Defn: A Turing machine $M$ runs in time $T(n)$ if for all inputs $x \in \Sigma^*$, $M(x)$ halts within $n = |x|$ steps.

If $M$ is an NTM, then we need all computations use $\leq t(n)$ steps:

$\text{DTIME}(t(n)) = \bigcup_{k=1}^\infty \text{NTIME}[t(n)]$.

$N^P = \bigcup_{k=1}^\infty \text{NTIME}[t(n)]$. "Nondeterministic Polytime".

The class $P$ ("Polynomial Time") is defined to be $\bigcup_{k=1}^\infty \text{DTIME}[n^k]$.

What belongs to $P$ — which problems are decidable in polytime?

- Every regular language $L$ is in $P$: Take a DFA $M$ for $L$, and $M$ is a DTM that runs in time $n$, in fact.
- $L \in P$.$\Longleftrightarrow$ $L$ is regular.

More generally, the DFA problem is in $P$.

**INST:** A DFA $M$ and an input $w$ to $M$, (coded as $X = (M, w)$)

**Question:** Does $M$ accept $w$? But $\text{TIME}(n, n^2) = \text{poly}(n)$ time?
How about \( \text{ALL-NFA} \)? It turns out that converting NFA to DFA is difficult for NFA.

**FACT:** Every CFL belongs to \( \text{P} \). The previous algorithm that took a \( \text{CFG} \) \( G \) in \( \text{CNF} \) st. \( L(G) = L \) and tried all derivations of length \( n-1 \) does not run in \( \text{P} \) time given a \( \text{CFG} \). In text (skim) there is a \( \text{D-Programming} \) algm in \( \text{O}(n^2) \) the \( \text{Not in Text} \) \( A_{\text{CFG}} = \{ \langle G, w \rangle : \text{WEL} \} \) is in \( \text{P} \) because there is a faster \( \text{CNF} \) conversion, not in the textbook.

- \( \text{CFG} \), \( \text{EPS} \) \( \{ \langle G \rangle : \varepsilon \in L \} \) are in \( \text{P} \) by \( \text{marking algm} \). They use an unbounded while loop, but each iteration either marks a new variable or the whole thing shrivels. \( \text{Time} \) \( \text{O}(1R|M|) \).

- How about \( \text{ALL-\langle G \rangle} \)? Undecidable, so certainly not in \( \text{P} \) or \( \text{NP} \).
A language \( L \) belongs to \( \text{NP} \) if and only if there are a polynomial \( q(n) \) and a language \( V \in \text{P} \) - a polynomial-time verifiable TM - \( V \) such that for all \( x \in \Sigma^* \):

\[
x \in L \iff (\exists y : 1 \leq q(|x|)) \land x, y \in V
\]

ie such that \( M_V a(\text{ybb} \leq |x|, y) \).

When \( x \in L \), any such \( y \) is called a certificate or witness.

Proof: if \( L = L(N) \) where \( N \) is an NTM running in \( \text{time } n^K \)

then given \( x \in L \), \( y \) can be an accepting computation history of \( N \) on \( x \).

Hence \( \text{time } = O(\text{length of } y) = O(n^K, n^K) = n^K = poly(n) \).

Consequently, given \( q \) and \( M_V \), build an NTM \( N \) that, on any input \( x \), guesses \( y \) and verifies \( (x, y) \in V \) by running \( M_V \).

The prime example of a language in \( \text{NP} \) is \text{SAT}:

**SATISFIABILITY (SAT):**

**Inst:** a Boolean formula \( \phi(x_1, \ldots, x_n) \in \{x_1, \overline{x_2}, x_3\} \land (x_1, \overline{x_2}, \overline{x_3}) \)

**Query:** Is there a truth assignment \( x_1 = a, \ldots, x_n = a_n \), \( \phi(a_1, \ldots, a_n) \in \{0, 1\} \)\(^3\),

that \( \text{makes } \phi(a_1, \ldots, a_n) = \text{TRUE}^2 \) \( \phi \) can be satisfied, indeed.

Equivalently, is \( \neg \phi \) not a tautology?

by all assignments except \( (0, 1) \), \( (0, 0) \), \( (0, 0) \), \( (1, 0) \), \( (1, 0) \), \( (1, 1) \), \( (1, 1) \).

**TAUT = \exists B : \text{fs } \forall : \text{fs is a tautology}\) is \( \neg \phi \) the complement of SAT.
Theorem: SAT \in NP, also TAUT \in NP.

Proof: If \( \phi \) is satisfiable, we can guess a satisfying assignment \( \vec{a} \) of \( \phi \) and check it, which is easily verifiable.

Since TAUT = SAT, TAUT \in NP. \( \Box \) so TAUT \in P.

**Definition:** A language \( A \) mapping reduces to a language \( B \) in polynomial time if there is a function \( f \) (computable in polynomial time) s.t. \( \forall x : x \in A \iff f(x) \in B \).

**Theorem:** If \( \text{NP} \subseteq \text{coNP} \) then \( \text{P} = \text{NP} \).

**Definition:** If \( B \in \text{NP} \) and for all \( A \in \text{NP} \), \( A \leq_m^P B \) then \( B \) is \text{NP-complete}.

**Theorem:** SAT is \text{NP-complete} (Cook-Levin Theorem, 1970).

**Proof:** Let any \( A \in \text{NP} \) be given. Show \( A \leq_m^P \text{SAT} \). Take \( \forall \exists \in \text{SAT} \).

\[ \forall x : x \in A \iff \exists y < x, y \in B \text{ where } 1 + q(n) \text{ for a polynomial } q. \]

**Burn:** Move initial \( x \) to bottom as handshakes as circuits \& NAND gates

\[ C_n = \begin{cases} 1 & \text{if } (\forall y < x) \exists y \text{ s.t. } 1 + q(n) \text{ for } \text{polynomial } q. \end{cases} \]

Every NAND gate functions correctly and the output will be \( W \) s.t. \( \phi \in SAT \), where \( \phi = W \wedge (A \wedge q) \text{ or } \forall \exists \).