

① PCS is polytime reducible to ILP.

PCS

- ILP → i) Directed Acyclic Graph (G). {Vertex \rightarrow Job (unit time)}
ii) No. of processors (p)
iii) Deadline (d)

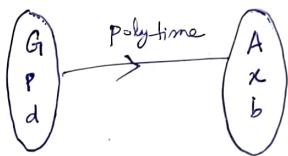
Decision Problem → Can G_1 be scheduled in p processors such that the execution completes in d steps?

ILP

- ILP → i) Matrix A ($m \times n$)
ii) Vector b, ($m \times 1$)

Decision Problem → Does there exist a vector ' x ' ($n \times 1$) s.t
 $Ax \leq b, x \in \{0, 1\}$

Reduction



In PCS arc $(u, v) \rightarrow u$ must execute before v . i.e $T(u) < T(v)$

Consider u & v

$t_u =$ Time at which vertex u is scheduled.

$x_{ut} = 1$, if u is scheduled at time t .
0, else

Constraint → i) Everytime at step t , only ' p ' nodes (Jobs) can be scheduled.

i.e $\sum_u x_{ut} \leq p \quad \forall \text{ all } t.$ (At time t , max of p nodes can be scheduled as max processors available)

$$\Rightarrow x_{1t} + x_{2t} + x_{3t} + \dots + x_{nt} \leq p \quad (\text{is } p)$$

(assumes we have a total of n nodes)

ii) Every node must be scheduled exactly at one instant. (Every job needs to executed without any preemption)

$$\text{Lc} \quad \sum_w x_w = 1 \longrightarrow O(d)$$

$$\Rightarrow \sum_w x_w \leq L$$

Now,

Consider the arc (u, v) - i.e. u executed before v .

$$\text{Time at which } u \text{ is scheduled} = 1 \cdot x_{u1} + 2 \cdot x_{u2} + \dots + d \cdot x_{ud}$$

\downarrow

if u is scheduled at time z ,
then $z \cdot 1 = z$

$\left[\begin{array}{l} \because \text{deadline} \\ \text{is } d, u \text{ must} \\ \text{executed at} \\ \text{time } \leq d \end{array} \right]$

Since, u must execute before v ,

$$\therefore 1 \cdot x_{u1} + 2 \cdot x_{u2} + \dots + d \cdot x_{ud} \leq 1 \cdot x_{v1} + 2 \cdot x_{v2} + \dots + d \cdot x_{vd}$$

$$\Rightarrow 1 \cdot x_{u1} + 2 \cdot x_{u2} + \dots + d \cdot x_{ud} \leq 1 \cdot x_{v1} + 2 \cdot x_{v2} + \dots + d \cdot x_{vd} - 1$$

$$\hookrightarrow O(V)$$

$\left(\begin{array}{l} \text{let } u \text{ execute at time } 4, \\ \text{then } v \text{ can execute at time } 5 \\ \text{So, } 4 \leq (5-1) \end{array} \right)$

$$\text{Time Complexity} = O(d + v) = O(n)$$

Thus, PCS can be reduced to the form $Ax \leq b$ (ILP).

$$x = \begin{bmatrix} x_{u1} \\ x_{u2} \\ \vdots \\ x_{vd} \end{bmatrix}, x \in (0, 1)$$

PCS \leq_p ILP.

② Undirected Ham-Cycle is NPc.

Ham-cycle \rightarrow Cycle in undirected graph $G = (V, E)$ that covers every vertex exactly once, excluding the source vertex.

Decision Problem \rightarrow Given a graph $G = (V, E)$, does it contain a Ham-cycle consisting of all the vertices of V ?

To prove NP

\exists : Sequence of vertices forming Ham-cycle i.e. Set $V' = \{v_1, v_n\}$

\exists : instance of the graph $G = (V, E)$, $|V| = n$

Verifier $V(x,y)$ will check whether all vertices in $V' \subseteq V$ & each pair of vertices in V' are adjacent.

$V(x,y) :$

- i) if $|V'| \leq |V|$, then
return false. $\left(\because V' \text{ will contain the source vertex twice} \right)$
- ii) for every pair of vertices $\{u, v\}$ in V' :
check if edge $(u, v) \notin E$,
return false
- iii) return true.

$V(x,y)$ can verify the solution V' in $O(V+E)$ i.e. polynomial time.

\therefore Ham-Cycle decision problem is in NP.

To proof NP-C

Reduction \rightarrow (Ham-path \leq_p Ham-Cycle.)

Every instance of the Ham-path problem in graph $G = (V, E)$ can be reduced to Ham-Cycle problem in graph $G' = (V', E')$.

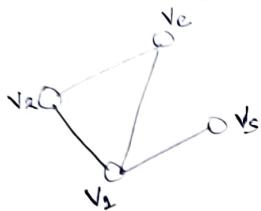
Construction of G' \rightarrow

i) Add v vertices of the graph G along with a new vertex v_{new} $(|V'| = |V| + 1)$

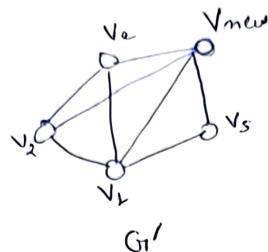
ii) Add edges E of the graph G in G' . Now add new edges between v_{new} and remaining $|V|$ vertices in G' such that v_{new} is connected to all the $|V|$ vertices $\cdot (E' = E + |V|)$

Now, assume graph G_1 has a Ham-path covering (v) vertices, say starting at v_s and ends at v_e . Extend the Ham-path, i.e., Ham-cycle by the edges (v_e, v_{new}) and (v_{new}, v_s) . Thus G_1' will now have a Ham-cycle traversing all vertices once.

G_1' will have a Ham-cycle iff G_1 has a Ham-ejected path.



G_1



$\{v_5, v_1, v_2, v_4, v_e\} \rightarrow$ Ham path in G_1

$\{v_5, v_1, v_2, v_e, v_{new}, v_s\} \rightarrow$ Ham cycle in G_1' .

This reduction can be done in $O(V+E)$ polynomial time,
i.e. Ham path \leq_p Ham-cycle.

③ Vertex-Cover is in NPC.

Vertex-Cover \rightarrow subset of vertices that covers all the edges in the graph.

Decision Problem \rightarrow Given an instance of $G_1 = (V, E)$ and the integer 'k', does there exist a vertex cover of size NP?

x: Instance of Graph $G_1 = (V, E)$, integer k

y: Subset of vertices $V' = \{v_1, v_2, \dots, v_k\}$

Verifier $V(x, y)$ will check whether the vertices in V' cover all the edges in G_1 .

$V(x, y)$: i) if $|V'| \neq k$,

return false.

ii) for each vertex v in V' :

remove all edges adjacent to v from E .

if $|E| \neq \emptyset$

return false

return true.

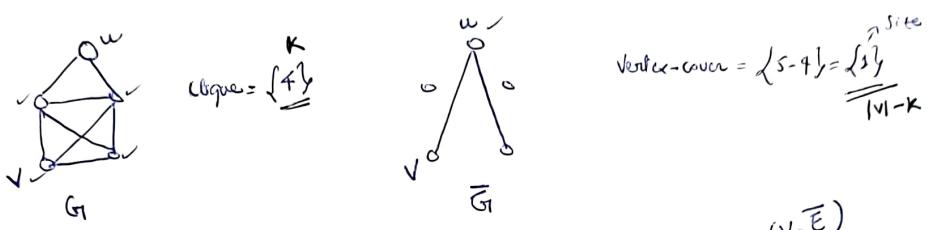
Verifier $V(wy)$ can verify the solution V' in $O(|E|+V)$ polynomial time.

\therefore Vertex-cover \in NP.

To proof NP-C, ($\text{Clique} \rightarrow \text{Vertex-Cover}$)

Given graph $G = (V, E)$, K does there exist a clique of size K .

To show that any instance (G, K) of clique problem can be reduced to an instance of the vertex-cover problem.



Consider Graph $G = (V, E)$ of the clique problem. \bar{G} is the complement of G . Problem of finding whether a clique of size 'k' exist in G is same as problem of finding whether a vertex-cover of size $|V|-k$ in \bar{G} .

Construction - Assume there exist clique of size k in G .

Set of vertices in clique be V' . i.e $|V'| = k$

In \bar{G} , consider any edge (u, v) . Then at least one of u or v must be in set $V-V'$. Because if both u & v are in V' \rightarrow edge $(u, v) \in E(G)$ i.e edge (u, v) is in G which contradicts the fact that $(u, v) \in E(\bar{G})$.

Thus, all edges in \bar{G} are covered by vertices in the set $V-V'$ i.e size of set = $|V|-k$.

Now, Assume there is a vertex cover V'' of size $|V|-k$ in \bar{G} . So, all edges in \bar{G} are connected to some vertex in V'' . This means for any edge (u, v) of \bar{G} , both u & v cannot be outside of V'' .

All the edges (u, v) s.t $u, v \notin V''$ are in G and forms a clique of size $|V| - (|V|-k) = k$.

So, there exist a clique of size K in G_1 iff there exist a vertex cover of size $|V| - K$ in $\overline{G_1}$.
Hence instance of Clique problem can be reduced to instance of Vertex-cover problem.

for generating $\overline{G_1}$, we need $O(n)$ time.

So, Clique \leq_p Vertex-cover.

(4) Set-Cover is in NPC

Set-cover \rightarrow Consider finite set S . Set cover is the subset of subsets of S_1, S_2, \dots, S_n s.t. $S_1 \cup S_2 \cup \dots \cup S_n = S$.

Decision Problem \rightarrow Given a finite set S , collection of S_1, S_2, \dots, S_n subsets and +ve integer K , does there exist a set-cover of size K ? (i.e. $S_1 \cup S_2 \cup \dots \cup S_K = S$)

Proof NP

$x: S, \{S_1, S_2, \dots, S_n\}, K$

$y: \text{Set of subsets } S' = \{S'_1, S'_2, \dots\}$

$V(x, y):$ i) if $|S'| \neq K$
return false.

ii) for all the elements in S'_i , $i=1 \text{ to } K$
mark elements in S & element $\in S'_i$ — $O(n)$

iii) for all elements in S ,
if any element is not marked
return false
return true. — $O(n)$

$V(x, y)$ can perform the verification in $O(n)$ time.

\therefore Set-cover \in NP.

Proof NPC

(Vertex-cover \rightarrow Set-cover)

Consider the problem instance of Vertex-cover of size K in $G = (V, E)$.

Construct an instance of set-cover \rightarrow

i) $S = E$ (Set of all edges in G_1 be S)

ii) $S_i = \{(u,v) \in E \mid u=i \text{ or } v=i\} \forall i \in V$.

(Create subset S_i for every vertex i consisting of all the edges incident to i)

iii) $K = k$ (size of set cover = size of vertex cover)

The vertices corresponding to the subsets in set cover of S will be vertex cover of G_1 .

Validation \rightarrow

Assume V' of size k is the ^{vertex cover} solution in Graph $G_1 = (V, E)$.

So, vertices in V' cover all the edges in E .

\therefore Any subset S_i is the set of edges incident on vertex i , union of all S_i s.t $i \in V'$ must contain all the edges in E .

\therefore all the edges in E form the set S , $\{S_i, i \in V'\}$ is the set cover of S , with size k .

Conversely,

Consider solution to the set cover problem $(S, \{S_1, \dots, S_k\}, K)$.

Solution be the set of subsets, $S' = \{S_1, S_2, \dots, S_K\}$,

Then $S_1 \cup S_2 \cup \dots \cup S_K = S$.

$\therefore E = S$, S' covers all the edges in E .

\because each $S_i \in S'$ are the edges incident on vertex i ,
the vertices $\{i : S_i \in S'\}$ forms a vertex cover of G_1
with size k .

This reduction can be done in polynomial time (need to
scan all the edges of G_1 to create subsets $S_i \in S$).

$O(E)$.

\therefore Vertex Cover \leq_p Set-cover.

⑤ Clique is in NPC

Clique \rightarrow Subgraph of a graph G in which all the vertices are connected. (Complete subgraph).

Decision Problem \rightarrow Given a graph $G = (V, E)$ and integer K , does G contains a clique of size K ?

Proof NP

x : instance of graph $G = (V, E)$, K .

y : set $V' \subseteq V(G)$.

$V(x, y)$: i) if $|V'| \neq K$,
return false.

ii) for every pair of Vertices (u, v) , $u \neq v$, $uv \in V'$
if edge $(u, v) \notin E$ or edge (u, v) do not exist
return false.

return true.

This verification can be done in polynomial time.

Proof NPC $(3\text{-SAT} \rightarrow \text{Clique})$

Consider the instance of 3-CNF-SAT.

Let $\phi = C_1 \wedge C_2 \wedge \dots \wedge C_K$ be a boolean formula in 3-CNF
with K clauses.

Each clause C_n , $n=1, 2, \dots, K$ has 3 distinct literals l_1^n, l_2^n, l_3^n .

Graph $G = (V, E)$ is constructed as below \rightarrow

1) For each clause $C_n = (l_1^n \vee l_2^n \vee l_3^n)$ in ϕ , create 3 vertices
in V . corresponding to each literals in C_n .

2) Create edge between the vertices v_i^n and v_j^n when —

a) v_i^n and v_j^n are in different clauses. i.e. $n \neq s$. (-)

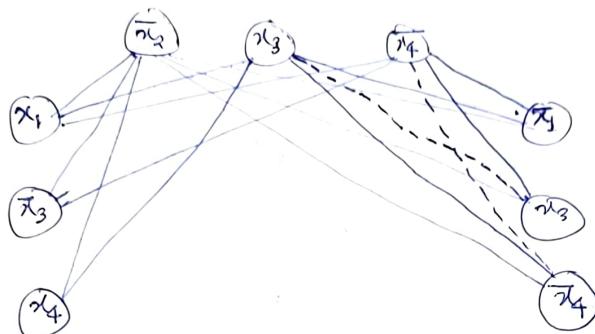
b) their corresponding literals are consistent. i.e. $l_i^n = \bar{l}_s^s$ (or when $l_i^n = l_s^s$ or $\bar{l}_i^n = \bar{l}_s^s$) (---)

This graph can be completed in polynomial time.

$G = (V, E)$ is 3-CNF satisfiable iff G has clique of size K .

- (\Leftarrow) i) for each literal, create a vertex.)
ii) Connect each literal x_i or \bar{x}_i to every other literal x_j or \bar{x}_j where $i \neq j$ in every other clause.
iii) Connect each $x_i \rightarrow x_i$ of other clause, $\bar{x}_i \rightarrow \bar{x}_i$ if other clause.

$$\text{e.g. } (x_1 \vee \bar{x}_3 \vee x_4) \wedge (\bar{x}_2 \vee x_3 \vee \bar{x}_4) \wedge (\bar{x}_1 \vee x_3 \vee \bar{x}_4)$$



Validation \rightarrow Suppose γ has a satisfying assignment.

Each clause C_k contains atleast 1 literal l^k that is assigned 1.

Such literal corresponds to vertex v_{l^k} .

From K clauses, K vertices v_{l^k} can be formed s.t $|V'| = K$.

Claim $\rightarrow V'$ is a clique.

For any 2 vertices $v_{l^k}, v_{l^s} - k \neq s$, corresponding literals l^k & l^s are mapped to 1 by satisfying assignment and thus literals cannot be complements. So edge $(v_{l^k}, v_{l^s}) \in E(G)$.

Conversely,

Suppose G has clique V' of size k .

V' contains exactly one vertex per clause as no edges in G connects vertices of same clause.

Assign 1 to the corresponding literal l^k if $v \in V'$ without worry of assigning 1 to both literal & its complement as no edges between inconsistent literals.

Each clause is satisfied and hence γ is satisfied.

$\therefore 3\text{-CNF} \leq_p \text{Clique}$.

(K) TSP is NPC

TSP → A salesman has to visit 'n' cities visiting each city exactly once. Cost is associated for travelling from city i to j . The salesman wants to complete the tour with minimum cost.

Decision Problem → Given graph $G_1 = (V, E)$, cost c is function from $V \times V \rightarrow \mathbb{Z}$, $K \in \mathbb{Z}$ does G_1 has a Travelling Salesman tour with cost at most K ?

NP Proof

$V(x, y)$:

$x \rightarrow$ instance of $G_1 = (V, E)$, K

$y \rightarrow$ Sequence of 'n' vertices, $V' = \{v_1, v_2, \dots, v_n\}$

$V(x, y)$: i) For each vertex in V' ,

check if any vertex repeats

return false

ii) For each pair of adjacent vertices $(v_i, v_j) \in V'$, $\forall i < j$ till $i=n-1, j=0$

$\text{sum} = \text{sum} + c(v_i, v_j)$
(initially)

if $\text{sum} \leq K$

return false

return true.

$V(x, y)$ can verify in polynomial time.

NPC Proof (Ham Cycle \rightarrow TSP)

Consider instance of the HAM cycle problem with graph $G = (V, E)$

Construct an instance of the TSP as below →

i) Form Complete graph $G' = (V, E')$, $E' = \{(i, j) : i, j \in V, i \neq j\}$

ii) Define Cost function $c(i, j) = \begin{cases} 0, & (i, j) \in E \\ 1, & (i, j) \notin E \end{cases}$

This construction can be done in polynomial time.

Claim → Graph G_1 has a Ham cycle iff graph G' has a tour of cost at most 0.

Validation →

Suppose G_1 has Ham-cycle h . Each edge in ' h ' belongs to E and has cost 0 in G' . Thus ' h ' is a tour in G' with cost 0.

Conversely,

Suppose graph G' has a tour ' h' ' of cost atmost 0. Since the cost of edges in E' are 0 or 1, the cost of tour ' h' ' is exactly 0 and each edge on the tour must have a cost 0. So, h' contains only edges in E .

Thus h' is a ham cycle in graph G_1 ,

∴ Ham Cycle \leq_p TSP.

② Longest simple cycle is NPC

Problem → Determining a simple cycle (no repeated vertices) of maximum length in a graph.

Decision Problem → Given graph $G_1 = (V, E)$, integer k does there exist a simple cycle of length atleast k ?

NP Proof

x : instance of graph $G_1 = (V, E)$, k

y : sequence of ' k ' vertices, $V' = \{v_1, v_2, \dots, v_k\}$

$V(x,y)$: i) For each vertex in V' ,

check if any vertex repeats

return false if YES.

ii) For each pair of adjacent vertices $v_i, v_j \in V'$, $i \neq j = i+1$,

check if edge $(v_i, v_j) \notin E$

till $j = k$

return false

return true

$V(x,y)$ takes linear time $O(\text{no. of vertices})$ or polynomial time.

NPC Proof. (Ham Cycle \rightarrow Longest simple cycle)

Consider the instance of Ham Cycle problem with graph $G = (V, E)$ and K .

Construct a new graph G' containing the same graph G , i.e., $G' = (V, C)$ with $|C| = |V|$.

Claim: If graph $G = (V, E)$ has a Ham cycle on $|V|$ vertices iff G' has a longest simple cycle of length K .

Assume G has a Ham cycle on $|V|$ vertices. So the cycle covers all the vertices of graph G exactly once. Consequently, graph G' will have a longest simple cycle of length K .

Conversely,

Assume G' has a longest simple cycle of length K . So the cycle covers all the $K = |V|$ vertices exactly once and thus G will have a Ham cycle on $|V|$ vertices.

Graph G' construction will take polynomial time.

(8) 3-Coloring is NPC

K-coloring Problem \rightarrow Assignment of colors to the nodes of a graph s.t. no two adjacent vertices have same color and atmost K colors can be used.

Decision Problem \rightarrow Given Graph $G = (V, E)$ and $K = 3$, can the graph be colored using atmost 3 colors s.t. no two adjacent vertices are assigned same color.

Proof NP

*: instance of graph $G = (V, E)$.

y : C is the list of colors in some order with vertices.
 $N(u, v)$: Assignment of colors $\{c_1, c_2, c_3\}$ where each vertex $v \in V$ is assigned one colour.
i) For each edge (u, v) in G , $u, v \in V$ & $u \neq v$
check if $color(u) \neq color(v)$
return false

return true.

$\rightarrow O(|E|)$ Polynomial.

NPC Proof

$(3\text{-SAT} \rightarrow 3\text{-Color})$

Consider the 3-SAT problem having 'm' clauses on 'n' variables. Let the variables be x_1, x_2, \dots, x_n .

Construct a graph from the formula as below—

- for every variable x_i , create vertex v_i and a vertex v'_i denoting \bar{x}_i .
- for each clause c in m , add 5 vertices j_1, j_2, \dots, j_5 .
- Create 3 special vertices denoting True, False and Base (T, F, B) .
- Edges are added among T, F, B to form a triangle.
- Edges are added among v_i and v'_i and B to form a Δ^e .

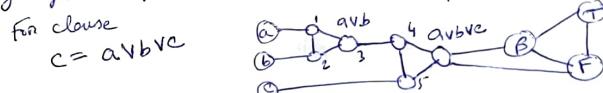
Constraints -



- For each pair of vertices v_i & v'_i , exactly one is assigned TRUE and other FALSE.
- For each clause c in ' m ' clauses, at least one of the literals in the clause is assigned TRUE.

, A OR-gadget graph can be constructed for each clause $C = (u \vee v \vee w)$ in the formula by i/p nodes u, v, w and connect output nodes of the gadget to both F & B special nodes.

Validation



Assume 3-SAT formula has satisfying assignment, then in every clause atleast one of the literals x_i has to be TRUE.

Then corresponding v_i can be assigned TRUE & \bar{v}_i FALSE.

Extending this, for each clause the corresponding OR-gadget graph can be 3-coloured. Hence, graphs can be 3-coloured.

Conversely,

If graph is 3-colourable, so if the vertex v_i is assigned TRUE, corresponding variable x_i is assigned TRUE. Hence, there is a satisfying assignment to 3-SAT clause.

$$\therefore \phi = c_1 \wedge c_2 \wedge \dots \wedge c_m = \text{TRUE}$$

Q1 ILP is NPC

D. Problem → Given an integer Matrix $A (m \times n)$ and an integer vector $b (m \times 1)$, does there exist an integer vector $x (n \times 1)$ with elements $\{0, 1\}$ s.t. $Ax \leq b$?

Proof NP

n: Matrix $A (m \times n)$, vector $b (m \times 1)$

g: Vector $x (n \times 1)$

$V(n; m) \leftarrow$ i) Compute $Ax = O(mn)$

ii) Compare elements of Ax and $b = O(m)$
if any element in $Ax > b$

return false

return true.

Verifier can verify the solution in $O(mn)$ i.e. polynomial time.

Proof NPC ($3\text{-CNF-SAT} \rightarrow 0\text{-1 ILP}$)

Consider the problem instance of 3-CNF having formula ϕ containing 'v' variables and 'c' clauses.

Construct $(c \times v)$ matrix A s.t

$$a_{ij} = \begin{cases} -1, & \text{if variable } j \text{ occurs } \overset{\text{only}}{\text{without negation}} \text{ in clause } i \\ 1, & " " " \text{ " " } \overset{\text{only}}{\text{with negation}} " " \\ 0, & \text{otherwise} \end{cases}$$

Construct vector $b (c \times 1)$ s.t

$$b_i = -1 + \sum_{j=1}^v \max(0, a_{ij}) \quad \text{no of negated literals in clause } i$$

Construction of A & $b \rightarrow O(Cl)$ polynomial time.

Claim — There exist vector $x (v \times 1)$ s.t. $Ax \leq b \Leftrightarrow \phi$ is satisfiable

Consider x to represent an assignment of variables in ϕ ,

$$x_i = \begin{cases} 1, & \text{if variable } i \text{ is assigned } \text{TRUE} \\ 0, & \text{if variable } i \text{ is assigned } \text{FALSE} \end{cases}$$

Let $y = Ax$, then

$y_i \leq b_i \Leftrightarrow$ sum of satisfied literals in clause $i \geq 1 \Leftrightarrow$
clause i is satisfied.

$\therefore y = Ax \leq b \Rightarrow x$ is an assignment satisfying ϕ .

(B) Independent Set is NPC

I-set \rightarrow Set S of graph $G_1 = (V, E)$ is a set of vertices s.t
no 2 vertices in S are adjacent to each other.
It consists of non-adjacent vertices.

problem \rightarrow Given a graph $G_1(V, E)$ and integer K , does there
exist an independent set of size $\geq K$?

No proof

x : instance of graph $G_1 = (V, E)$, K

y : set of vertices $\& S = \{v_1, v_2, \dots, v_k\}$

$V(x,y)$: i) if $|S| < K$
return false.

ii) for every edge in G_1 , check
 $(u, v), u \neq v$
if $u, v \in S$ (both vertices are in S)
return false

return true.

Proof NPC ($\text{Clique} \rightarrow \text{I-S}$)

Consider the clique problem consisting of graph $G_1(V, E)$ and
integer K .

Construct graph G'_1 as below $\rightarrow O(V+E)$
(V', E')

$V' = V$

E' = complement of edges E , i.e. edges not present in G_1 .

G'_1 is complement of graph G_1 .

Graph G_1 has a clique of size K iff G'_1 has IS of size K .

Validation \rightarrow

Assume, G has clique of size K . This implies there are k vertices in G where each vertex is connected by an edge with the remaining vertices $(K-1)$. Since these edges are in G , \therefore they are not present in G' . So, these K vertices are not adjacent to each other in G' and hence form an I.S of size K .

Conversely,

Suppose G' has I.S of size K . None of these vertices share an edge with any other vertices. Take complement of G' to obtain G where these K vertices will share edge with each other and hence form a clique of size K in G .