Confronted by an NP-complete problem, say constructing a timetable, what can one do?

- It’s a single instance, does asymptotic complexity matter?
- What’s the critical size? Is scalability important?
- Are there guaranteed restrictions on the input? Will a special purpose algorithm suffice?
- Will an approximate solution suffice? Are performance guarantees required?
- Are there useful heuristics that can constrain a search? Ways of ordering choices to control backtracking?
- Can you use a SAT-solver?
We define $\text{VAL}$—the set of valid Boolean expressions—to be those Boolean expressions for which every assignment of truth values to variables yields an expression equivalent to $\text{true}$.

$$\phi \in \text{VAL} \iff \neg \phi \not\in \text{SAT}$$

By an exhaustive search algorithm similar to the one for $\text{SAT}$, $\text{VAL}$ is in $\text{TIME}(n^22^n)$.

Is $\text{VAL} \in \text{NP}$?
Validity

\[ \overline{\text{VAL}} = \{ \phi \mid \phi \notin \text{VAL} \} \] —the complement of VAL is in NP.

Guess a falsifying truth assignment and verify it.

Such an algorithm does not work for VAL.

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

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

Define,

**co-NP** – the languages whose complements are in NP.
The complexity class **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 *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 \( p(n) \).
co-NP

As co-NP is the collection of complements of languages in NP, and P is closed under complementation, co-NP can also be characterised as the collection of languages of the form:

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

NP – the collection of languages with succinct certificates of membership.

co-NP – the collection of languages with succinct certificates of disqualification.
Any of the situations is consistent with our present state of knowledge:

- $P = NP = \text{co-NP}$
- $P = NP \cap \text{co-NP} \neq NP \neq \text{co-NP}$
- $P \neq NP \cap \text{co-NP} = NP = \text{co-NP}$
- $P \neq NP \cap \text{co-NP} \neq NP \neq \text{co-NP}$
**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.

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

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

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

This problem is in co-NP.

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

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 $p - 1$. 