## The complexity class NP

Consider this problem: Given an undirected graph G, does G have a Hamiltonian path?

Given a graph G, if we guess a list  $\pi$  then it is easy to check whether  $\pi$  is a Hamiltonian path of G

- Check that the items of π are a permutation of nodes(G)
- Check that successive nodes of  $\pi$  are adjacent in G

These checks can be carried out in p-time.

Thus the problem becomes easy (p-time) if we guess the path. The Hamiltonian path  $\pi$  acts as a certificate that HamPath(G).

If we guess  $\pi$  and we discover that  $\pi$  is not a Hamiltonian path of G, then we are none the wiser, since it might be that:

- G has a (different) Hamiltonian path or
- G has no Hamiltonian path

Nevertheless, it remains the case that if G has a Hamiltonian path then some guess will prove correct.

To make this more precise, let us define the associated verification problem which we call Ver-HamPath:

• Given a graph G and a list , is  $\pi$  a Hamiltonian path of G?

Note that  $Ver\text{-}HamPath(G,\pi)$  is in P. Also,

$$HamPath(G) \iff \exists \pi. Ver\text{-}HamPath(G, \pi)$$

A decision problem D(x) is in NP (non-deterministic polynomial time) if there is a problem E(x, y) in P and a polynomial p(n) such that

- $D(x) \iff \exists y. E(x,y)$
- $E(x,y) \implies |y| \le p(|x|)$  (E is polynomially balanced)

We require that the certificate y is polynomially bounded in x since otherwise it would take too long to guess y.

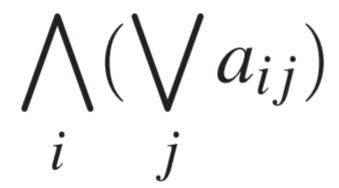
The guess for the Hamiltonian path can be p-bounded in the size of G. Therefore,  $HamPath \in NP$  according to the definition of NP.

To sum up the difference between P and NP, we have:

- P class of decision problems which can be efficiently solved
- NP class of decision problems which can be efficiently verified

We now introduce a famous decision problem from logic: Boolean satisfiability

A formula  $\phi$  of propositional logic is in conjunctive normal form (CNF) if it is of the form:



where each  $a_{ij}$  is either a variable x or its negation  $\neg x$ 

- Terms  $a_{ij}$  are called *literals*
- Terms  $\bigvee_{j} a_{ij}$  are called *clauses*

The SAT (satisfiability) problem is:

• Given a formula  $\phi$  in CNF, is  $\phi$  satisfiable (is there an assignment v to the variables of  $\phi$  which makes  $\phi$  true)?

It seems that SAT is not decidable in p-time: we have to try all possible truth assignments. If  $\phi$  has m variables there are  $2^m$  assignments — exponentially many.

We can let  $|\phi|$  be the number of symbols in  $\phi$  and |v| be m (size of the domain of v). Notice that m can be of similar size to  $|\phi|$  - every literal could be a different variable.

However SAT does belong to NP, as we can see using the guess and verify method. Given a formula  $\phi$ :

- guess a truth assignment v
- verify in p-time that v satisfies  $\phi$

As we did with the Hamiltonian Path problem, we define the associated verification problem VER-SAT:  $VER\text{-}SAT(\phi,v)\iff \phi$  is in CNF and v satisfies  $\phi$ 

Then:

- $SAT(\phi) \iff \exists v. VER\text{-}SAT(\phi, v)$
- $VER\text{-}SAT(\phi, v) \implies |v| \le |\phi|$  (VER-SAT is p-balanced)

So we have confirmed that  $SAT \in NP$ 

## If a decision problem is in P then it is in NP, i.e. $P \subseteq NP$

**Proof.** Suppose that problem *D* is in P.

Idea: to verify that D(x) holds we don't need to guess a certificate y — we can decide D(x) directly.

More formally, we define E(x,y) iff D(x) and  $y = \varepsilon$  (the empty string — a dummy guess). Then clearly

$$D(x)$$
 iff  $\exists y. E(x, y)$  and  $|y| \le p(|x|)$ 

It remains unknown whether P = NP despite many researchers' attempts.

Most researchers believe that  $P \neq NP$