p(n)$ time by nondet. TM M. Show that a verifier V exists.

remember... a verifier is deterministic!