The Class P

Def 7.12: The class P consists of languages that are decidable in polynomial time, i.e.,

\[ P = \bigcup_k \text{TIME}(n^k) \]

Example:

\[ \text{PATH} = \{ <G,s,t> \mid G \text{ is a digraph that has a path from } s \text{ to } t \} \]

- e.g. BFS from s, see if hit t

  - Running time: \( O(n+m) \)
  - \( n \) = # vertices
  - \( m \) = # edges

  - Or less accurate: \( O(n^2) \) → good enough to show \( \text{PATH} \in P \)

  - (a little more on a TM, still polynomial)
The Class P

Def 7.12: The class $P$ consists of languages that are decidable in polynomial time, i.e.,

$$P = \bigcup_k \text{TIME}(n^k)$$

Example:

RELPRIME = \{ \langle x, y \rangle \mid x \text{ and } y \text{ are relatively primes} \}

for $i=2$ to $\min(x,y)-1$:
  
  if $i$ divides both $x$ and $y$ then return not rel.primes

  return rel.primes

run.time: $O(\min(x,y))$

another idea:

Euclid’s gcd algo
→ takes $O(\log x + \log y)$ steps

Hence, RELPRIME $\in P$

Another way:

to write down $x$, we need only $\lceil \log_{10} x \rceil + 1$ digits

length of the input $\langle x, y \rangle$ is about $\lceil \log_{10} x \rceil + \lceil \log_{10} y \rceil + 3$

about $O(10^n) = \text{exponential time}$ in the length of the input
Def 7.12: The class $P$ consists of languages that are decidable in polynomial time, i.e.,

$$P = \bigcup_k \text{TIME}(n^k)$$

Example:

$\text{PRIME} = \{ <x> \mid x \text{ is a prime} \}$

for $i = 2$ to $\sqrt{x}$:
  if $i$ divides $x$, then not prime
  return prime

$O(\sqrt{x})$ steps $= O(\sqrt{10^n}) = O(\sqrt{10^n})$ exptime

the length of the input $= \log x = \log 10^n = \log x = 10^n$
The Class NP

Example:

\[ \text{HAMPATH} = \{ <G,s,t> \mid G \text{ is a graph with Hamiltonian path from } s \text{ to } t \} \]

NP:

1. guess sequence of vertices starting at \( s \), ending at \( t \), going through every vertex once
   
   e.g.
   
   \[ 5 \ 2 \ 4 \ 1 \ 3 \ 1 \]

2. verify:
   every pair of consecutive vertices is connected by an edge
The Class NP

Example:

\[ \text{COMPOSITES} = \{ x \mid x = pq, \text{ for some } p, q > 1 \} \]

NP:

1. guess \( p \), an integer \( > 1 \) and \( < x \)

2. verify:
   \[ \text{whether } p \text{ divides } x \]
Def 7.18: A **verifier** for a language $A$ is an algorithm $V$, where

$$A = \{ w \mid V \text{ accepts } <w,c> \text{ for some string } c \}$$

**Polynomial-time verifier** runs in (deterministic) time polynomial in the length of $w$. The string $c$ is called the **certificate**, or **proof**, of the membership in $A$. 
The Class NP

Def 7.18: A **verifier** for a language A is an algorithm A, where

\[ A = \{ w \mid V \text{ accepts } <w,c> \text{ for some string } c \} \]

**Polynomial-time verifier** runs in (deterministic) time polynomial in the length of w. The string c is called the **certificate**, or **proof**, of the membership in A.

Def 7.19: **NP** is the class of languages that have polynomial time verifiers.
The Class NP

Thm 7.20: A language is in NP iff it is decided by some nondeterministic polynomial-time TM.

Def 7.21: $\text{NTIME}(t(n)) = \{ L \mid L$ is a language decided by a $O(t(n))$-time nondeterministic TM $\}$.

Thus, $\text{NP} = \bigcup_k \text{NTIME}(n^k)$
The Class NP

Thm 7.20: A language is in NP iff it is decided by some nondeterministic polynomial-time TM.

Def 7.21: \( NTIME(t(n)) = \{ L \mid L \text{ is a language decided by a } O(t(n))-\text{time nondeterministic TM} \} \).

Thus, \( NP = \bigcup_k NTIME(n^k) \).
The Class NP

Example:

\[ \text{CLIQUE} = \{ <G,k> \mid G \text{ is undirected graph with a } k\text{-clique} \} \]
The Class NP

Example:

\[ SAT = \{ \langle \phi \rangle \mid \phi \text{ is a satisfiable Boolean formula} \} \]

\[ \phi = (x_1 \lor x_2) \land (\neg x_1 \lor x_3 \lor \neg x_2) \land (x_1 \lor \neg x_3) \land (\neg x_1 \lor \neg x_3) \]

Is there a TRUE/FALSE assignment for \( x_1, x_2, x_3 \) s.t. \( \phi \) is TRUE?

E.g. for the above \( \phi \):
- \( x_1 = F \)
- \( x_2 = T \)
- \( x_3 = F \)

In NP:
1) guess a T/F assignment
   - for every variable
2) verify whether \( \phi \) is true

(not known to be in P)
The Class NP

Wrapping up:
- P - exists polynomial-time algorithm
- NP - exists polynomial-time verifier

BIG open problem:

Is P = NP ???

Note: also exists a class coNP, the class of complements of problems in NP (e.g. CLIQUE\(^c\), “is every clique of a given graph of different size than k?”). We do not know if NP = coNP.