Week 6

Lecture-11& Lecture 12 : Distance in Graph and Spanning Tree,Matching
*

The class notes contains proving the theorem with respect to distances in graph and later covers spanning trees and matching.

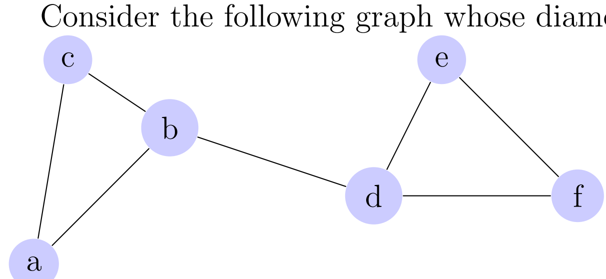
6.1 Distance in Graphs- Contd..

Theorem 6.1. *To prove that G is a simple graph, diameter of G* $dia(G) >= 3$ *implies =>* $dia(G^c) <= 3$

Proof. (i) Consider a graph with $dia(G) >= 3$, then there exists two vertices(say u,v) which belong to V(G) such that $dia(G)(u, v) = 3$ and hence they are not adjacent.

(ii) Vertices u,v have no common neighbhour=> edge uv doesnot exist for any vertex say x belongs to $V(G) - u, v$;

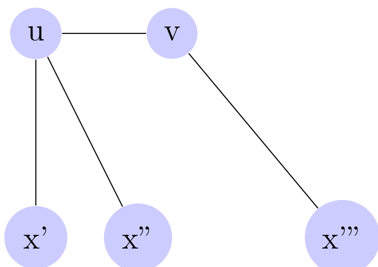
Consider the following graph whose diameter is 3



*Lecturer: Anand Mishra . Scribe: Kalpana Vankayala MT19AIE250.

(a) when edge ux is existing , vx will not exist and vice versa as they donot share a common adjacent vertex as per above .

(b) ux and vx both donot exist.
In G^c uv exists length is 1. In the complement graph ux and vx are mutual exclusive in existing or both will exist in G^c where diameter is atmost 2.For all other vertices, the maximum eccentricity is 3. Therefore, the diameter of complementary graph $dia(G^c) <= 3$.The following graph can be used for above explanation.



∎

Theorem 6.2. *Let G be a simple graph, diameter of G $dia(G) >= 4$ Prove that the diameter of the complement graph is $dia(G^c) <= 2$*

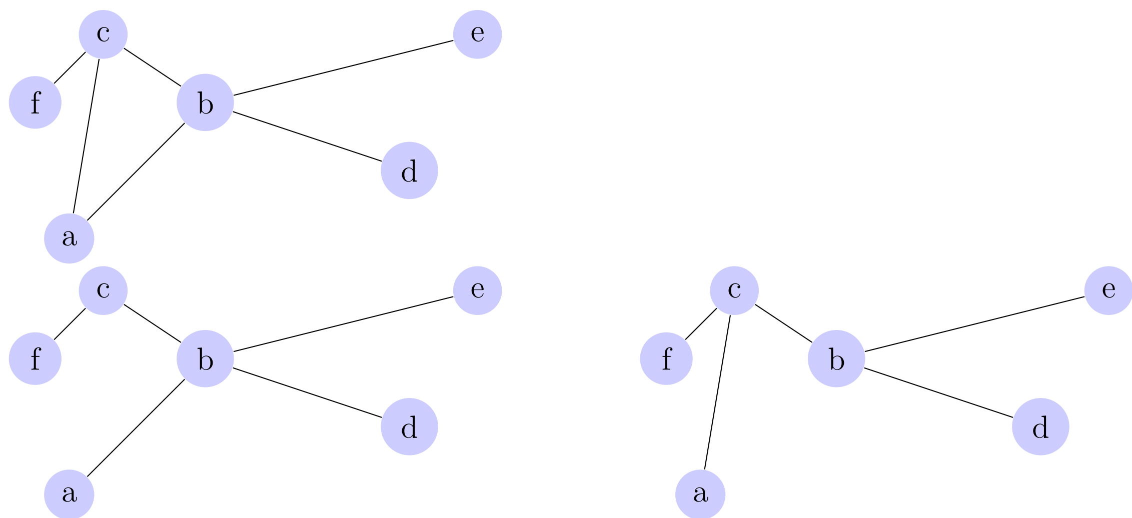
Proof. We have to prove that P implies Q $P => Q$ where P: $diam(G) >= 4$; Q: $diam(G^c) =< 2$

The equivalent to above is the contrapositive that is Q => P ; Q: $diam(G^c) >= 3$ and P: $diam(G) <= 3$
As seen the theorem 6.1, we know that in a simple graph, the $diam(G) >= 3$ then $diam(G^c) =< 3$.
We take complement of $diam(G^c) >= 3$ then $diam((G^c)^c) =< 3$.
which is $diam(G^c) >= 3$ then $diam(G) =< 3$
Hence the statement $Q =>$ P holds and hence the initial $P => Q$ is proven.

∎

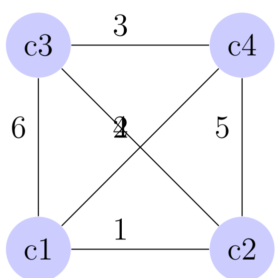
6.1.1 Spanning Tree

Spanning tree of a graph is a tree which connects all the vertices with minimum possible edges.A spanning tree is connected and doesnot contain a cycle.For example consider the following graph. The spanning trees would be as follows



6.1.2 Minimum spanning Tree(MST)

A minimum spanning tree or minimum weight spanning tree is a subset of the edges of a connected, edge-weighted undirected graph that connects all the vertices together, without any cycles and with the minimum possible total edge weight.



We need to find the minimum weighted path between the vertices. The different methods are as follows.

Prim's Algorithm

This is greedy algorithm to find the minimum cost path of the tree. The steps are as follows.

- Initialize a tree with a single vertex, any vertex can be choosen.
- Grow the tree by one edge: of the edges that connect the tree to vertices not yet in the tree, find the minimum-weight edge, and transfer it to the tree.
- Repeat step 2 until all vertices are exhausted in the tree.
- the complexity of this algo is of the order of $O(V^2)$)where V is the number of vertices.

Krushkal's Algorithm

Start with the edges instead of the nodes, sort the edges in ascending order,Keep on choosing the edges and do not choose an edge that forms the cycle. The complexity in this algorithm is of the order of $O(Elog(E)$ Also if the node is already visited then we skip

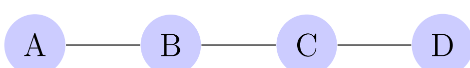
- Initialize a tree with a single vertex, any vertex can be choosen.
- Grow the tree by one edge: of the edges that connect the tree to vertices not yet in the tree, find the minimum-weight edge, and transfer it to the tree.
- Repeat step 2 until all vertices are exhausted in the tree.

6.1.3 Application of MST: Image Segmentation

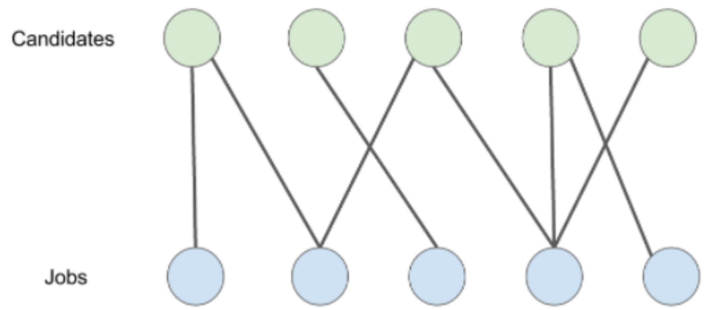
Efficient graph based segmentation was authored by Felzenszwalb et al using the minimum spanning tree based graph method for Image segmentation. In image segmentation,it is required to find out the boundary between two different regions.In this paper authors propose a method, based on selecting edges from a graph, where each pixel corresponds to a node in the graph, and certain neighboring pixels are connected by undirected edges.Weights on each edge measure the dissimilarity between pixels.Then it calculates the intensity differences across the boundary and also intensity differences in the local neighbouthood.The method is fast as compared to other methods using eigen vectors and also captures non local image characteristics and computationaly efficient as it is a graph based method.

6.2 Matching and Covers

Matching is a set of non loop edges with no shared end points.
For example: In the following graph



Matching M1 = ”{”AB,CD”}” ; M2 = ”{”BC”}”
M1 is called the perfect match as all the nodes are exhausted or included.
Some of the applications of matching are in flow networks,scheduling and planning,modelling chemical bonds,graph coloring.
A simple use case for matching is in matching jobs to candidates. A group of candidates and a set of jobs, and each candidate is qualified for at least one of the jobs. We can use graph matching to see if there is a way we can give each candidate a job they are qualified for.



6.2.1 Maximum and Maximal Matching

Maximum Matching:

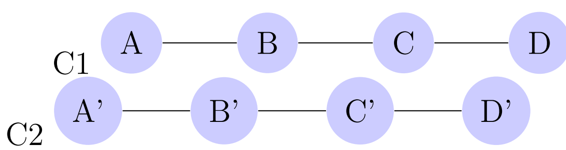
The matching of the graph which has the maximum size than any other matching is called the maximum matching. For example in the above graph,
M1 = ”{”AB,CD”}” is the maximum matching.

Maximal Matching:

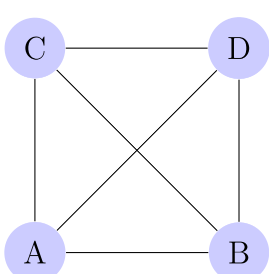
The matching of the graph when no other edge can be matched is the maximal matching. In the above example M2 = ”{”BC”}” is the maximal matching as we cannot add AB or CD as they are linked to B or C.

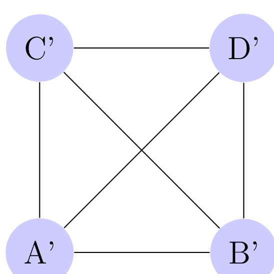
Perfect Matching

If there are K components in a graph, the number of perfect matching is
#PM of graph = #PM c1 * #PM c2 *....#PM ck



C1 And C2 are components of the graph, then #PM c1 = 1 ; #PM c2 = 1
then #PM of the graph is 1*1 = 1



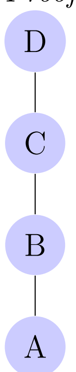


In the above example, the perfect matching of component 1 is 3 and the perfect matching of component 2 is 3. Hence the perfect matching of graph G with both components is 3*3 = 9

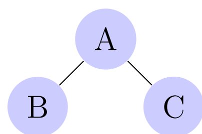
6.3 Lecture-12: Matching Contd..

Theorem 6.3. *Prove that every tree has atmost one perfect matching.*

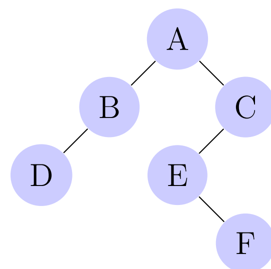
Proof. Consider the following trees



A

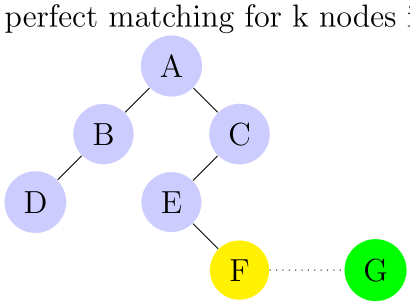


B

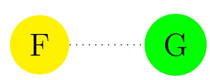


C

The number of perfect matching #PM(A) = 0, #PM(B) = 1, #PM(C) = 1,
Proof by Induction:
Base condition: A tree having n=1 node has #PM = 0 and n=2 has #PM = 1.
Induction Hypothesis: Suppose the above condition is true for $n <= k$ nodes.Therefore, the perfect matching for k nodes is less than or equal to 1.



C1



C2

Add one node (g) and check the condition. So the nodes are now k+1. remove one node from the k node graph, still the number of perfect matching less than or equal to 1#PM C1 <= 1.
Now the k node and k+1 node form an edge and the #PM C1 <= 1. Therefore the two components have Perfect matching equal to 1. By the rule of perfect matching of components of a graph, multiply PM number of each component
#PM = #PM C1 * #PM C2 = 1* 1 = 1. Hence #PM <= 1 is proved.

□

6.3.1 Hall's Marraige Theorem- Introduction

Theorem 6.4. *An X-Y bigraph G has a matching that saturates X if and only if* $|N(s)|$ *¿=* $|S|$*. Cardinality of* $|N(s)|$ *¿= cardinality of* $|S|$ *for every S subset of X. Here N(s) is a subset of Y which is a set of neigbhours of elements in S*

End of Class