Additional Content

P, NP, NP-Hard and NP-Complete

Dr. Shruti Mishra
SCOPE

Contents

- Satisfiability
- Reducibility
- Clique Problem

Introduction

Polynomial Time

Linear Search – n

Binary Search – $\log n$

Insertion Sort – n^2

Merge Sort – $n\log n$

Matrix multiplication – n^3

Exponential Time

0/1 Knapsack Problem – 2^n

TSP – 2^n

Sum of Subsets – 2^n

Graph Colouring – 2^n

Hamiltonian Cycle – 2^n

We want all these to solve in polynomial time as these are time consuming algorithms

Satisfiability

- We need a base solution to solve all the exponential problems.
- We are able to solve one all can be solved.
- But need a base problem.
- Satisfiability defines that called CNF-Satisfiability based on propositional calculus formula using Boolean variables.

$x_i = \{ x_1, x_2, x_3 \}$

CNF= $\underbrace{(x_1 \vee \bar{x}_2 \vee x_3)}_{C_1} \wedge \underbrace{(\bar{x}_1 \vee x_2 \vee \bar{x}_3)}_{C_2}$ → How much we will take to solve this.

clause :- C_1 C_2

Cont.

Possible values of x_i can be

x_1	x_2	x_3
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

$\boxed{\text{8 possibilities.}}$

$2^3 = 2^n$

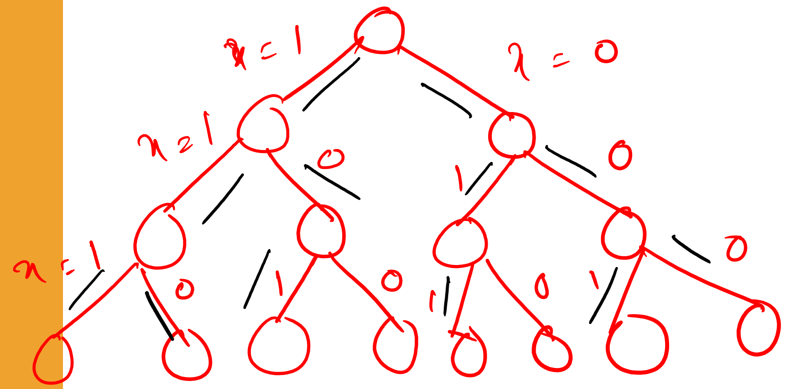
Can be placed in the clauses to check

For 'n' variables 2^n, that is exponential time to solve.

CNF satisfiability similar to exponential problems.

Contd..

Search space tree:-



Path given us a solⁿ of all possible values.

Hence, Satisfiability too can be represented in a search space tree.

If we could prove that the above problem is solvable in polynomial time, then same can be proved for exponential problem.

Reducibility

Let us take two separate problems.

Sat $\propto$ 0/1 knapsack

↑

Reducible.

Let $I_1 \xrightarrow{\text{taken poly time}} I_2$

[If Sat can be NP hard, then knapsack also is NP-hard]

Hence, we can prove the relationship like this.

Sat $\propto$ Long.., L

↑

NP-hard.

If one of them can be solved in polynomial time, then the other can be solved too.

'Sat' has transitivity property. Hence, Sat $\propto L_1$, then $L_1 \propto L_2$

NP-hard is also NP-hard.

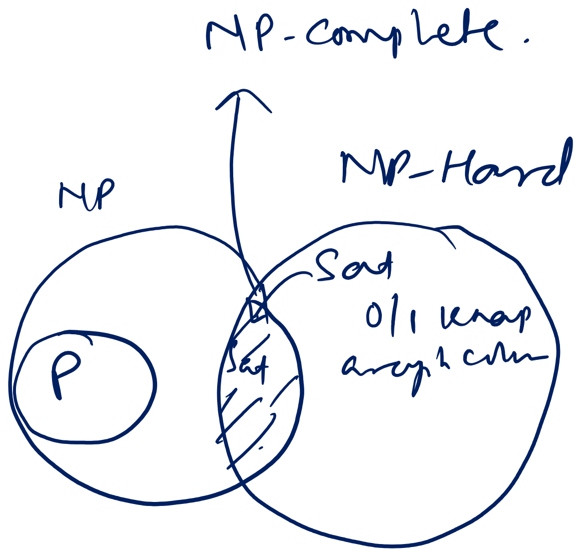
Sat is NP-Hard and the same can be used to prove for all the exponential problems.

Sat is also NP-complete.

$\text{Sat} \propto L$ — NP, NP Hard — NP C

NP

give something and proved that it is NP.



Reducibility

- a problem Q can be reduced to another problem Q′ if any instance of Q can be "easily rephrased" as an instance of Q′, the solution to which provides a solution to the instance of Q.
- Is a linear equation reducible to a quadratic equation?
  - Sure! Let coefficient of the square term be 0
- Also has a transitive relation.

An example reduction

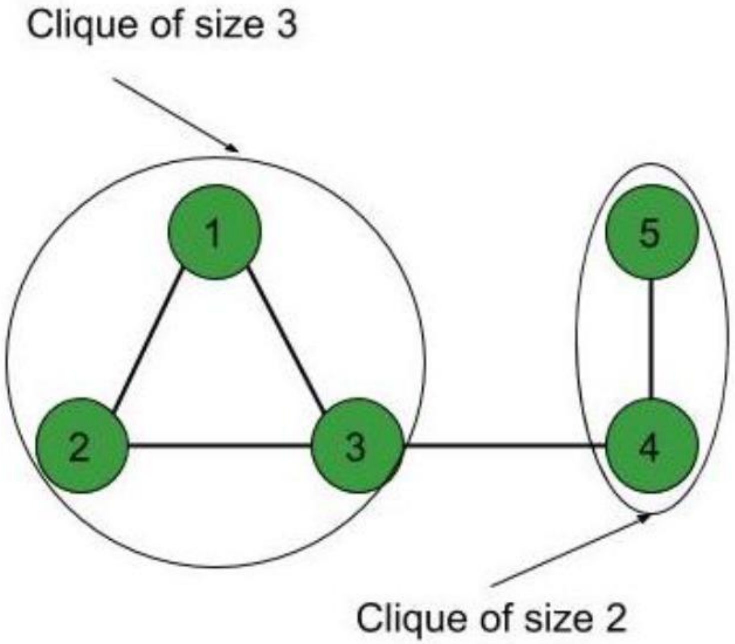
CLIQUE problem

- A clique in an undirected graph is a subset of vertices such that each pair is connected by an edge
- Take a problem instance in 3-CNF SAT and convert it to CLIQUE finding

CLIQUE

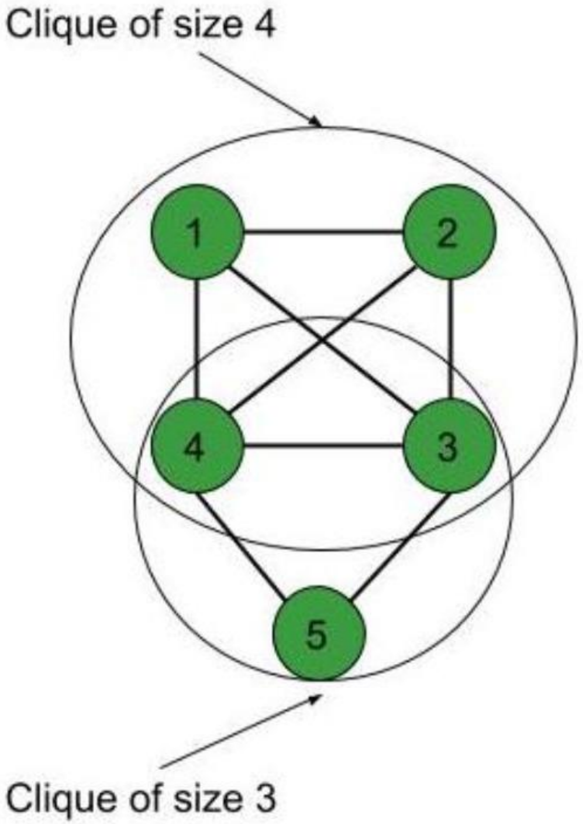
Proof that Clique Decision problem is NP-Complete

- A clique is a subgraph of a graph such that all the vertices in this subgraph are connected with each other that is the subgraph is a complete graph.
- The *Maximal Clique Problem is to find the maximum sized clique of a given graph G*, that is a complete graph which is a subgraph of G and contains the maximum number of vertices.
- This is an *optimization problem*.
- Correspondingly, the Clique Decision Problem is *to find if a clique of size k exists in the given graph or not.*



The above graph contains a maximum clique of size 3

Fig. (1)



The above graph contains a maximum clique of size 4

Fig. (2)

* A clique of size 2 is also present in Fig. (2)

Contd..

- To prove that a problem is NP-Complete, we have to show that it belongs to both NP and NP-Hard Classes. (Since NP-Complete problems are NP-Hard problems which also belong to NP)

- Let us prove first NP-Hard, later you can prove NP by verifiability/gaining certificate.

The Clique Decision Problem belongs to NP-Hard

- A problem L belongs to NP-Hard if every NP problem is reducible to L in polynomial time.
- Now, let the Clique Decision Problem by C.
- To prove that C is NP-Hard, we take an already known NP-Hard problem, say S, and reduce it to C for a particular instance.
- If this reduction can be done in polynomial time, then C is also an NP-Hard problem.
- The Boolean Satisfiability Problem (S) is an NP-Complete problem as proved by the Cook's theorem.
- Therefore, every problem in NP can be reduced to S in polynomial time.
- Thus, if S is reducible to C in polynomial time, every NP problem can be reduced to C in polynomial time, thereby proving C to be NP-Hard.

Contd..

- Let the boolean expression be –

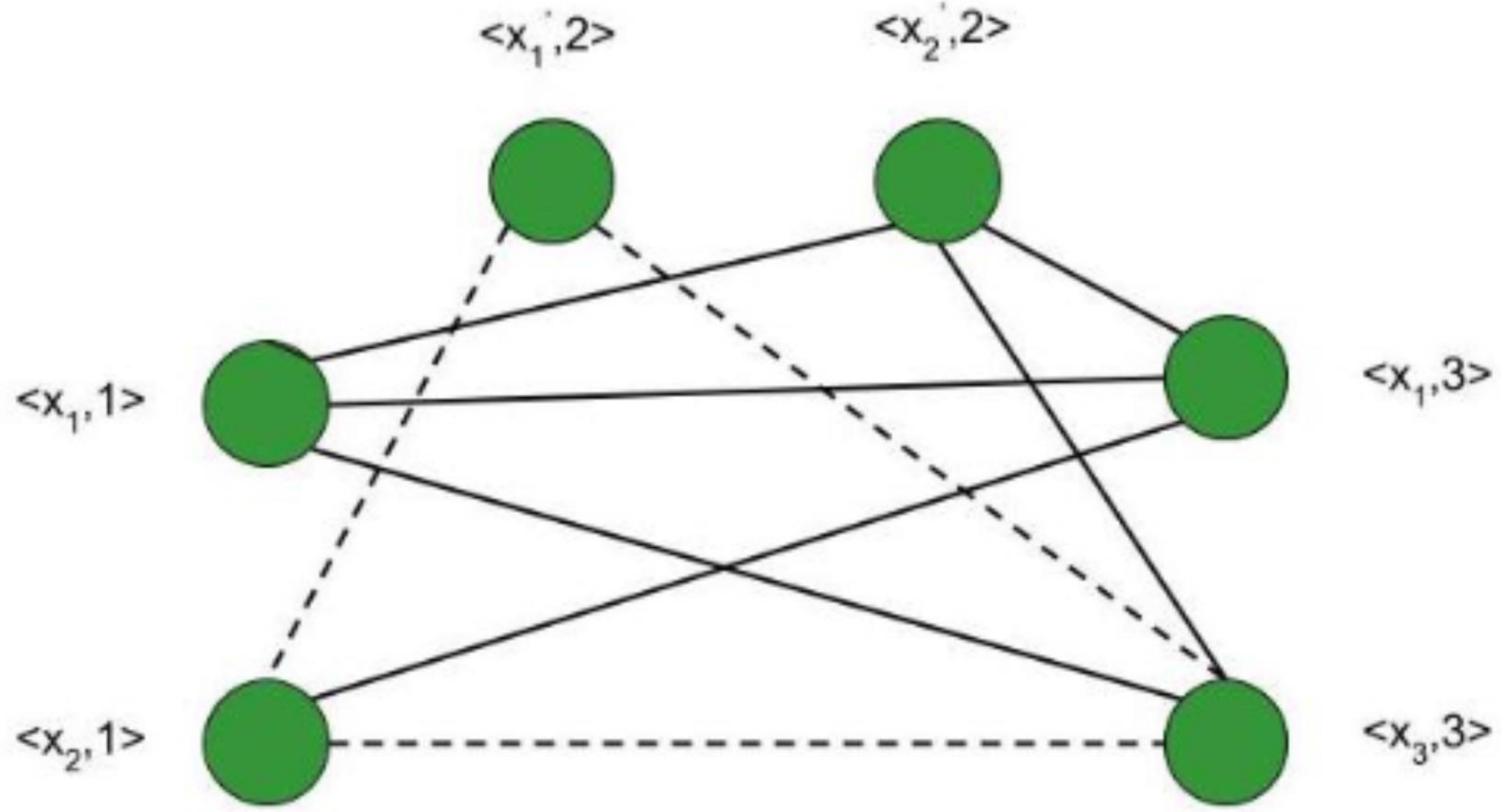
$F = (x_1 \vee x_2) \wedge (x_1' \vee x_2') \wedge (x_1 \vee x_3)$ where x_1, x_2, x_3 are the variables, '^' denotes logical 'and', 'v' denotes logical 'or' and x' denotes the complement of x.

- Let the expression within each parentheses be a clause. Hence, we have three clauses – C_1, C_2 and C_3.

- Consider the vertices as –

$<x_1, 1>$; $<x_2, 1>$; $<x_1', 2>$; $<x_2', 2>$; $<x_1, 3>$; $<x_3, 3>$ where the second term in each vertex denotes the clause number they belong to.

- We connect these vertices such that –
    - No two vertices belonging to the same clause are connected.
    - No variable is connected to its complement.


$<x_1',2>$
$<x_2',2>$
$<x_1,1>$
$<x_1,3>$
$<x_2,1>$
$<x_3,3>$

Contd..

- Thus, the graph G (V, E) is constructed such that – V = { <a, i> | a belongs to C_i } and E = { (<a, i>, <b, j>) | i is not equal to j ; b is not equal to a' } .
- Consider the subgraph of G with the vertices $<x_2, 1>$; $<x_1', 2>$; $<x_3, 3>$. It forms a clique of size 3 (Depicted by dotted line in above figure) .
- Corresponding to this, for the assignment – $<x_1, x_2, x_3> = <0, 1, 1>$ F evaluates to true.
- Therefore, if we have k clauses in our satisfiability expression, we get a max clique of size k and for the corresponding assignment of values, the satisfiability expression evaluates to true.
- Hence, for a particular instance, the satisfiability problem is reduced to the clique decision problem.
- Therefore, the Clique Decision Problem is NP-Hard.