Lecture 15

Section 1.9

$\boxed{\text{SAT is NP Complete}}$

In this section, we shall explain what the above means and why it is important.

We first have to understand the time complexity of an algorithm/programme.

~~Let~~ Given a programme, we define its time complexity as follows: amongst all inputs of length n let $f(n)$ be the longest time needed to compute an output amongst all those inputs. To compute $f(n)$ for a given programme is difficult, but it turns out that it is good enough to get an estimate (say, an upper bound).

The form taken by such an $f(n)$ leads to the following terminology:

if $f(n) = an$ $(a \in \mathbb{R})$

then we say f is <u>linear</u>

if $f(n) = an^2$ we say f is <u>quadratic</u>

.... if $f(n) = an^m$ for some fixed m

we say $f(n)$ is <u>polynomial</u>.

<u>Definition</u> The class of problems P is the class of all problems which have a polynomial time algorithm.

It can take a lot of work to decide whether a problem is in P. For example, the problem of deciding whether a nat. no. is prime or not was only shown to be in P in 2004.

Problems in P are "<u>nice</u>" – polynomial time algorithms are fast

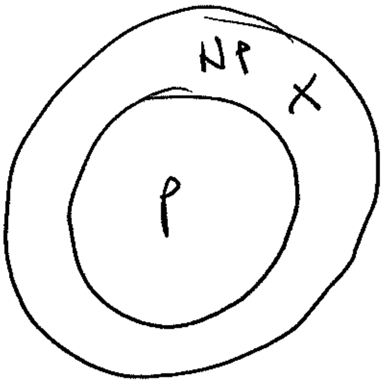
We now define the class NP — non-deterministic polynomial time problems. These are problems where if you come up with a possible solution you can check whether it is a solution or not "quickly" — meaning in polynomial time.

Example SAT $\in$ NP.

Example Graph coloring $\in$ NP.

Clearly, $P \subseteq NP$.

We suspect that $P \neq NP$ but there is no proof



How might we go about resolving the issue of whether $P = NP$ or $P \neq NP$?

Imagine a problem $X \in NP$ with the following characteristic: if $X \in P$ then $P = NP$.
i.e. X is the "nastiest" problem in NP.

We call any such problem NP-Complete
(Stephan Cook)

<u>Cook's Theorem</u> (1971) SAT is NP-Complete
(proved independently by the Russian Leonid Levin

Actually, many problems are NP-Complete and they are all equivalent to each other.

Why should you care about NP-Complete Problems? Because 1000's of natural problems have been proved to be NP-Complete.

If a polynomial time algorithm was ever found to solve one NP-complete problem then, in fact, $P = NP$.