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:

The running time of an algorithm on a TM is a function $f : N \rightarrow N$ s.t. $f(n)$ is the maximum number of steps taken by the TM over all inputs of length $n$.

Definition:

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

Example:

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

Is $\exists b^*ib \geq 0^* \in \text{TIME}(n)$?

Tell me! Tell me!
I must know!

Using just one tape!
Polynomial Time
and the Class $\mathcal{P}$

**Definition:**

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

\[ k \in \mathbb{Z}^+ \]

**Example**

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

\[ \Rightarrow L \in \mathcal{P} \]

**Note**

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

for some positive integer $k$. 
Hey! Wanna see a neat trick? I can write myself onto a Turing Machine tape!

- Is $P$ really of any practical consequence?
  - "$P$ is all about deciding languages, not about solving real problems."
  - "$P$ measures running time on a TM, not on a real computer."
  - "An $n^{100}$ algorithm is in $P$, but it's not very efficient!"

In your face, theory-dude!
what is \( \mathcal{NP} \)?

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

Somebody should ask here: "Even though we know that this definition is wrong, how do you define "exponential time"?"

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

Get outta town!
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:

\[
\text{graph encoding length} = n \quad \text{or} \quad \text{“certificate”}
\]
Defining \( \mathsf{NP} \) Formally...

**Definition:** Let \( L \) be a language. A TM \( V \) is a polynomial-time verifier for \( L \) if there exists a polynomial \( p(n) \) such that:

1. For any input \((w,c)\), \( V \) halts in time \( p(1^{|w|}) \).
   - For each \( w \in L \), there exists \( c \in \Sigma^* \) such that when \((w,c)\) is written on the tape, \( V \) accepts in time \( p(1^{|w|}) \).
   - We call this "n" usually.

2. For each \( w \notin L \), \( V \) rejects \((w,c)\) in \( p(1^{|w|}) \) time for all \( c \in \Sigma^* \).

**Definition:** A language \( L \) is in \( \mathsf{NP} \) if there exists a polynomial-time verifier for \( L \).
Another View of $\mathbb{NP}$

Deterministic TM $\{p(n) \text{ steps} \}$

Nondeterministic TM

Theorem:

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

This is where the $n$ and $p$ come from!
Theorem:

A language \( L \) is in \( NP \) iff it is decided by some nondeterministic polynomial-time TM

Proof:

\( \Rightarrow \) Assume \( L \) is in \( NP \).
Then by definition, \( \exists \) a polynomial-time verifier \( V \) for \( L \).

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