The P vs NP problem

Polynomial — an expression consisting of variables raised to some power and their coefficients. For example a general second order polynomial is of the form ax^2+bx+c.

Time complexity of an algorithm— the amount of time taken by an algorithm to run as a function of the length of input. Commonly expressed using the big O notation. For example if we write an algorithm to print out all the elements in an array of size 2n one-by-one, its time complexity would be O(n).

Polynomial time complexity — the time complexity of the algorithm is n^{O(1)}
P = the set of problems that are solvable in polynomial time by a Deterministic Turing Machine.
NP = the set of decision problems (answer is either yes or no) that are solvable in nondeterministic polynomial time i.e can be solved in polynomial time by a Nondeterministic Turing Machine[4].

Nondeterministic Turing Machine(NTM) — a branching machine. If many options for the next step of computation exists this machine can go down all of them at the same time. NTMs are capable of guessing the correct option out of polynomially many options in O(1) time.

OBSERVATION

An alternative definition to NP would be the set of decision problems for which if a potential solution is provided, a DTM can verify its correctness in polynomial time. It is important to note that all P problems also belong to NP because if a problem is solvable by DTM in polynomial time, verifying a potential solution will be easier than solving. So a DTM would be able to verify also in polynomial time.

So trivially, $P \subseteq NP$ i.e P is a subset of NP.

Also it is important to know that all computers that exist today are DTMs and NTM is a purely theoretical computer used for thought experiments.

A problem is NP complete ?

Proving NP-completeness of a problem involves 2 steps. First we have to show that the problem belongs to NP and then we have to show it is NP-hard. The steps can be explained further as follows,

Step 1 — Showing $X \in NP$. find a nondeterministic algorithm for X. But a more practical way would be to give a polynomial time verification for X if a potential solution is provided. Step 2 — Showing X is NP-hard. Reduce from a known NP-complete problem to X. So by the implication 3 we saw previously will mean that X is NP-hard.