Theorem: A language \( A \) belongs to \( \text{NP} \) iff there is a 
Verifier \( \text{DTM} \ V \) that runs in polynomial time and a polynomial \( p \) such 
for all \( x \): \( x \in A \iff (\exists y : |y| \leq p(|x|)) \ [V \text{ accepts } \langle x, y \rangle] \). 

Moreover, the body of \( V \) can be either:
- The predicate \( T(<N_A>, x, \mathcal{C}) \) applied to a poly-time \( \text{NFA} \ N_A \) for, 
where whole computations \( \mathcal{C} \) are the "\( y \)".
- An easy-to-build (given \( x \) and \( n = |x| \)) 
sequence \( \{C_n\} \) of poly-size circuits of \( \text{NAND} \) gates and \( n + p(n) \) inputs.

Wlog: \( \Sigma \subseteq \{0, 1\} \), we can demand \( |y| = p(n) \), all nonder \( \mathcal{C} \) steps by \( \text{NFA} \) are binary 
and we use single-tape \( \text{DTM} \)s and \( \text{DFMs} \). 
Building \( C_n \) from \( x \) and \( n \) takes \( \text{depol time} \).

Proof: \( \iff \) Given \( V \) in any form above, we can take \( p(n) \) to be its polynomial 
runtime or circuit size. Define an \( \text{NFA} \ N \) that on any input \( x \) uses (upto) \( p(n) \) 
nondeterministic steps to "guess" \( y \) and then deterministically runs \( V(x, y) \), accepting 
\( x \) on that run if and only if \( V \) accepts \( \langle x, y \rangle \). Then \( N \) is a poly-time \( \text{DTM} \) 
\( L(N) = A \).

\( \Rightarrow \) Given \( A \in \text{NP} \), we can take an \( \text{NFA} \ N_A \) that runs in some polynomial \( \text{NFA} \ p(n) \) 
such that \( L(N_A) = A \). Within \( p(n) \) steps, IDs of \( N_A \) \( (x) \) can grow to size at most \( p(n) \). 
Hence accepting computations \( \mathcal{C} \), 
can be coded by strings \( y \in \{0, 1\}^* \) of length \( q(n) = O(p(n) \times p(n)) \). 
So the \( T(<N_A>, x, \mathcal{C}) \) predicate from 
the last lecture is a poly-time verifier.
Moreover we can stack \( 2 \text{Os} \) at \( N \) like \( X \).
Focal Example of a Problem/Language in NP:

SAT: 
  INST: A Boolean formula $\phi(x_1, \ldots, x_n)$ in variables $x_1, \ldots, x_n$ with logical gates $\land, \lor, \neg$.

**Q.** Is there an assignment $a_1, \ldots, a_n \in \{0, 1\}^n$ that satisfies $\phi$, i.e., $\phi(a) = \text{TRUE}$?

$$N = \lvert \phi \rvert \quad \text{then} \quad n = o(N).$$

$$\text{SAT} = \left\{ \langle \phi \rangle : \left( \exists \vec{a} \in \{0, 1\}^n \right) : \phi(\vec{a}) = \text{true} \right\}$$

Again $n << \lvert \phi \rvert$ but we think of $n$ as the size.

$: SAT \in NP.$

Example 2: 
  INST: An undirected graph $G = (V, E)$ and an integer $K \leq n = \lvert V \rvert$.

$: \text{INDSET} \in \text{NP}.$

**Q.** Does there exist a set $I \subseteq V, \lvert I \rvert = K$ st. no two nodes in $I$ have an edge between them.

$: \text{INDSET} \subseteq \text{SAT} \subseteq (\forall \vec{a} \in \{0, 1\}^n) \phi(\vec{a}) \not= \text{TRUE}.$

$: \text{TUT is complementary to SAT, so it is in } \text{co-NP} = \{ L : \overline{L} \in \text{NP} \}.$
**Cook-Levin Theorem:** SAT \( \leq^P \) NP and for all

Let any \( A \in \text{NP} \) be given. Take a \( p(n) \)-time NIM \( A_N \) s.t. \( L(A_N) = A \).

Given any \( X \), take \( n=|X| \), and compute the circuit \( C_n \) of NAND gates for the \( p(n) \).

We start with the property that \( x \in A \iff \exists \gamma \in \{0,1\}^p \).

\[ C_n(x, \gamma) = w_0 = 1. \]

Every NAND gate in \( C_n \) must function correctly by

- AND-ing together all \( \gamma \) clauses \( \phi_g \) over all gates \( g \) in \( C_n \).
- Conjoin the singleton clause \( (w_0) \) mandating \( w_0 = 1 \).
- Finally, given a particular \( x \in \{0,1\}^n \), use \( n \) singleton clauses \( (x_i) \) or \( \neg x_i \) to set each bit.

Then \( \phi \) has one variable for each wire or input gate of \( C_n \) but \( C_n \) has \( O(p(n)^2) \) wires and is easy to build so \( f(x) = \phi \) is a polynomial time computable function. And \( x \in A \iff \exists \gamma \) then is an assignment to \( \gamma_1 \ldots \gamma_p \) that induces an assigned value to every wire that satisfies \( \phi \).

Thus \( A \leq^P \text{SAT} \), indeed \( \exists \text{SAT} \) with \( \phi \) is a conjunction \( (1 \lor \cdots \lor C_n) \) and each clause \( C_i \) has \( n \) most \( 3 \) literals.
Another NP-complete Problem: $\neg{\text{ALL}}^n_{\text{NFA}}$

$\neg{\text{ALL}}^n_{\text{NFA}}$ is in $\text{NP}$

because we can guess $a$ and verify that $N$ does not accept $a$.

- not by converting $N$ to DFA

but by tracing "lights" directly

(3) SAT $\leq^P_{\text{m}} \neg{\text{ALL}}^n_{\text{NFA}}$

$\phi \leq_{\text{m}} N_0$

$\phi = \bigvee_i C_i \
C_1 \land C_2 \land \ldots \land C_m$

We will make it so that some string $a$ is not accepted if $a$ does not refute any clauses, i.e. it satisfies all $C_i = (x_1 \lor \neg x_2 \lor x_3)$$

$N_0$ has $O(nm)$ states and is built in polynomial time, so $\neg{\text{ALL}}^n_{\text{NFA}}$ is complete. (So is $\text{INDUCE}$, and similarly, ditto)

Also, by complementing, we get $\text{TAUT} \leq^P \text{ALL}^n_{\text{NFA}}$

$\text{ALL}^n_{\text{NFA}}$ is complete for $\text{co-NP}$. Since $\text{TAUT}$ is complete for $\text{CNP}$, this gives us $\text{ALL}^n_{\text{NFA}} \equiv^P \text{TAUT}$

just like $\neg{\text{ALL}}^n_{\text{NFA}} \equiv^P \text{SAT}$.

The $\text{ALL}^n_{\text{NFA}}$ problem is maybe harder since

strings $x$ longer than the # of states of $N$ are involved. It is co-NP hard.