The Class $\mathcal{NP}$

$\mathcal{NP}$ *is the problems that can be solved in polynomial time by a nondeterministic machine.*

$\mathcal{NP}$

The ***time*** taken by nondeterministic TM is the length of the *longest* branch.

The collection of all problems that can be solved in polynomial time by a nondeterministic machine is called $\mathcal{NP}$.

That is, language $L \in \mathcal{NP}$ if there is k and an NTM that decides L that runs in time $O(n^k)$.

$\mathcal{P}$ *versus* $\mathcal{NP}$

It is immediate that

$$\mathcal{P} \subseteq \mathcal{NP}$$

But does nondeterminism buy one anything?

Example: Satisfiability

In boolean formulas, a ***clause*** is the *or* of literals. A formula is in ***conjunctive normal form*** if the *and* of clauses.

A ***satisfying assignment*** is one that makes the formula TRUE. For example,

$$x \wedge (x \vee y \vee \bar{z}) \wedge (\bar{x} \vee \bar{y})$$

is in conjunctive normal form and is satisfiable: set x to TRUE and y, z to FALSE.

Example: SAT

$\texttt{SAT} = \{ \langle\phi\rangle : \phi$ is a boolean formula in conjunctive normal form having a satisfying assignment $\}$.

$\texttt{SAT} \in \mathcal{NP}$. The machine nondeterministically guesses the assignment. However, the number of assignments in exponential, so there is no obvious polynomial-time algorithm.

Arthur versus Merlin

It can be shown that NP is equivalent to the following situation. We have a (deterministic) Verifier that runs in polynomial time (sometimes called Arthur). They need to solve the problem given a hint from an omniscient consultant (sometime called Merlin). The hint is called the ***certificate***.

Certificate

The certificate here is the satisfying assignment. Note that Arthur *must* get the correct answer. If the answer to the instance is yes, then given the correct assignment, Arthur can convince themselves that the boolean formula is satisfiable. If the answer to the instance is no, then Arthur must not be conned into saying yes, no matter what magic Merlin tries.

Example: HAMPATH

A ***hamiltonian*** path in a graph is a path that visits each vertex once and only once.

$$\texttt{HAMPATH} = \{\, \langle G, a, b \rangle : G \text{ is graph with hamiltonian path from } a \text{ to } b \,\}$$

We do not know how to decide HAMPATH in polynomial time. (Trying all possible paths fails, as there are exponentially many.)

Example: `HAMPATH` *is in* $\mathcal{NP}$

But there is a fast nondeterministic program. Guess path node by node, at each stage choosing an unvisited node. Time taken is at most quadratic in the number of nodes.

Equivalently, the certificate for `HAMPATH` is the hamiltonian path. All Arthur has to do is to check that the edges form a path (end of one is start of next), that the path starts at a and ends at b, and that each node is visited exactly once.

$\mathcal{P}$ *versus* $\mathcal{NP}$

Maybe $\mathcal{P}$ and $\mathcal{NP}$ are different sets. However, we do not know.

Conjecture. $\mathcal{P} \neq \mathcal{NP}$

The Clay Institute offers \$1 million for a proof or disproof.

Consequences

One can, however, identify problems that are the hardest in $\mathcal{NP}$, called ***$\mathcal{NP}$-complete problems***. They have the property that, if there is a polynomial-time algorithm for any one of them, then there is a polynomial-time algorithm for all of $\mathcal{NP}$.

There are numerous $\mathcal{NP}$-complete problems that industry would love to solve quickly. But, almost all cryptography assumes that decoding without the secret key is harder than decoding with the secret key, which might not be true if $\mathcal{P} = \mathcal{NP}$.