- Some problems/ languages that are known to be in P:
  - Is the length of a string odd?
  - Strings of the form $0^n 1^n$.
  - Given a graph G and 2 vertices u & v, is there a path from u to v?

- Given a graph G, is the graph 2-colorable? (Is G bipartite?)

[- Try writing down algorithms for these]

- Given two numbers a and b, are they relatively prime to each other?

$$\{\langle a, b\rangle \mid a \text{ and } b \text{ are relatively prime}\}$$

PATH (G, u, v):

- Mark the vertex u;

Repeat (until no new vertex get marked)

- If there is an edge (a, b) with 'a' already marked, mark b also.

- If v is marked, answer YES.

else, NO.

- We started writing algorithms in a 'higher' level. But that is OK. It is easy to see it runs (makes moves) in time proportional to

(at most)

2 * No. of edges in the graph.

- 'Size' of the input is proportional to $n+m$ (no. of nodes + no. of edges)

- Rel-Prime (a,b): Use a 'subroutine'.

GCD: 1. If $b=0$, return a;

2. Call $GCD(b, a \bmod b)$;

- We can have a TM that performs this.

- Rel-Prime (a,b) : If $GCD(a,b)=1$, YES.

Else NO.

- What is the running time?
- Values of a and b are reduced by half (at least) in every recursive call to the subroutine GCD.
- Input size: no. of bits used for a & b.

$$n = \log_2 a + \log_2 b + 1$$

- Number of calls to GCD:

  lesser of $\log_2 a$ and $\log_2 b$.

  i.e., $O(n)$.

- Finding a mod b:

  Can be done in time polynomial in n.

Def^n: NTIME$(t(n))$:

$\{L \mid L$ is a language decided by an $O(t(n))$ time non-deterministic TM$\}$.

Def^n: $NP = \bigcup_k NTIME(n^k)$.

- An alternative, easier, definition:

A YES/NO problem (language) $\in NP$ if and only if a 'YES' answer can be 'verified' in polynomial time; with the help of a 'certificate'.

Examples:

- Is the given number N composite?

If someone says 'YES',
they can give a non-trivial factor of N
(a factor other than 1 and N)
as a certificate. We can verify it easily.

- Given a graph G and a number k, Does G have a clique of size k?
  - For a YES answer, k vertices of the clique can be given as a certificate.
  - We can verify by checking the adjacency lists of those k vertices, in poly. time.

This means

$$\text{COMPOSITES} \in NP$$

$$\text{CLIQUE} \in NP.$$

- However, since there exists a polynomial time algorithm to check if a number is PRIME or COMPOSITE, COMPOSITES $\in P$ also.
- It is not known if CLIQUES $\in P$.

- Note that $TIME(n^k) \subseteq NTIME(n^k)$,

  and hence $P \subseteq NP$.

- By the other definition also,

  if a problem/language is in P,

  it is obvious that a YES answer can be verified in polynomial time. (No certificate is needed for that. 'NO' answer also can be verified.)