Validity

\[ \overline{\text{VAL}} = \{ \phi \mid \phi \not\in \text{VAL} \} \] — the complement of \text{VAL} is in \text{NP}.

Guess a \textit{falsifying} truth assignment and verify it.

Such an algorithm does not work for \text{VAL}.

In this case, we have to determine whether \textit{every} truth assignment results in \textit{true} — a requirement that does not sit as well with the definition of acceptance by a nondeterministic machine.
Complementation

If we interchange accepting and rejecting states in a deterministic machine that decides the language \( L \), we get one that accepts \( \overline{L} \).

If a language \( L \in \mathbf{P} \), then also \( \overline{L} \in \mathbf{P} \).

Complexity classes defined in terms of nondeterministic machine models are not necessarily closed under complementation of languages.

Define,

\( \text{co-NP} \) – the languages whose complements are in \( \mathbf{NP} \).
The complexity class \( \text{NP} \) can be characterised as the collection of languages of the form:

\[
L = \{ x \mid \exists y R(x, y) \}
\]

Where \( R \) is a relation on strings satisfying two key conditions

1. \( R \) is decidable in polynomial time.
2. \( R \) is \textit{polynomially balanced}. That is, there is a polynomial \( p \) such that if \( R(x, y) \) and the length of \( x \) is \( n \), then the length of \( y \) is no more than \( p(n) \).
As \( \text{co-NP} \) is the collection of complements of languages in \( \text{NP} \), and \( \text{P} \) is closed under complementation, \( \text{co-NP} \) can also be characterised as the collection of languages of the form:

\[
L = \{ x | \forall y |y| < p(|x|) \rightarrow R'(x, y) \}
\]

\( \text{NP} \) – the collection of languages with succinct certificates of membership. 
\( \text{co-NP} \) – the collection of languages with succinct certificates of disqualification.
Any of the situations is consistent with our present state of knowledge:

- \( P = \text{NP} = \text{co-NP} \)
- \( P = \text{NP} \cap \text{co-NP} \neq \text{NP} \neq \text{co-NP} \)
- \( P \neq \text{NP} \cap \text{co-NP} = \text{NP} = \text{co-NP} \)
- \( P \neq \text{NP} \cap \text{co-NP} \neq \text{NP} \neq \text{co-NP} \)
VAL – the collection of Boolean expressions that are *valid* is *co-NP-complete*.

Any language $L$ that is the complement of an NP-complete language is *co-NP-complete*.

Any reduction of a language $L_1$ to $L_2$ is also a reduction of $\overline{L_1}$–the complement of $L_1$–to $\overline{L_2}$–the complement of $L_2$.

There is an easy reduction from the complement of SAT to VAL, namely the map that takes an expression to its negation.

$$\text{VAL} \in P \Rightarrow P = \text{NP} = \text{co-NP}$$

$$\text{VAL} \in \text{NP} \Rightarrow \text{NP} = \text{co-NP}$$
Consider the decision problem **PRIME**:  

*Given a number $x$, is it prime?*

This problem is in **co-NP**.

$$
\forall y (y < x \rightarrow (y = 1 \lor \neg(div(y, x))))
$$

*Note again, the algorithm that checks for all numbers up to $\sqrt{n}$ whether any of them divides $n$, is not polynomial, as $\sqrt{n}$ is not polynomial in the size of the input string, which is $\log n$.***
Another way of putting this is that Composite is in NP.

Pratt (1976) showed that PRIME is in NP, by exhibiting succinct certificates of primality based on:

A number \( p > 2 \) is prime if, and only if, there is a number \( r \), \( 1 < r < p \), such that \( r^{p-1} = 1 \mod p \) and \( r^{\frac{p-1}{q}} \neq 1 \mod p \) for all prime divisors \( q \) of \( p - 1 \).
In 2002, Agrawal, Kayal and Saxena showed that PRIME is in P.

If $a$ is co-prime to $p$,

$$(x - a)^p \equiv (x^p - a) \pmod{p}$$

if, and only if, $p$ is a prime.

Checking this equivalence would take too long. Instead, the equivalence is checked \emph{modulo} a polynomial $x^r - 1$, for “suitable” $r$.

The existence of suitable small $r$ relies on deep results in number theory.
Consider the language \textbf{Factor}

\[ \{(x, k) \mid x \text{ has a factor } y \text{ with } 1 < y < k\} \]

\textbf{Factor} \in \text{NP} \cap \text{co-NP}

\textit{Certificate of membership}—a factor of \( x \) less than \( k \).

\textit{Certificate of disqualification}—the prime factorisation of \( x \).
Graph Isomorphism

Given two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, is there a bijection

$$\iota : V_1 \rightarrow V_2$$

such that for every $u, v \in V_1$,

$$(u, v) \in E_1 \quad \text{if, and only if,} \quad (\iota(u), \iota(v)) \in E_2.$$
Graph Isomorphism is

- in \text{NP}
- not known to be in \text{P}
- not known to be in \text{co-NP}
- not known (or \text{expected}) to be \text{NP}-complete
- recently shown to be in \text{quasi-polynomial time}, i.e. in \text{TIME}(n^{(\log n)^k}) for a constant \(k\).
The Travelling Salesman Problem was originally conceived of as an optimisation problem to find a minimum cost tour.

We forced it into the mould of a decision problem – TSP – in order to fit it into our theory of NP-completeness.

Similar arguments can be made about the problems CLIQUE and IND.
This is still reasonable, as we are establishing the *difficulty* of the problems.

A polynomial time solution to the optimisation version would give a polynomial time solution to the decision problem.

Also, a polynomial time solution to the decision problem would allow a polynomial time algorithm for *finding the optimal value*, using binary search, if necessary.