Week 07, ***Lecture: 13.***

Submitted by: Dhriti Prasanna Paul(M21CS056)

In Lecture 13, we have discussed about the solutions of the quiz 2 followed by Hall's Theorem, it's proof, vertex cover and some concepts related with it. Quiz 2 solutions: https://bit.ly/GTA_QUIZ2

1 Hall's Theorem

Statement: An X-Y bipartite graph G has a matching that saturates X iff
$|N(S)| >= |S| \qquad \forall S \subseteq X$

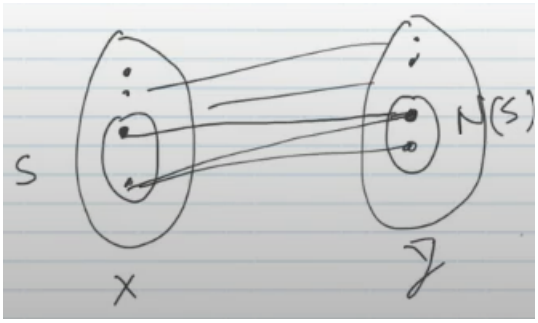


Figure 1: X and Y along with S and $|N(S)|$

[1]

Here,$N(S) \subseteq Y$is a set of neighbour of elements in S.

Proof:

Necessary conditions:-

Suppose X-Y bipartite graph has a matching that saturates X. Then obviously, $|S| <= |N(S)| \ \forall S \subseteq X$

Sufficient conditions:-

If $\forall S \subseteq X$, $|N(S)| >= |S|$

Then, there is a matching that saturates X.

We shall prove the following contrapositive:

If there is no such matching M that saturates X, then $\exists S \subseteq X$ such that $|S| >= |N(S)|$

Let u $\in X$ be a vertex unsaturated by a matching M.

Suppose two subsets S $\in X$ and T $\in Y$ are considered as follows:-

*1.*S is endpoint of M-alternating paths starting from U with the last edge belonging to M.

*2.*T is endpoint of M-alternating paths starting from U with the last edge not belonging to M.

[1]Figures used are the diagrams used demonstrated by Sir in class for easy better understanding.

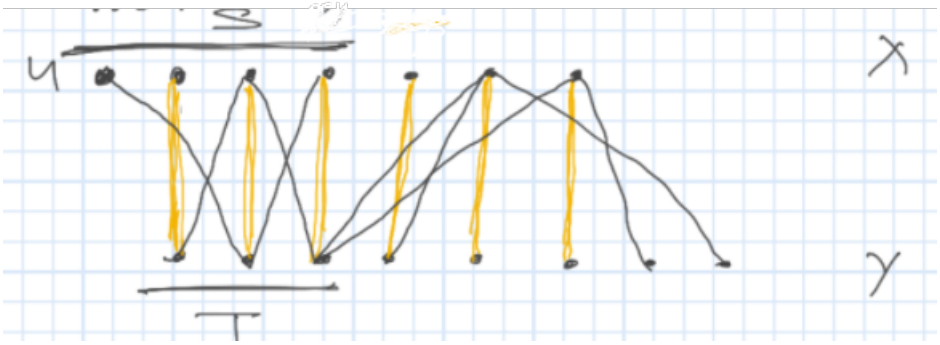


Figure 2: S, T, X, Y as shown above.

$|S| = 1 + |T| = 1 + N(S)$
$(S) > |N(S)|$

Degree Constant Graph :
If $\forall x \in X$, deg(x) <= d
If $\forall y \in Y$, deg(y) >= d

Vertex Cover :
A vertex cover of a graph G is a set $\theta \subseteq V(G)$ that contains atleast one end point of every edge.

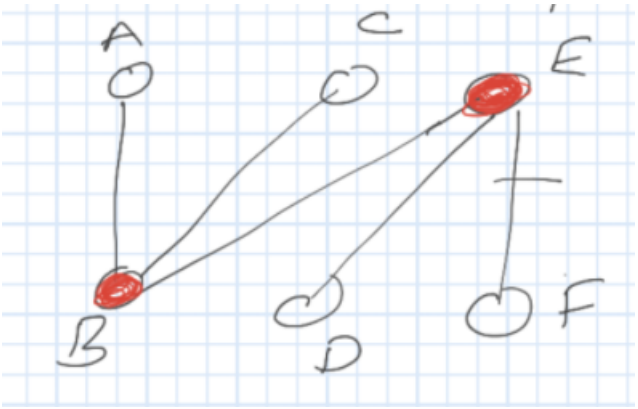


Figure 3: Vertex coverings

$\theta_1 = B, E$
$\theta_2 = A, C, E$

Independent Set :
Set of vertices in a Graph G, which are not adjacent to each other.
Independent Set $= A, C$

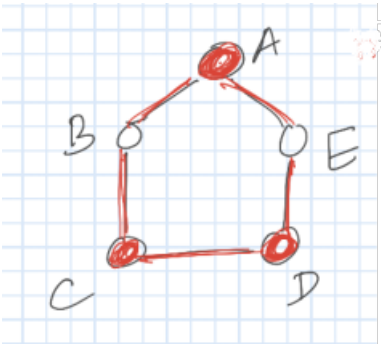


Figure 4: Independent Set

Some interesting important concepts :

$\alpha(G)$= Maximum size of Independent Set.
$\alpha'(G)$ = Maximum size of Matching.
$\beta(G)$= Maximum size of Vertex Cover.
$\beta'(G)$= Minimum size of Edge Cover.

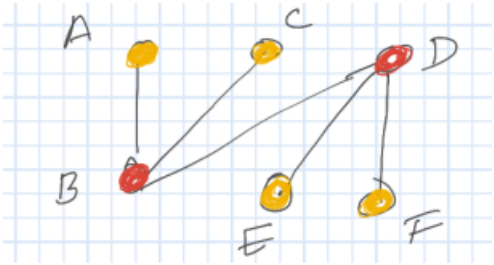


Figure 5: Illustration.

$\alpha(G)$= 4
$\alpha'(G)$= 2
$\beta(G)$= 2
$\beta'(G)$= 1

Week 07, ***Lecture: 14.*** Submitted by: Dhriti Prasanna Paul(M21CS056)

In Lecture 14, we have discussed about the Theorem 2.1.11 followed by certain proof

---

Theorem 2.1.11

If G is a simple graph, then if diam(G) >= 3 then diam(G^c)¡=3.

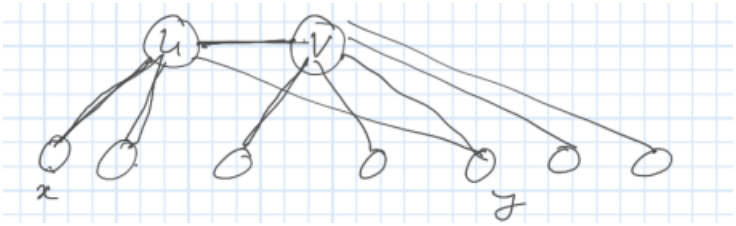
Proof :

diam(G)>= 3

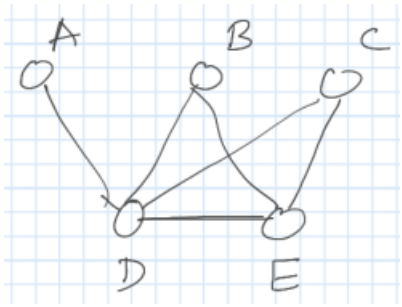
$\rightarrow \exists$ **u and v such that (i) uv** $\nexists E(G)$ **(ii) u and v donot have a common neighbour.**

$\forall$ **x** $\exists$ **V(G) has atleast one of u,v is non-neighbour.**

Since,
uv $\nexists$ **E(G)**
$\rightarrow$ **uv** $\exists$ **E(G**^{C}**)**
$ux'''\exists$ **E(G**^{C}**)**, $vx'''\exists$ **E(G**^{C}**)**
Below is the figure of G^C



$\dot{\alpha}(G) + \beta(G) = n(G)$
Proof:-
Let S be an independent set of maximum size, then every edge is incident to atleast one vertex of $\overline{S}$
$S = A, B, C$

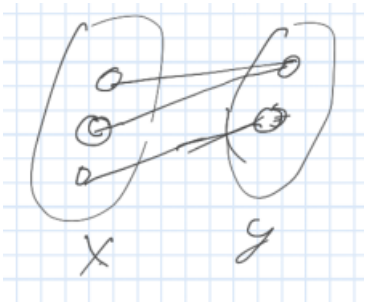


$\overline{S} = D, E$
S $\cup\overline{S} = n(G)$
$\overline{S}$ **covers all the edges.**
$\overline{S}$**is minimum size vertex cover.**
$\beta(G) = |\overline{s}|$
S is maximum size independent set.

$\alpha(G) = |S|$

$\beta(G) + \alpha(G) = |S| + |\overline{s}| = V(G) = n(G)$
Hence,
$\beta'(G) + \alpha'(G) = n(G)$
$\underline{\beta(G) + \alpha(G) = n(G)}$

If G is a Bipartite graph with no isolated vertices then $\alpha(G) = \beta'(G)$



$\beta(G) + \alpha(G) = n(G)$ ————**(i)**
$\beta'(G) + \alpha'(G) = n(G)$————**(ii)**
$\alpha'(G) = \beta(G)$————**(iii)**
$\beta(G) + \alpha(G) = \beta'(G) + \alpha'(G) = \beta'(G) + \beta(G)$
$\alpha(G) = \beta'(G)$

Let G be a Bipartite graph. Prove that $\alpha(G) = n(G)/2$ **if G has perfect matching**
We know,
$\beta(G) + \alpha(G) = n(G)$
$\alpha(G) = n(G) - \beta(G) = n(G) - \alpha'(G)$
If G has Perfect Matching, than the maximum size of the matching would be :
= n(G) - n(G)/2.
= n(G)/2.

Week 08, Lecture: 15. Submitted by: Dhriti Prasanna Paul(M21CS056)
In Lecture 15, we have discussed about the Chinese Postman Theorem followed by Hungarian Algorithm. We also find the time complexity associated with it.

Chinese Postman Theorem

It is a type of Eulerian circuit problem for an undirected graph.
An Euler Circuit is a closed walk that covers every edge once starting and ending position is same.

Our aim is to find the shortest path that visits every edge of the graph at least once.

Here, we have two cases depending on the degree of the nodes.

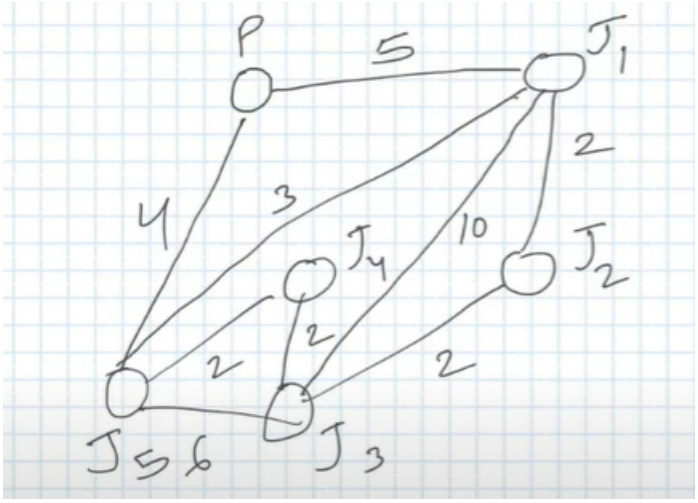


Figure 6: Example having all even degree.

Let us consider the graph G having vetex V and edge E.
In the diagram we can directly apply and find out the minimum cost, since all the nodes have even degree here.

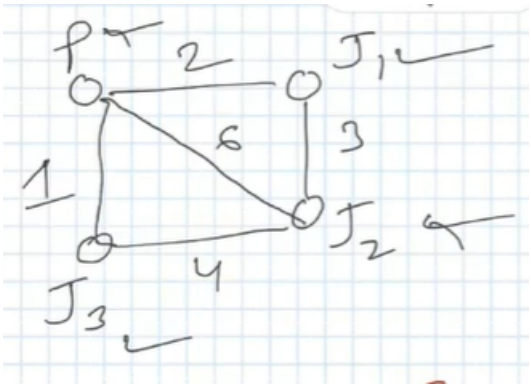


Figure 7: Graph having odd degree vertices.

Case 1:
$\forall V_i \in V(G)$
$deg(V_i)$ **is even.**

Case 2:

$\exists V_i \in V(G)$
$deg(V_i)$ **is odd.**

In the figure 7, we can see that the vertices J_1 **and** J_3 **are even,** P **and** J_2 **are odd.**

To make the odd degrees even, we can add additional edges.The edges can be added as shown below in three ways, (A), (B), (C), and then calculate the cost.

The time complexity is $O(n^3)$

It is because, all nodes must be covered but the edges must be covered at max once.

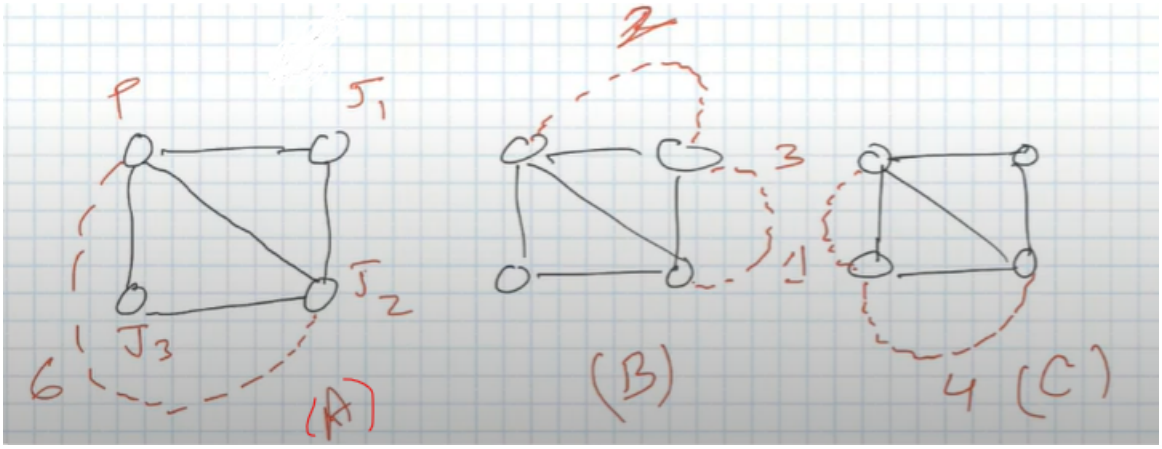


Figure 8: Making the odd vertices even by adding edges.

Weighted Bipartite Matching.

Here, we take a matrix which shows us the requirement and we need to find the minimum possible value.

	S_1	S_2	S_3	S_4
C_1	5	10	15	20
C_2	3	5	6	9
C_3	9	8	1	15
C_4	15	16	1	5

In the matrix above, S_i **represents the sites present and** C_i **represents the cranes time required in the specific sites.**

From the matrix we can see, which crane takes which time.

Traversals

A traversal of a N*N matrix consists of N positions one in each row and one in each column.

Finding manually the least time/effort/requirement in a brute force approach would cost us n! number of traversals.So, we use

a method called Hungarian method which helps us find the in polynomial time.

Assignment problem as optimisation:

min $\sum_i \sum_j C_i j X_i j$ **such that**
$\sum i\mathbf{X}_i j$ **= 1**
$\sum j\mathbf{X}_i j$ **=**
$X_i j$ **0,1;** $X_i j >= 0$

Hungarian Algorithm :

The Hungarian method is a combinatorial optimization algorithm that solves the assignment problem in polynomial time and which anticipated later primal–dual methods.

The time complexity is order of n^3

Let us consider the below matrix for better understanding sing an example.

$$\begin{bmatrix} 40 & 60 & 15 \\ 25 & 30 & 45 \\ 55 & 30 & 25 \end{bmatrix}$$

(i) We find the row minimum. (15, 25, 25)

(ii) Subtract row minimum from every element.The matrix becomes:-

$$\begin{bmatrix} 25 & 45 & 0 \\ 0 & 5 & 20 \\ 30 & 5 & 0 \end{bmatrix}$$

(iii) Find column minimum.(0, 5, 0)

(iv) Subtract column minimum from every element.The matrix becomes:-

$$\begin{bmatrix} 25 & 40 & 0 \\ 0 & 0 & 20 \\ 30 & 0 & 0 \end{bmatrix}$$

Now , we find the no of lines which can cover the zeroes present.

If,

no of lines = n; we stop here.

Total cost = 25+30+15 = 70.(optimal)

else,

We find the minimum of uncovered values and add the value at intersection point.This would than make it similar to the above case i.e.no of lines(lines required to cover all zeroes) = n.

Thank you.