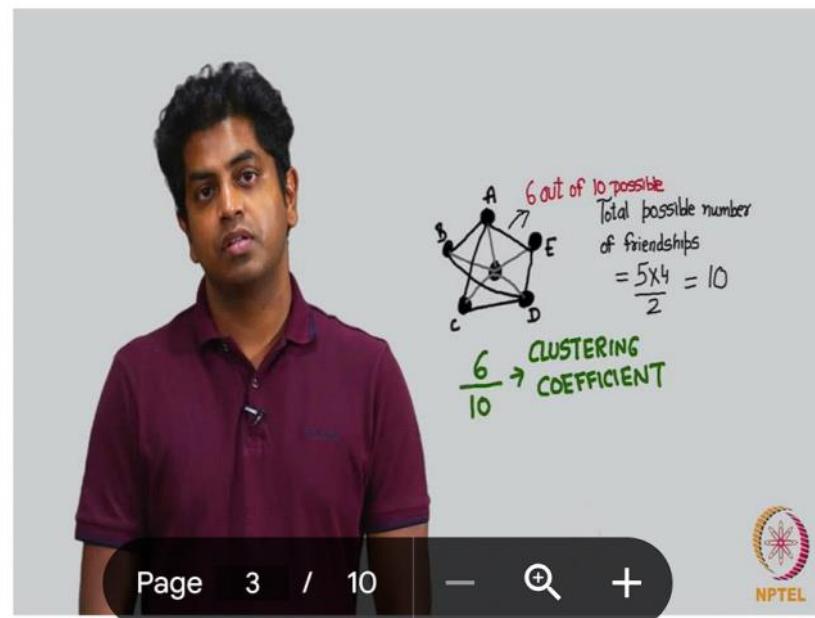
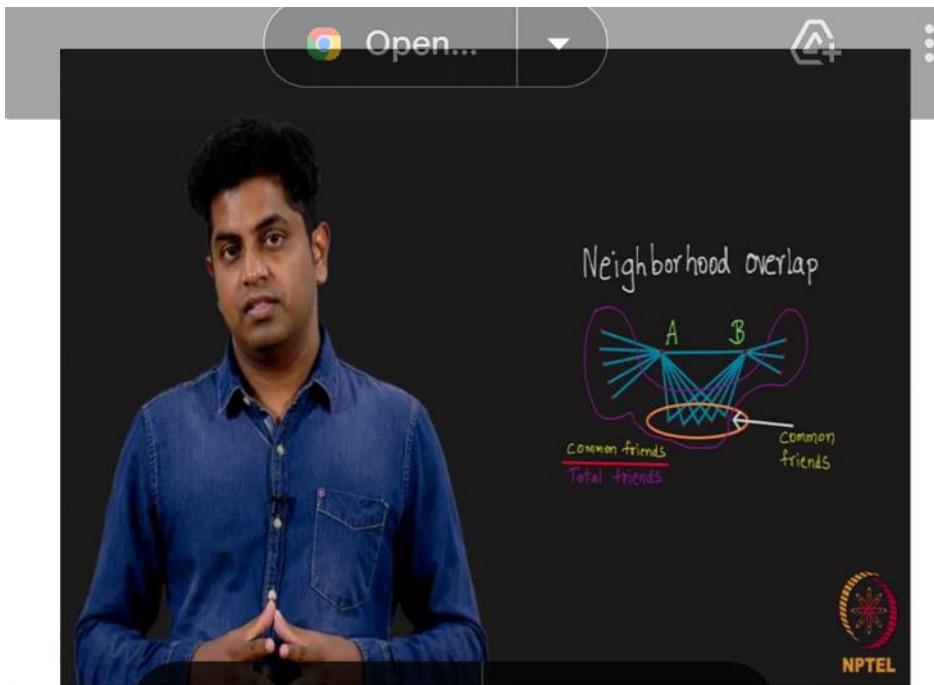


Week 3 onwards :



How close your friends are : measured by clustering coefficient !!

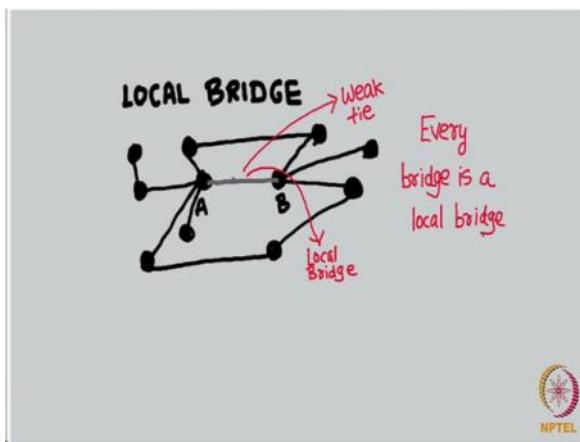


 ChatGPT

Neighborhood overlap refers to a concept in social network analysis that describes the degree to which individuals within one person's social network are also connected to each other. It measures the extent to which the social circles of two or more individuals intersect or share common connections.

In simpler terms, if you have two people in a social network and you examine their immediate connections (their friends, family, colleagues, etc.), neighborhood overlap would indicate how many of those connections are shared between the two individuals.

Neighborhood overlap is a useful metric for understanding the structure and dynamics of social networks. High neighborhood overlap suggests a strong tie or close relationship between individuals, as they tend to share common connections and likely interact within similar social circles. On the other hand, low neighborhood overlap indicates more distinct social circles or less interaction between individuals' networks.



Local bridge means : a-b edge but no other c exists such that : a-c and b-c edge is there !!

Local bridges are : weak ties !! { vvimp !!}

A local bridge is mostly a weak tie //

Page 6 / 7

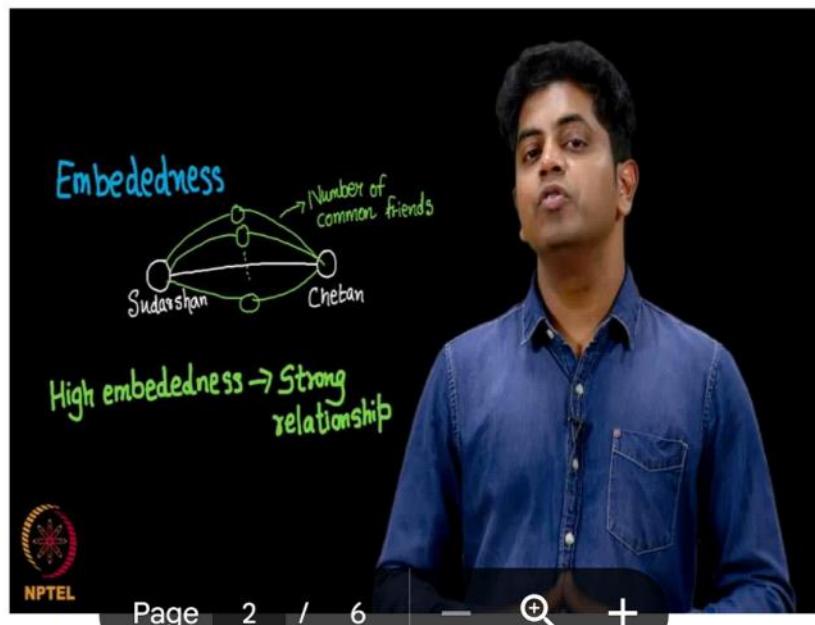


NPTEL

Page 3 / 4

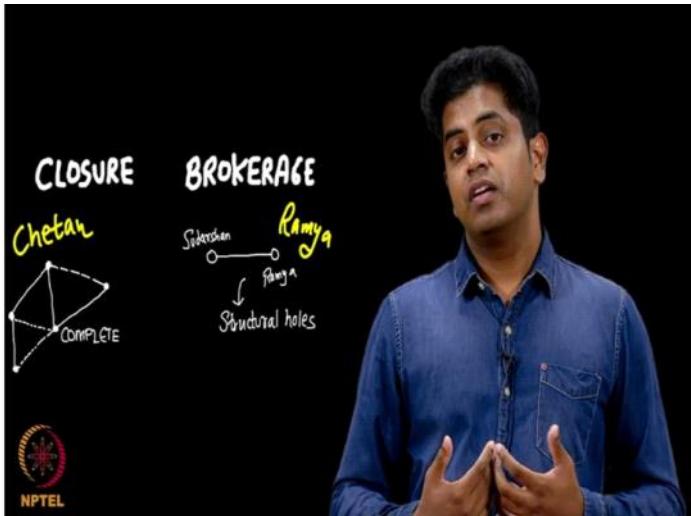
0 neighbourhood overlap is the definition for : local bridge !!

NPTEL



How low embeddedness can be useful in some situations !!

WHAT IS SOCIAL CAPITAL ?



Closure means friends friend should become friends, when a friends friend become is a friend the network completely become sort of complete everybody will be friends with each other is not it. Here is a friends friend you become friends with him, here is a friend friends you become friends with him so on and so forth you see this sort of friends friend becoming a friend cascades and finally, everyone will become friends and there is no fun here, but is closure. Brokerage means there should be an edge, as I told you in my previous lecture me and Ramya. Ramya acts as a bridge here she benefits with the fact that I am we do not have any common friends, it is not easy for me to reach the other side of the community right Ramya is the bridging gap.

So, there has to be what is called the structural holes two at the same time some closures should also happen. As an within the previous example me and Chethan were good friends we had a lot of common friends, we benefit with that kind of relationship me and Ramya are of course friends, but it is a sort of a local bridge it is a bridge she is a bridge for the for me to connect to the other community that is also important brokerage is important closure is also important. In general the bigger question is given a choice for you to come out with a nice network, I will give you 100 people and I will give you a 1000 friendships and ask you place this friendships properly so that this particular 100 friends maximum yield maximum benefit if they work in a team.

THE UNHIGHLIGHTED PART ABOVE IS WHAT CALLED AS DEFINATION OF SOCIAL CAPITAL !! EG : FACEBOOK FRIEND SUGGESTIONS TO KEEP US HOOKED !

FINDING 2 COMMUNITIES USING BRUTE FORCE :

-> then find this ratio : no of inter edges / total intra edges !!
the one with least value of it take that combination !!

Strong and Weak Relationships Community Detection Using Girvan Newman Algorithm :

What is edge betweenness value of an edge ?

ChatGPT

Edge betweenness centrality is a measure used in network analysis to quantify the importance of an edge (connection) in a network based on the number of shortest paths between pairs of nodes that pass through that edge. In other words, it measures how frequently an edge serves as a bridge connecting different parts of the network.

Edge betweenness centrality is useful for identifying critical edges in a network that act as bridges between different communities or clusters. Removing edges with high betweenness centrality can disrupt the flow of information or influence within the network, potentially leading to fragmentation or the formation of distinct subgroups.

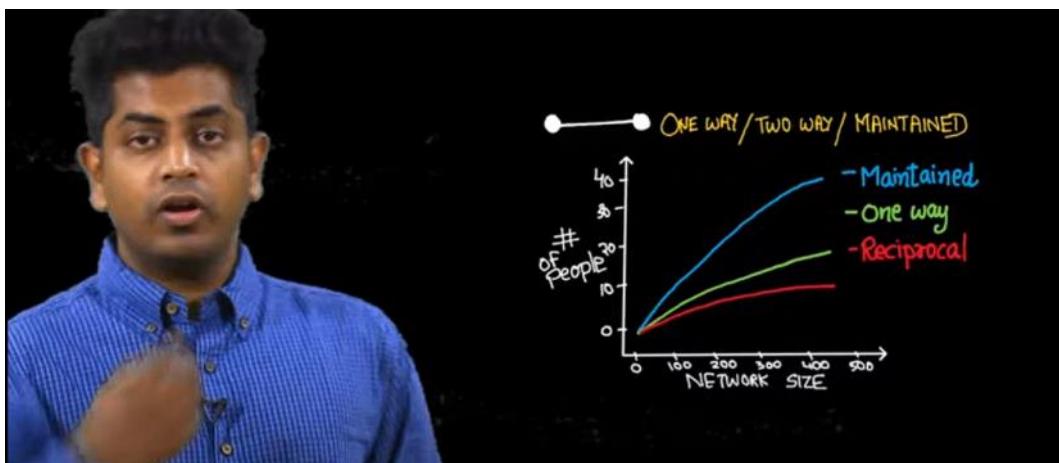
EVERY TIME YOU REMOVE THE EDGE WITH MAX EDGE BETWEENNESS TILL THE TIME >1 COMMUNITIES ARE NOT FORMED !!



Passive engagement refers to a level of involvement or participation where an individual or group is present or exposed to a particular activity, event, or content but does not actively contribute or interact. It often involves observing, listening, or consuming information without actively engaging in dialogue, interaction, or action.

In various contexts, passive engagement can take different forms:

1. **Social Media:** On social media platforms, passive engagement may involve scrolling through a feed, viewing posts, photos, or videos, without liking, commenting, or sharing content. Users passively consume information without actively participating in conversations or contributing their own content.



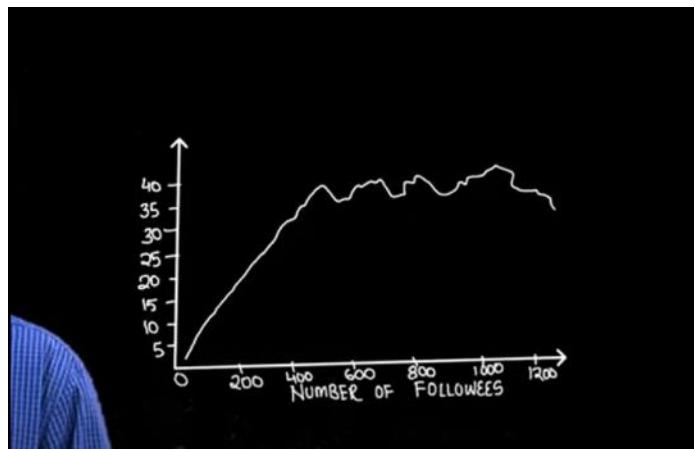
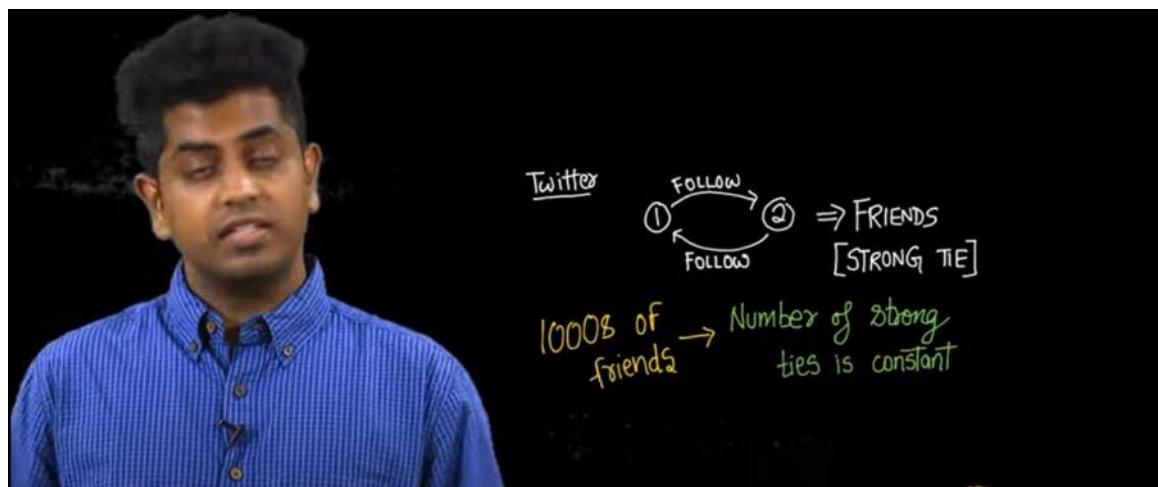
Maintained relationships : also called as passive relationships !!

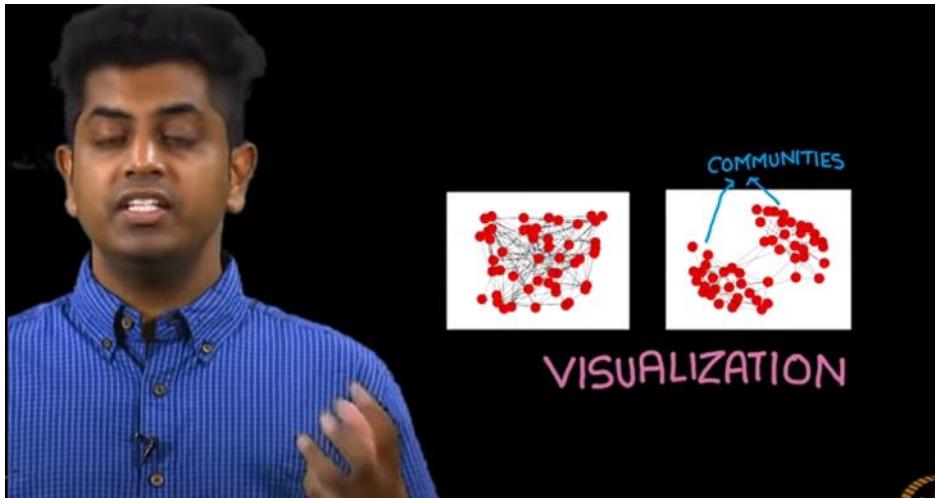
L You
tell wrt social media

ChatGPT

In the context of social media, passive engagement refers to behaviors where users consume content or observe interactions without actively participating or contributing to the conversation. Here are some common examples of passive engagement on social media platforms:

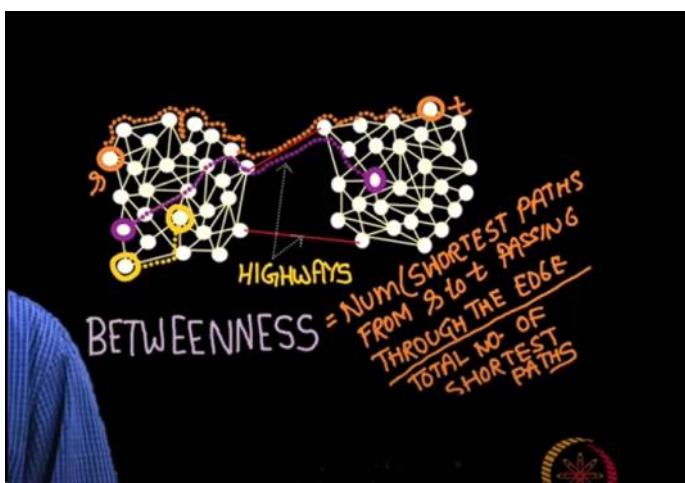
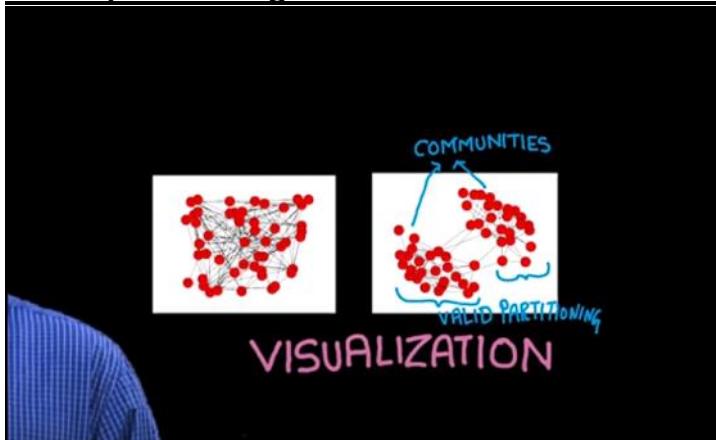
1. **Scrolling Through Feeds:** Users scroll through their social media feeds, viewing posts, photos, and videos without actively engaging with the content by liking, commenting, or sharing.
2. **Lurking:** Lurking refers to the practice of silently observing conversations or interactions on social media platforms without actively participating or revealing one's presence. Lurkers may follow discussions in groups or forums without contributing their own comments or reactions.
3. **Consuming Content:** Users consume content shared by others, such as articles, videos, or memes, without actively engaging with the creators or providing feedback.
4. **Passive Following:** Users follow accounts or pages on social media to stay updated on news, trends, or updates from specific sources without actively engaging with the content or interacting with the account owner.
5. **Silent Observers:** Users may be active on social media platforms but prefer to remain silent observers, choosing not to express their opinions, thoughts, or reactions publicly.





Communitites are group of nodes having a lot of intra edges and a low number of inter edges !!

Such a partitioning is called : VALID PARTITIONING !!



For btwness : I take sum of all these fractions for all s and t { for a given e}

L You
What is homophily ?

ChatGPT
Homophily is a social phenomenon referring to the tendency of individuals to associate and bond with others who are similar to themselves in certain attributes, such as demographic characteristics, beliefs, attitudes, interests, or behaviors. The term "homophily" comes from the Greek words "homo" (meaning "same") and "philia" (meaning "love" or "liking"), indicating a preference for similarity in interpersonal relationships.

What are social influence and selection ?

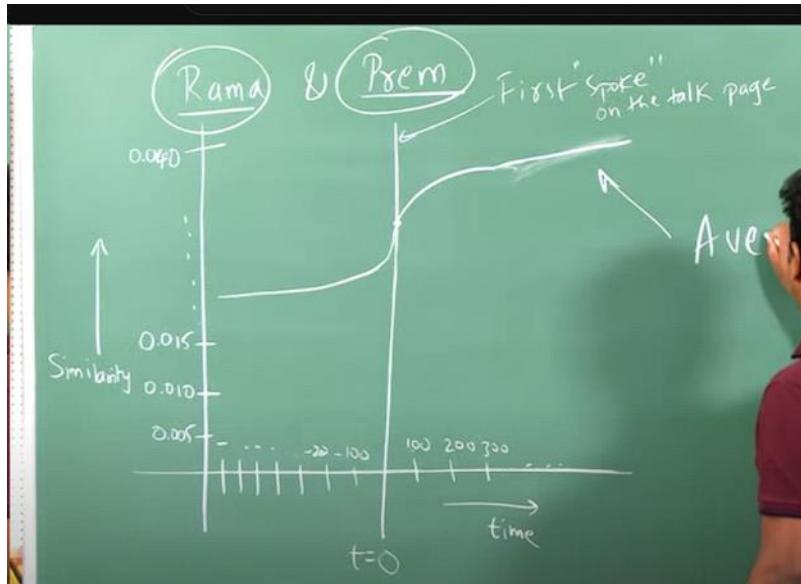




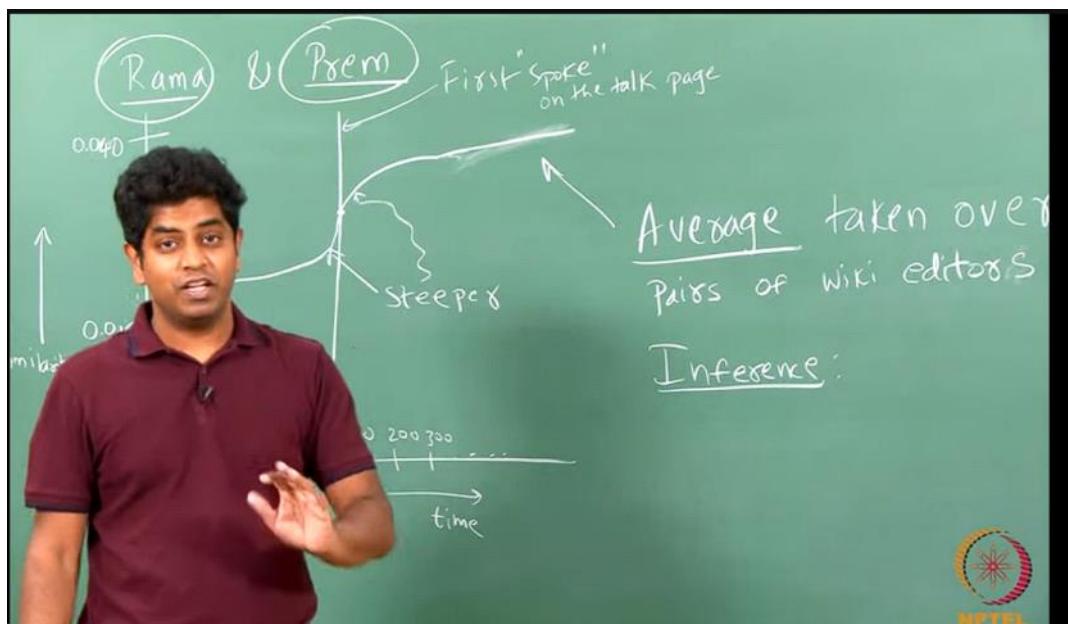
WHAT IS SIMILARITY MEASURE ?

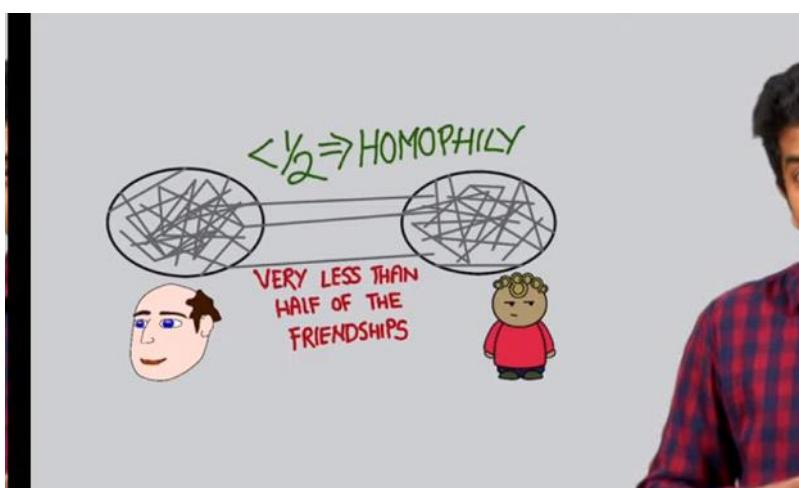
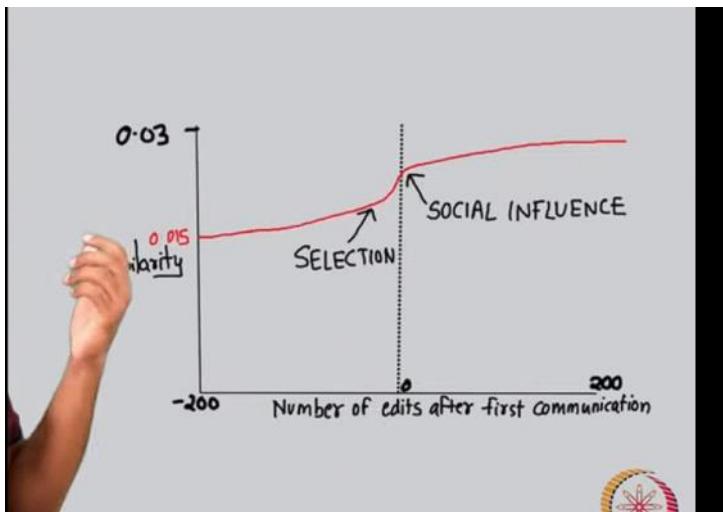


{ SOMETHING SIMILAR TO : NEIGHBOUR HOOD OVERLAP }

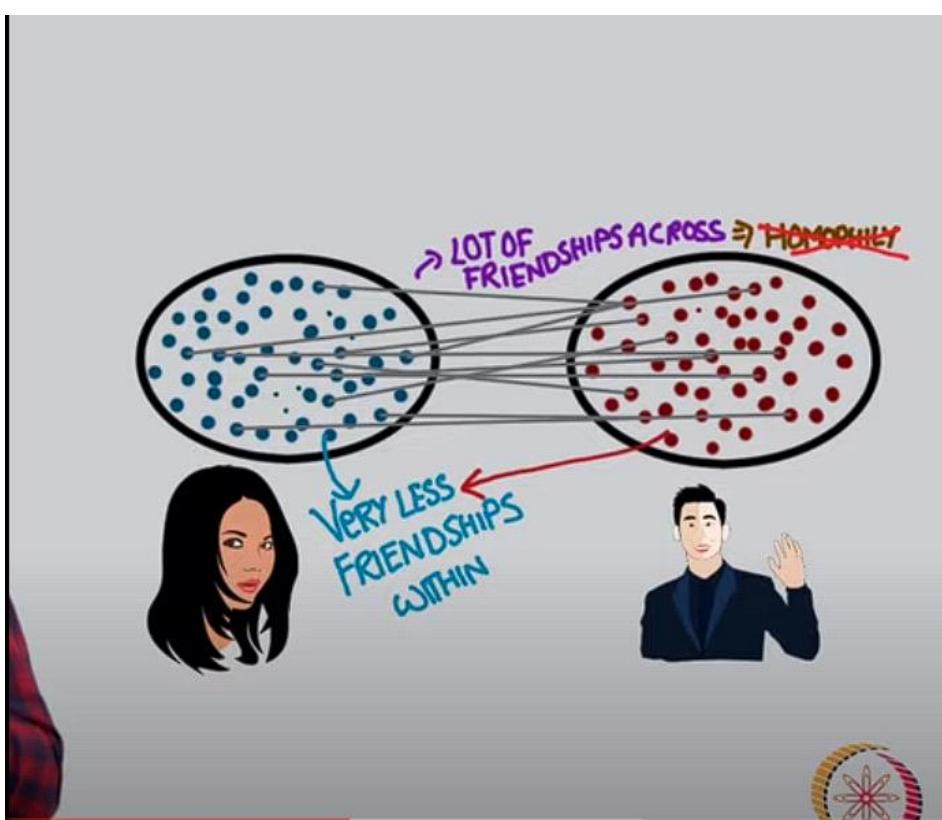


Average taken over all possible pairs of wiki editors who spoke to each other





If total no of friendships btw dis similar groups / total friendships < 1/2 then it is called homophily !!!





$$1 - \frac{\text{NUMBER OF FRIENDSHIPS PRESENT (3)}}{\text{EXPECTED NUMBER OF FRIENDSHIPS (9)}}$$

$$= 1 - \frac{\text{Actual}}{\text{Expected}}$$

Formula for homophily !!



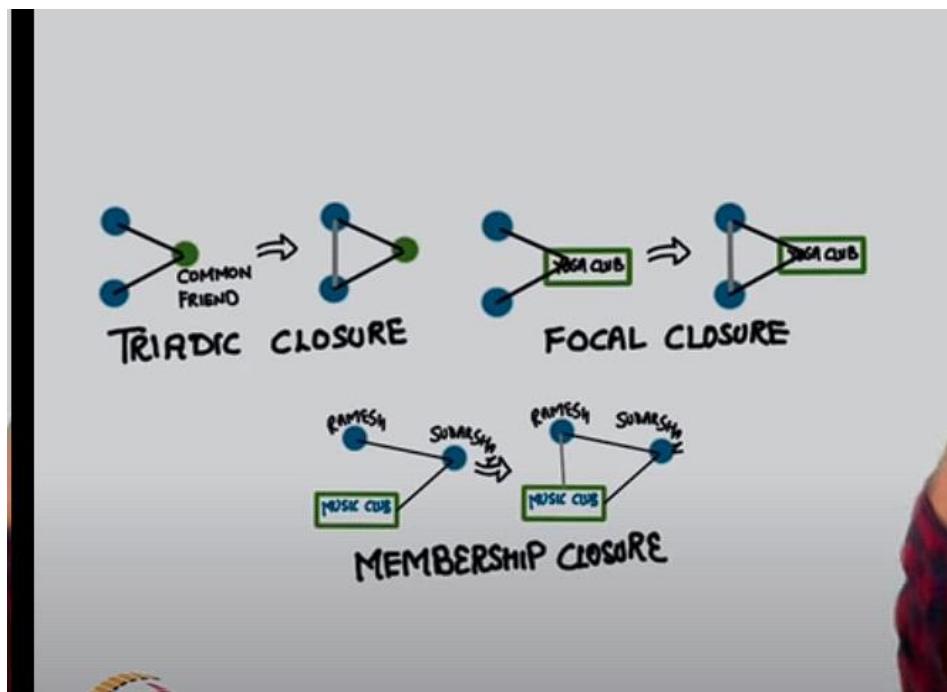
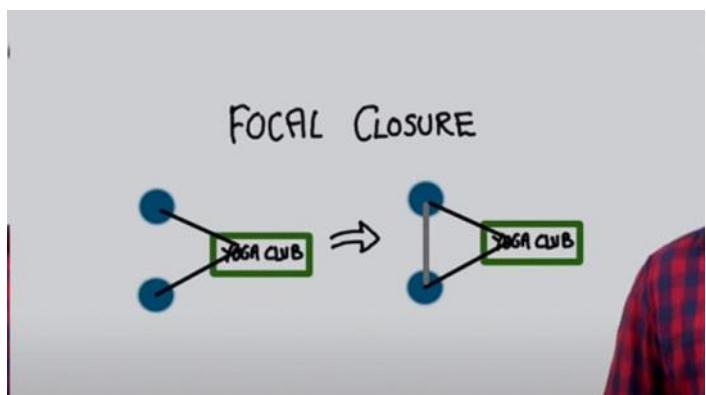
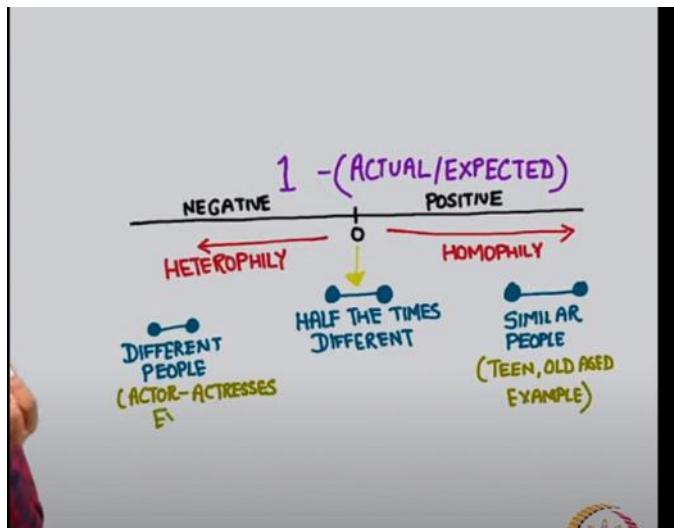
$$1 - \frac{\text{Actual (3)}}{\text{Expected (9)}}$$

$= \text{Zero}$
 $(\text{ACTUAL} = \text{EXPECTED})$
 $\Rightarrow \text{NO HOMOPHILY}$

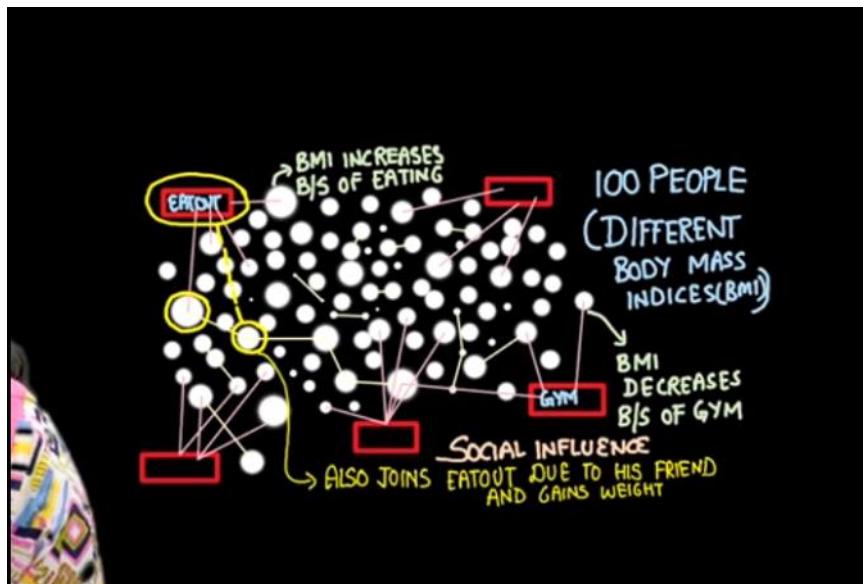


$$1 - \frac{\text{Actual (3)}}{\text{Expected (9)}}$$

\Rightarrow NEGATIVE
 (ACTOR ACTRESSES)
 $\Rightarrow \text{HETEROGENEITY}$



Fatman Evolution Model : { Coding }



```
*evolutionary_model.py -: gedit
  File  Open  Save  Undo  V  Python  9:11AM 0
  *evolutionary_model.py *
dict1={}
for each in G.nodes():
    dict1[each]=G.node[each]['name']
return dict1

def get_size(G):
    array1=[]
    for each in G.nodes():
        array1.append(G.node[each]['name'])
    return array1

G=create_graph()
assign_bmi(G)
labeldict=get_labels(G)
nodesize=get_size(G)

visualize(G,labeldict,nodesize)
```

Python • Tab Width: 8 • Ln 42, Col 31 INS

```
evolutionary_model.py -: gedit
  File  Open  Save  Undo  V  Python  9:12AM 0
  * evolutionary_model.py *

def visualize(G,labeldict,nsizel):
    nx.draw(G,labels=labeldict,node_size=nsizel)

    plt.show()

def assign_bmi(G):
    for each in G.nodes():
        G.node[each]['name']=random.randint(15,40)
        G.node[each]['type']='person'

def get_labels(G):
    dict1={}
    for each in G.nodes():
        dict1[each]=G.node[each]['name']
    return dict1
```

Saving file '/home/yayati/evolutionary_model.py...' Python • Tab Width: 8 • Ln 15, Col 31 INS

L You

What is social foci ?

0

ChatGPT

"Social foci" refers to specific points of reference or focal points within a social network where interactions, connections, or activities converge. These foci can be individuals, groups, organizations, events, locations, or topics around which social interactions revolve or are organized.

In the context of social network analysis, social foci play a significant role in shaping the structure and dynamics of social networks by influencing patterns of interaction, information flow, and the formation of social ties. Here are some examples of social foci:

1. **Individuals:** Influential individuals within a social network, such as leaders, opinion-makers, or trendsetters, can serve as social foci around which others gather or interact. These individuals often play a central role in facilitating communication, collaboration, or influence within the network.
2. **Groups and Communities:** Social groups, communities, or organizations can act as social foci that attract members and facilitate interactions among them. These groups may be based on shared interests, identities, affiliations, or goals, and they provide a platform for members to connect, communicate, and collaborate.
3. **Events and Activities:** Social events, gatherings, or activities serve as temporary social foci that bring people together for shared experiences or purposes. Examples include parties, conferences, festivals, or sports events, where individuals converge to socialize, network, or engage in common interests or activities.
4. **Locations and Spaces:** Physical locations or virtual spaces can act as social foci where people gather or interact. This includes public spaces such as parks, cafes, or markets, as well as online platforms such as social media sites, forums, or virtual communities.
5. **Topics and Discussions:** Specific topics, issues, or discussions can serve as social foci around which individuals gather to exchange ideas, share information, or express opinions. This includes online forums, interest-based communities, or discussion groups focused on particular themes or subjects.

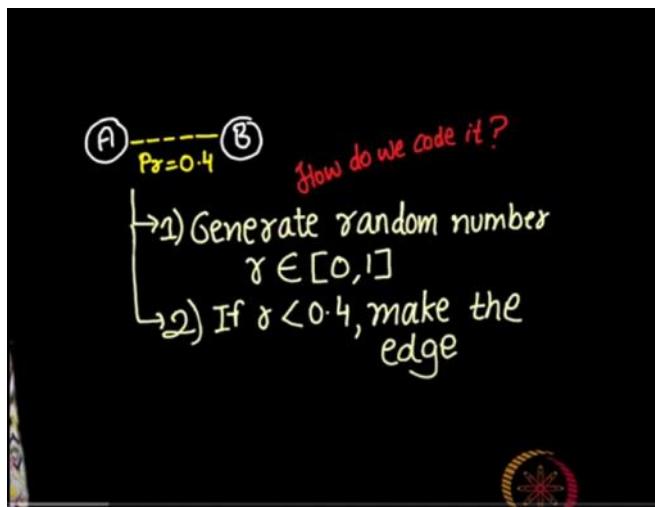
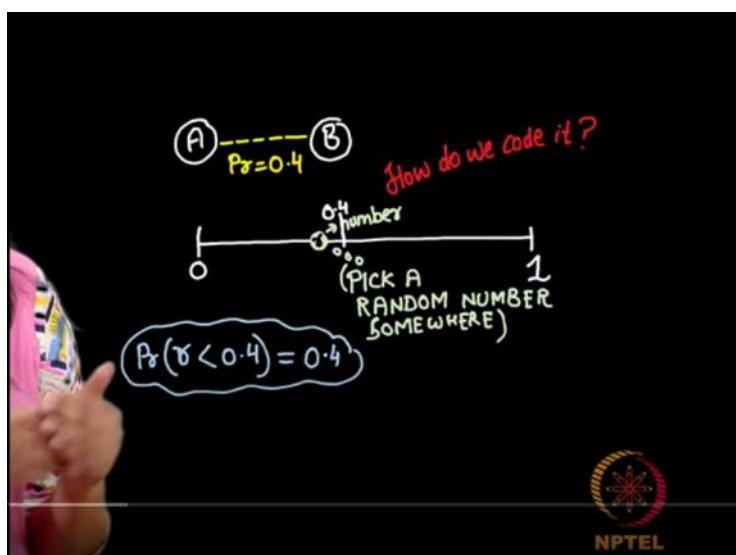
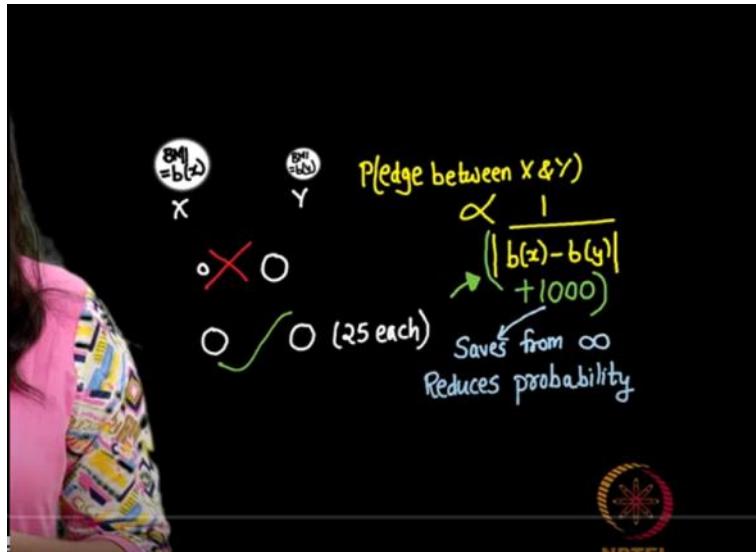
↓

BMI
= $b(x)$
X

BMI
= $b(y)$
Y

$$P(\text{edge between } X \& Y) \propto \frac{1}{|b(x) - b(y)|}$$

To implement homophily we introduced the logic that probability to become friends is inversely proportional to the : diff of bmi !!



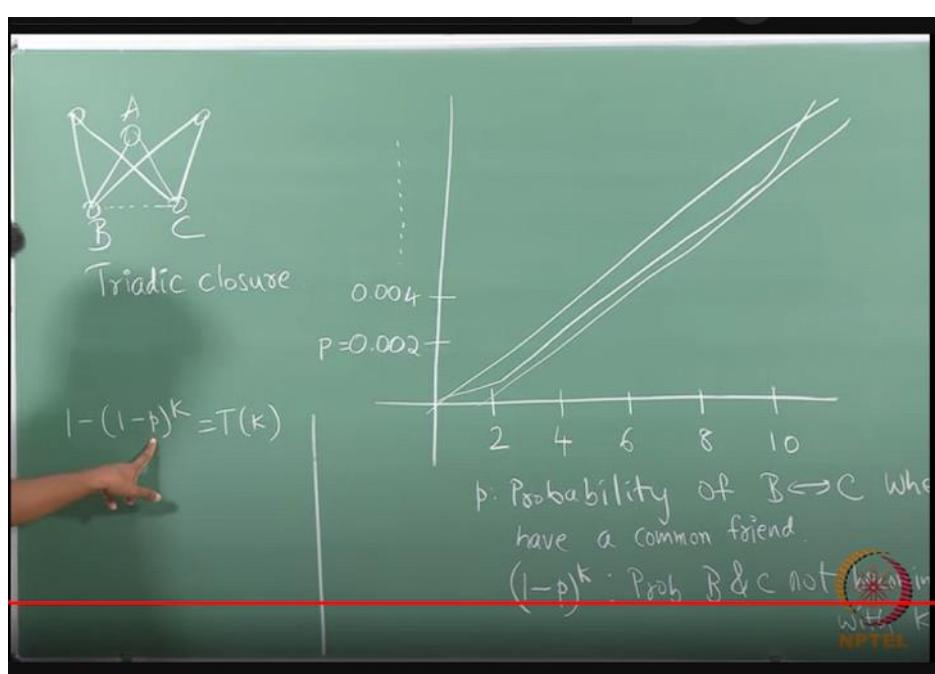
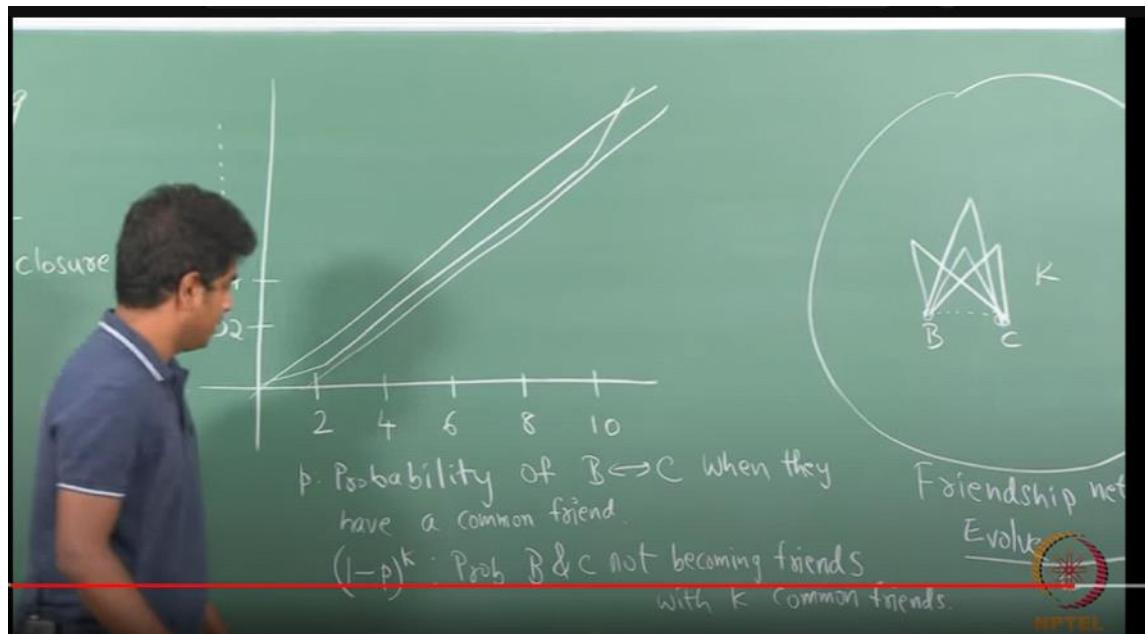
```

person_nodes=get_persons_nodes()
for each in person_nodes:
    r=random.choice(foci_nodes)
    G.add_edge(each,r)

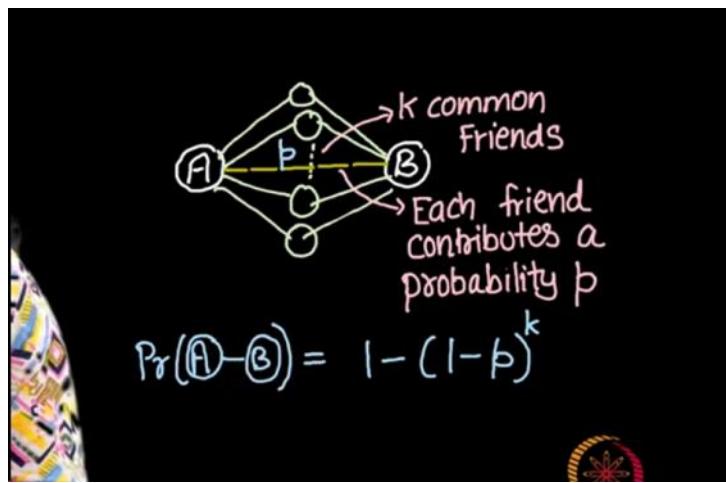
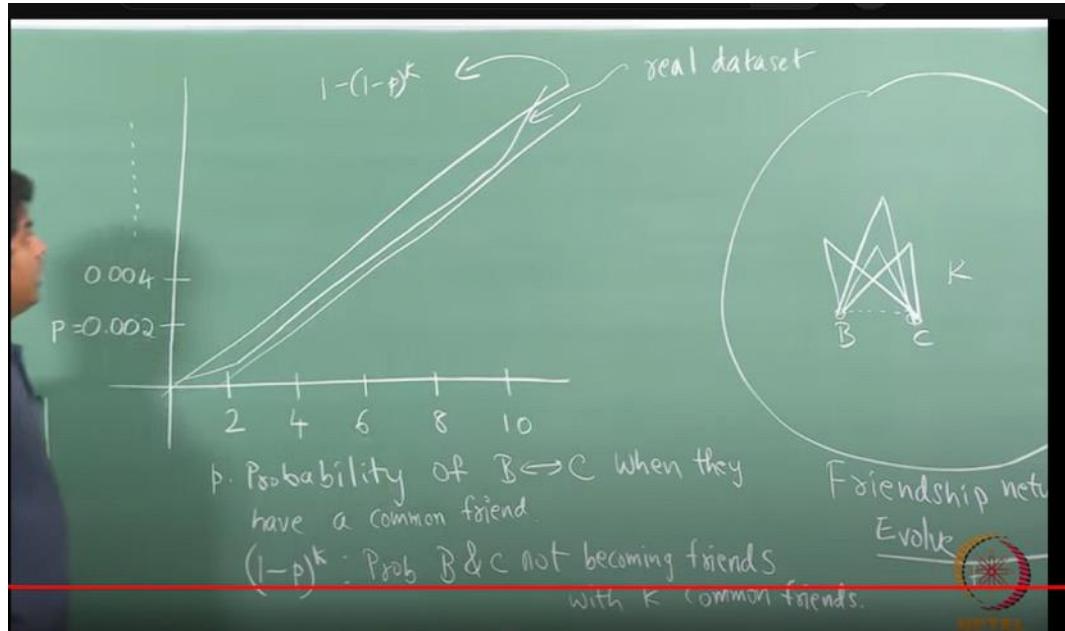
def homophily(G):
    pnodes=get_persons_nodes()
    for u in pnodes:
        for v in pnodes:
            if u!=v:
                diff=abs(G.node[u]['name']-
G.node[v]['name'])
                p=float(1)/(diff+1000)
                r=random.uniform(0,1)
                if r<p:
                    G.add_edge(u,v)

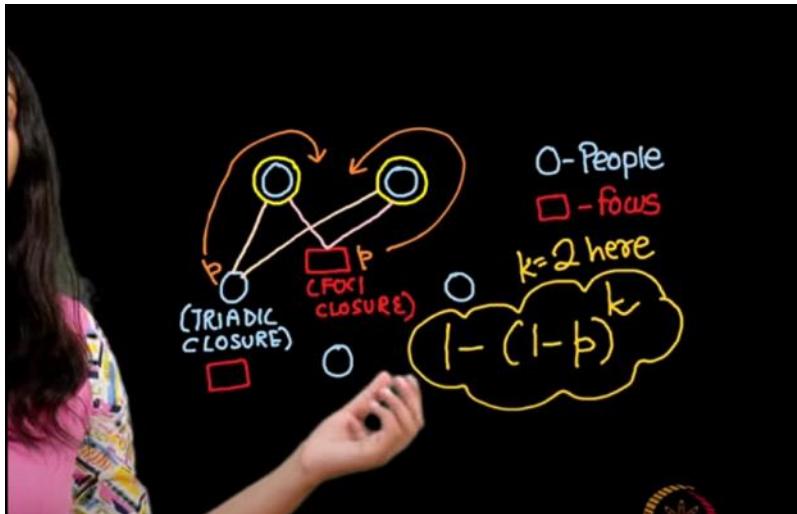
```

02 Python Tab Width: 8 Ln 93, Col 43 IINS



Probability of 2 people becoming friends when they have k common friends = $1 - (1-p)^k$





```

evolutionary_model.py * r=random.uniform(0,1)
if r<p:
    G.add_edge(u,v)

G=create_graph()
assign_bmi(G)
add_foci_nodes(G)
labeldict=get_labels(G)
nodesize=get_size(G)
color_array=get_colors(G)
add_foci_edges()

visualize(G,labeldict,nodesize,color_array)
#homophily(G)
closure(G)

visualize(G,labeldict,nodesize,color_array)

```

```

evolutionary_model.py (1).pyz * for v in pnodes:
    if u!=v:
        diff=abs(G.node[u]['name']-
G.node[v]['name'])
        p=float(1)/(diff+5000)
        r=random.uniform(0,1)
        if r<p:
            G.add_edge(u,v)

def cmn(u,v,G):
    nu=G.neighbors(u)

def closure(G):
    array1=[]
    for u in G.nodes():
        for v in G.nodes():
            if u!=v and (G.node[u]
['type']=='person' or G.node[v]['type']=='person'):

```

```

evolutionary_model.py (2).pyz *
def closure(u):
    array1=[]
    for u in G.nodes():
        for v in G.nodes():
            if u!=v and (G.node[u]
['type']=='person' or G.node[v]['type']=='person'):
                k=cmn
                (u,v,G)
                p=1-math.pow((1-0.1),k)
                tmp=[]
                tmp.append(u)
                tmp.append(v)
                tmp.append(p)
                array1.append(tmp)

print array1
G=create_graph()
assign_bmi(G)

```

```

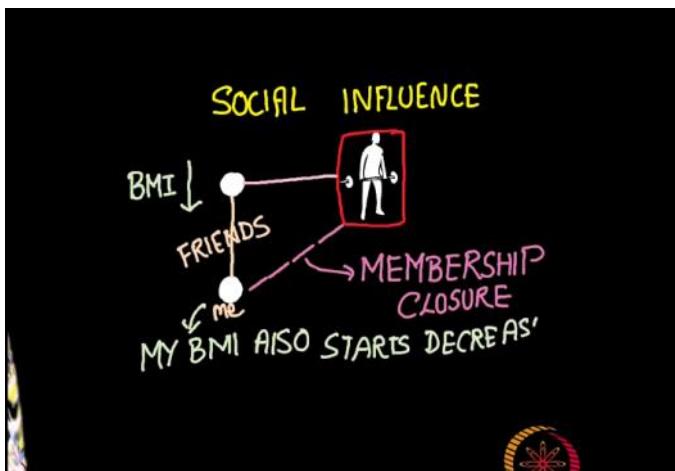
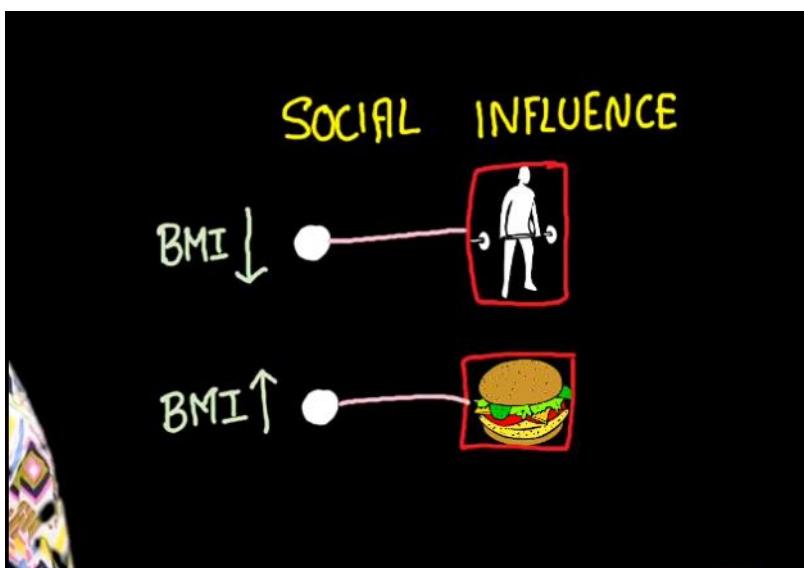
revolutionary_model.py [1]-gedit
K=CMN

(u,v,G)
    p=1-math.pow((1-0.1),k)
    tmp=[]
    tmp.append(u)
    tmp.append(v)
    tmp.append(p)
    arrayl.append(tmp)

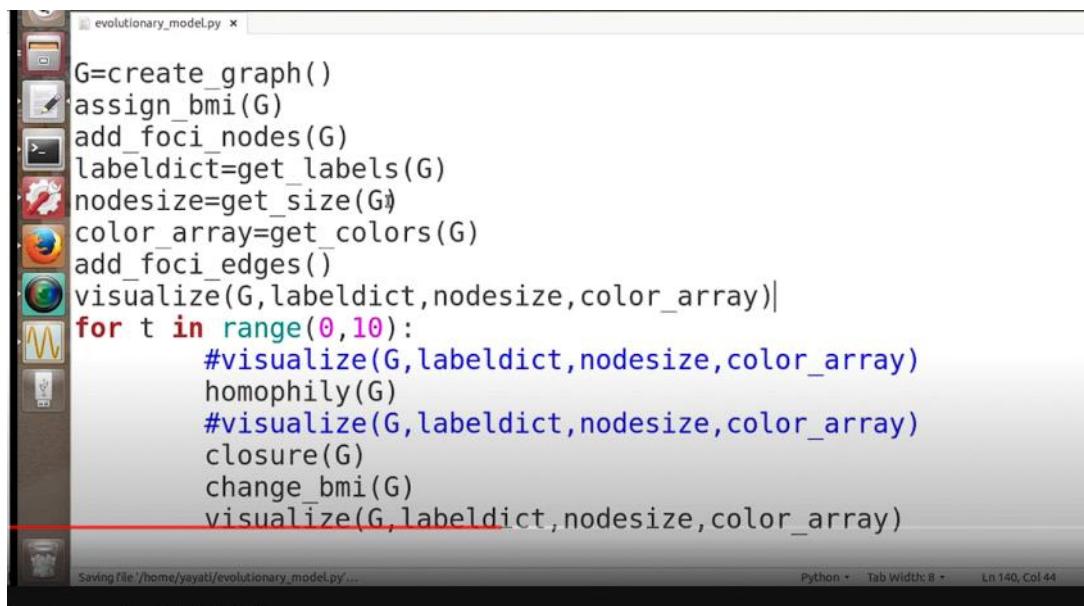
for each in arrayl:
    u= each[0]
    v=each[1]
    p=each[2]
    r= random.uniform(0,1)
    if r<p:
        G.add_edge(u,v)

G=create_graph()
assign_bmi(G)

```

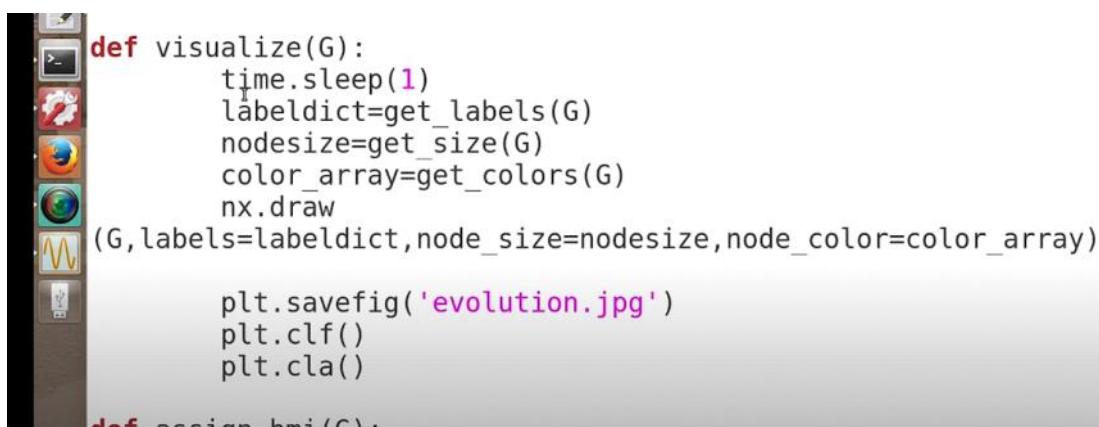


```
*evolutionary_model.py x
change_bmi(G):
    fnodes=get_foci_nodes()
    for each in fnodes:
        if G.node[each]['name']=='eatout':
            for each1 in G.neighbors(each):
                if G.node[each1]['name']!=40:
                    G.node[each1]
['name']=G.node[each1]['name']+1
        if G.node[each]['name']=='gym':
            for each1 in G.neighbors(each):
                if G.node[each1]['name']!=15:
                    G.node[each1]
['name']=G.node[each1]['name']-1
```



```
G=create_graph()
assign_bmi(G)
add_foci_nodes(G)
labeldict=get_labels(G)
nodesize=get_size(G)
color_array=get_colors(G)
add_foci_edges()
visualize(G,labeldict,nodesize,color_array)
for t in range(0,10):
    #visualize(G,labeldict,nodesize,color_array)
    homophily(G)
    #visualize(G,labeldict,nodesize,color_array)
    closure(G)
    change_bmi(G)
    visualize(G,labeldict,nodesize,color_array)
```

Saving file '/home/yayati/evolutionary_model.py'... Python Tab Width: 8 Ln 140, Col 44



```
def visualize(G):
    time.sleep(1)
    labeldict=get_labels(G)
    nodesize=get_size(G)
    color_array=get_colors(G)
    nx.draw
    (G,labels=labeldict,node_size=nodesize,node_color=color_array)

    plt.savefig('evolution.jpg')
    plt.clf()
    plt.cla()

def assign_bmi(G):
```

Social Networks

Strong and Weak Relationships (Continued) & Homophily



Fatman Evolutionary Model -
Storing and analyzing longitudinal data



S. R. S. Iyengar
Department of Computer Science
IIT Ropar



You
what is density of a graph ?



ChatGPT
In graph theory, the density of a graph is a measure of how "dense" the connections are between the vertices of the graph. It quantifies the proportion of edges that are present in the graph relative to the total number of possible edges in the graph.

Formally, the density of a graph G , denoted by $D(G)$, is calculated as follows:

$$D(G) = \frac{2|E|}{|V|(|V|-1)}$$

Where:

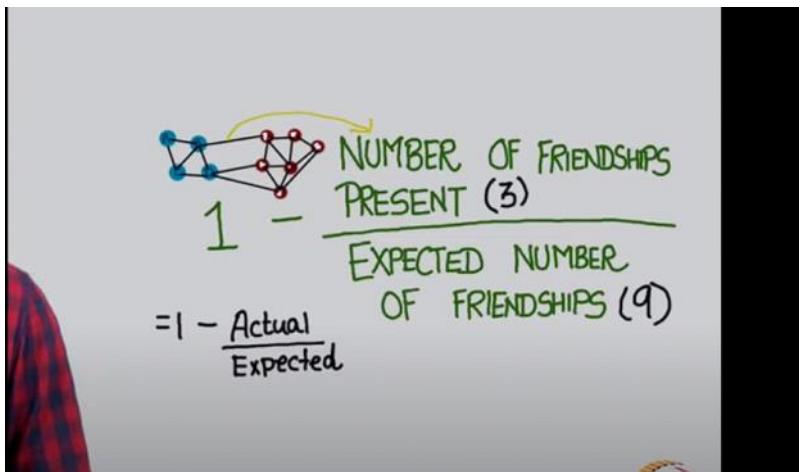
- $|E|$ is the number of edges in the graph.
- $|V|$ is the number of vertices (nodes) in the graph.

The factor of 2 in the numerator is because each edge connects two vertices, and dividing by $|V| \cdot (|V| - 1)$ gives the total number of possible edges in an undirected graph with $|V|$ vertices.

Super duper imp formula : DENSITY OF A GRAPH !!

```
evolutionary_model.py x analyze.py x
for each in G.nodes():
    if G.node[each]['name']==40:
        num=num+1
return num

def plot_obesity():
    x=[]
    y=[]
    for i in range (0,11):
        G=nx.read_gml('evolution_'+str(i)+'.gml')
        x.append(i)
        y.append(obesity(G))
    plt.xlabel('Time')
    plt.ylabel('Obesity')
    plt.title('Change in Obesity')
    plt.plot(x,y)
    plt.show()
```

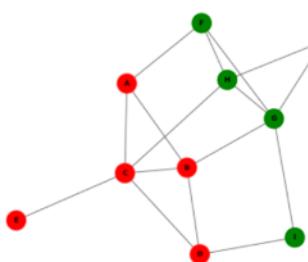


HOMOPHILY FORMULA KA ASLI EXPLANATION :

EXPECTED NUMBER OF FRIENDSHIPS = $1/2 * (\text{TOTAL EDGES})$

{ VVVVVVVIMP }

- : 6) Given the friendship network in a dance school, 5 girls represented by red nodes and 5 boys represented by green nodes. Find the Homophily of 1 person in the network.

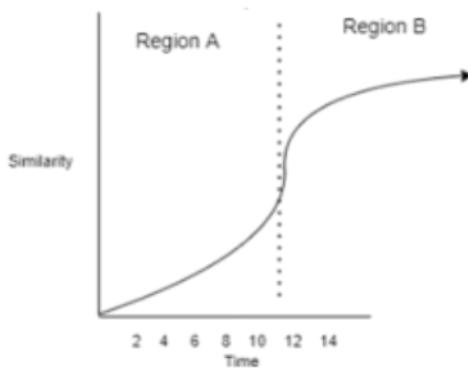


- 0
- 1/4
- 1/2
- 1

Yes, the answer is correct.



- 10) Which phenomenon is represented in region B of the given graph?



- Selection
- Social Influence
- Focal Closure
- Membership

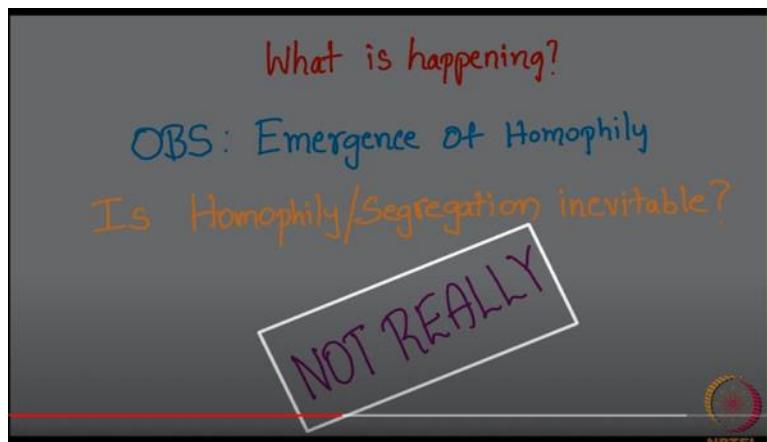
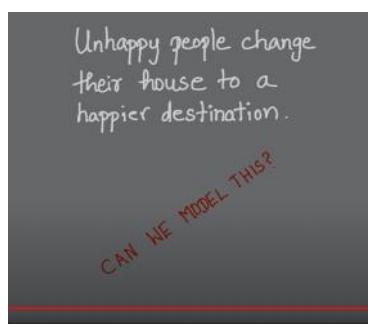
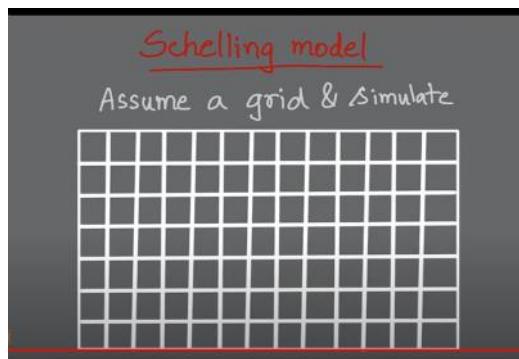
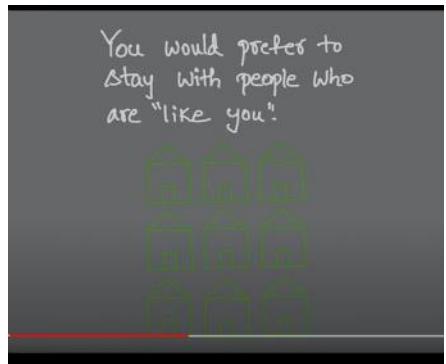
Yes, the answer is correct.

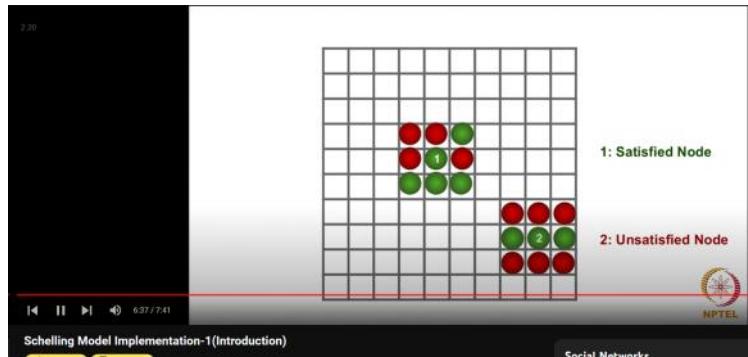
Score: 1

Accepted Answers:

Social Influence

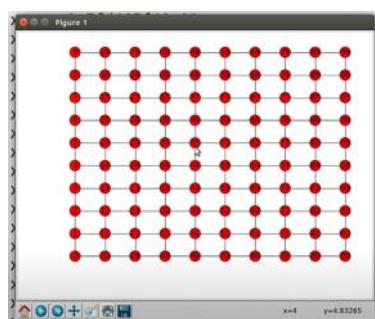
REGION A : SELECTION AND REGION B : SOCIAL INFLUENCE !!





```
In [2]: N = 10
In [3]: G = nx.grid_2d_graph(N,N)
In [4]: import matplotlib.pyplot as plt
In [5]: nx.draw(G)
In [6]: plt.show()
In [7]: G.nodes()
Out[7]:
[(7, 3),
 (6, 9),
 (0, 7),
 (1, 6),
 (3, 7),
 (2, 5),
 (8, 5),
 (5, 8),
 (4, 0),
 (5, 2)]
```

```
...More details...
In [8]: pos = dict((n,n) for n in G.nodes())
In [9]: nx.draw(G, pos)
In [10]: plt.show()
In [11]: labels = dict(((i,j), i*10+j) for i,j in G.nodes())
In [12]: nx.draw(G, pos, with_labels = False)
In [13]: plt.show()
In [14]: nx.draw(G, pos, with_labels = False)
In [15]: nx.draw_networkx_labels(G, pos, labels = labels)
```



```
23
24     for n in G.nodes():
25         G.node[n]['type'] = random.randint(0,2)
26
27     type1_node_list = [n for (n,d) in G.nodes(data=True) if d['type'] == 1]
28     type2_node_list = [n for (n,d) in G.nodes(data=True) if d['type'] == 2]
29     empty_cells = [n for (n,d) in G.nodes(data=True) if d['type'] == 0]
30
31
32
33
34 # print type1_node_list
35 # print type2_node_list
36 # print empty_cells
37
38 def display(graph):
39     nodes_g = nx.draw_networkx_nodes(G, pos, node_color = 'green', nodeList = type1_node_list)
40     nodes_r = nx.draw_networkx_nodes(G, pos, node_color = 'red', nodeList = type2_node_list)
41     nodes_w = nx.draw_networkx_nodes(G, pos, node_color = 'white', nodeList = empty_cell)
42
43     nx.draw_networkx_edges(G,pos)
44     nx.draw_networkx_labels(G, pos, labels = labels)
45     plt.show()
46
47
48
```

Schelling Model Implementation-3 (Visualization and Getting a list of boundary and internal nodes)

Social Networks

```

51
52
53 # print type1_node list
54 # print type2_node list
55 # print empty_cells
56
57 display_graph(G)
58
59 boundary_nodes_list = get_boundary_nodes(G)
60 internal_nodes_list = list(set(G.nodes()) - set(boundary_nodes_list))
61
62 print boundary nodes list
63 print internal nodes list

```

Social Networks

Homophily (Continued) &

Positive and Negative Relationships

Schelling Model Implementation - Getting a list of unsatisfied nodes



```
schelling_video.py  # 3.27 0
9 def get_neigh_for_internal(u,v):
10    return [(u-1, v), (u+1, v), (u, v-1), (u, v+1), (u-1, v+1), (u+1, v-1), (u-1, v-1),
11
12 def get_neigh_for_boundary(u,v):
13    # global N
14    # print 'uv', u,v
15    if u == 0 and v == 0:
16        return [(0,1), (1,1), (1,0)]
17    elif u == N-1 and v == N-1:
18        return [(N-2,N-2),(N-1,N-2), (N-2, N-1)]
19    elif u == N-1 and v == 0:
20        return [(u-1, v), (u, v+1), (u-1, v+1)]
21    elif u == 0 and v == N-1:
22        return [(u+1, v), (u+1, v-1), (u,v-1)]
23    elif u == 0:
24        return [(0,v-1), (u, v+1), (u+1, v), (u+1, v-1), (u+1, v+1)]
25    elif u == N-1:
26        return [(u-1, v), (u, v-1), (u, v+1), (u-1, v+1), (u-1, v-1)]
27    elif v == N-1:
28        return [(u, v-1), (u-1, v), (u+1, v), (u-1, v-1), (u+1, v-1)]
29    elif v == 0:
30        return [(u-1, v), (u+1, v), (u, v+1), (u-1, v+1), (u+1, v+1)]
```

```

51     return [(u-1, v), (u+1, v), (u, v+1), (u-1, v+1), (u+1, v+1)]
52
53 def get_unsatisfied_nodes_list(G, internal_nodes_list, boundary_nodes_list):
54     unsatisfied_nodes_list = []
55     t = 3
56     for u, v in G.nodes():
57         type_of_this_node = G.node[(u,v)]['type']
58         if type_of_this_node == 0:
59             continue
60         else:
61             similar_nodes = 0
62             if (u,v) in internal_nodes_list:
63                 neigh = get_neigh_for_internal(u,v)
64             elif (u,v) in boundary_nodes_list:
65                 neigh = get_neigh_for_boundary(u,v)
66
67             for each in neigh:
68                 if G.node[each]['type'] == type_of_this_node:
69                     similar_nodes += 1
70
71             if similar_nodes <= t:
72                 unsatisfied_nodes_list.append((u,v))
73
74 return unsatisfied_nodes_list
75
76 wing the graph >

```

Social Networks

Homophily (Continued) &
Positive and Negative Relationships



Schelling Model Implementation -
Shifting the Unsatisfied Nodes &
Visualizing the Final Graph

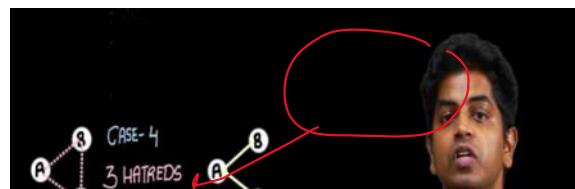
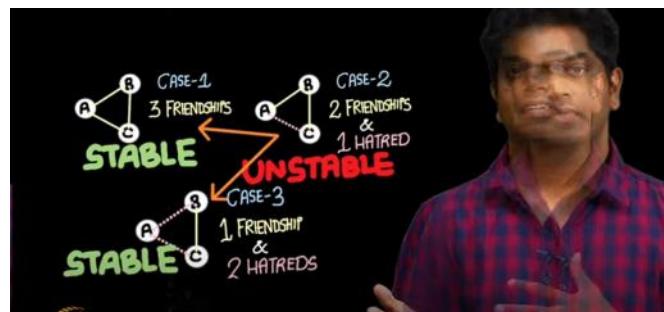


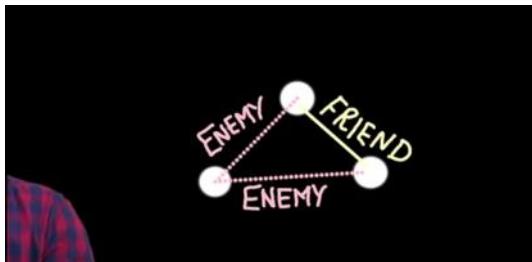
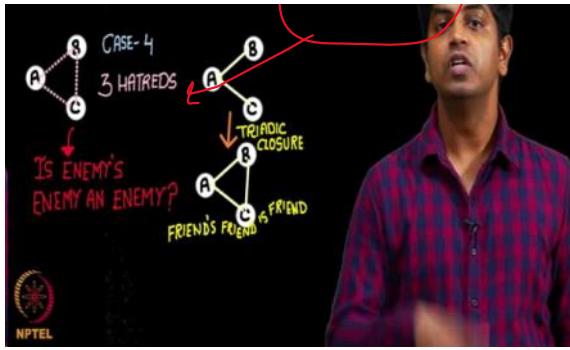
S. R. S. Iyengar
Department of Computer Science
IIT Ropar



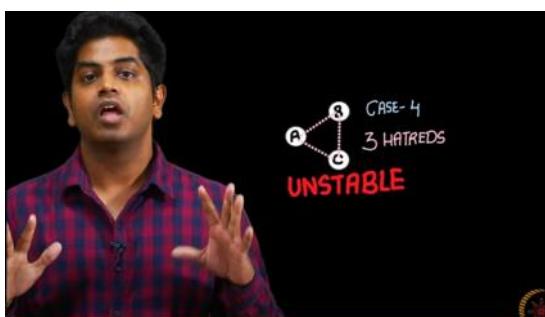
```
74 def make_a_node_satisfied(unsatisfied_nodes_list, empty_cells):
75     if len(unsatisfied_nodes_list) == 0:
76         node_to_shift = random.choice(unsatisfied_nodes_list)
77         new_position = random.choice(empty_cells)
78
79         G.node[new_position]['type'] = G.node[node_to_shift]['type']
80         G.node[node_to_shift]['type'] = 0
81         labels[node_to_shift], labels[new_position] = labels[new_position], labels[node_to_shift]
82
83     else:
84         pass
85
86
```

Positive and Negative Relationships :



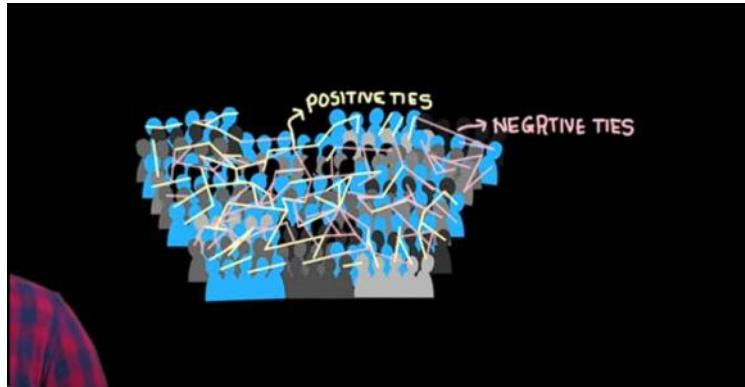


Dushman ka dushman dost hota hai !! : famous proverb !!

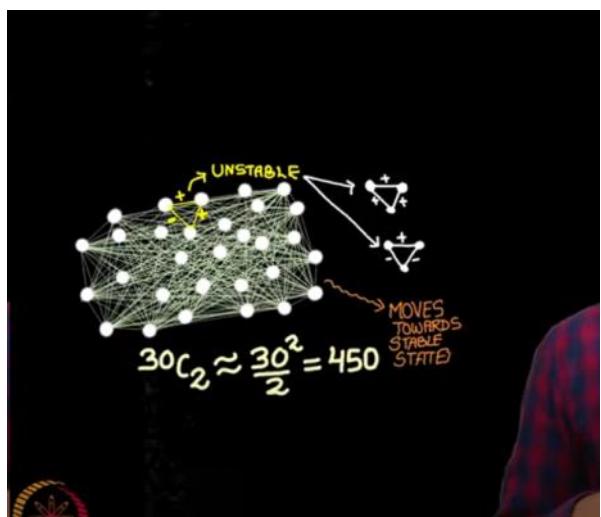


Reason for this is given in above ss !!

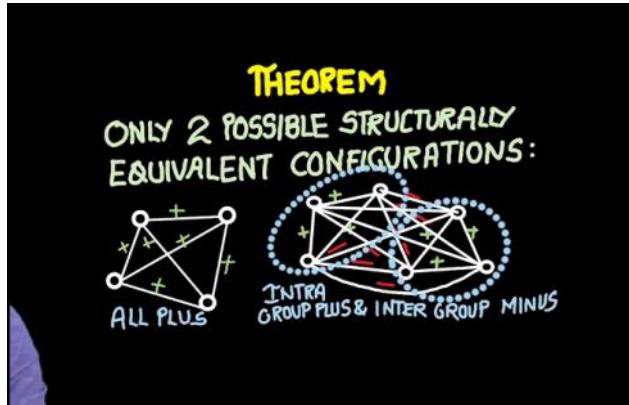
Characterizing the structure of balanced networks



Positive relationships Means : friendships and negative means hatred
jo dekha tha abhi hamne



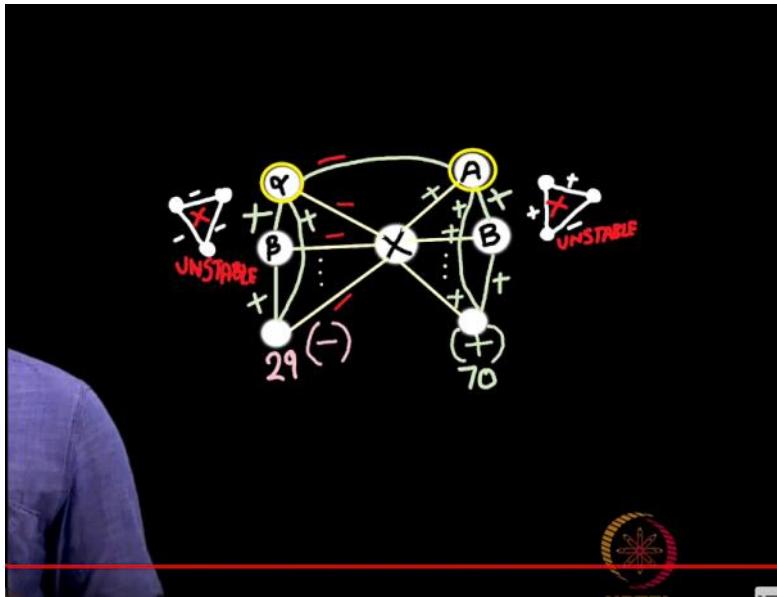
HOW DOES THE FINAL STABLE
GRAPH LOOK LIKE?



This is called Balance Theorem which states that only these 2 stable / balanced configurations exist in case of positive and negative ties graph in stable state !!

//This is based on a common sense idea that : 3 teams never exist in a war / fight

bcz : enemy's enemy is friend !!



Proof for this theorem !!

Code Implementation of above idea :

Steps for Implementation

1. Create a graph with 10 nodes, where the nodes are the countries.
2. Make it a complete graph by adding all possible edges. Also, assign '+' or '-' signs as weights to all the edges randomly.
3. Display the network.
- 4.1 Get a list of all the triangles in the network.
- 4.2 Store the sign details of all the triangles.
- 4.3 Count the number of unstable triangles in the network.



5. While the number of unstable triangles is not zero, do the following:
 - 5.1. Choose a triangle in the graph that is unstable.
 - 5.2. Make that triangle stable.
 - 5.3. Count the number of unstable triangles

6. Now that there is no unstable triangle in the network, it can be divided into two coalitions, such that in each coalition, the intra-edges are positive, and the inter-edges are negative.

- 6.1. Choose a random node. Add it to the first coalition.
- 6.2. Also put all the 'friends' of this node in the first coalition.
- 6.3. Put all the 'enemies' of this node in the second coalition.
- 6.4. Repeat steps 6.2 and 6.3 for all the 'unprocessed' nodes of first coalition.

7. Display the network with coalitions



```
10 G.add_nodes_from([(i for i in range(1, n+1))])
11 mapping = {1:'Alexandra',2:'Anterim',3:'Bercy', 4: 'Bearland', 5: 'Eplex'}
12 G = nx.relabel_nodes(mapping, G)
13
14 # 2. Make it a complete graph by adding all possible edges. Also, assign
15 signs = ['+', '-']
16 for i in G.nodes():
17     for j in G.nodes():
18         if i != j:
19             G.add_edge(i, j, sign = random.choice(signs))
20
21 # 3. Display the network.
22 edge_labels = nx.get_edge_attributes(G, 'sign')
23 pos = nx.circular_layout(G)
24 nx.draw(G)
25 nx.draw_networkx_edge_labels(G, pos, edge_labels = edge_labels, font_size=10)
26
27 # 4.1 Get a list of all the triangles in the network.
28 # 4.2 Store the sign details of all the triangles.
29 # 4.3 Count the number of unstable triangles in the network.
30
```

```
45
46 # 4.1 Get a list of all the triangles in the network.
47 nodes = G.nodes()
48 tris_list = [list(x) for x in itertools.combinations(nodes,3)]
49 # 4.2 Store the sign details of all the triangles.
50 all_signs = get_signs_of_tris(tris_list, G) #[[['+', '+', '-'],[],[],[]]
51 # 4.3 Count the number of unstable triangles in the network.
52 unstable = count_unstable(all_signs)
53
```

```
def get_signs_of_tris(tris_list, G):
    #tris_list = [[1,2,3],[4,5,3]]
    #all_signs = [['+', '-', '+'],[],[]]
    all_signs = []
    for i in range(len(tris_list)):
        temp = []
        temp.append(G[tris_list[i][0]][tris_list[i][1]]['sign'])
        temp.append(G[tris_list[i][1]][tris_list[i][2]]['sign'])
        temp.append(G[tris_list[i][2]][tris_list[i][0]]['sign'])
        all_signs.append(temp)
    return all_signs
```

Social Networks

Homophily (Continued) &

Positive and Negative Relationships



Moving a network from an
unstable to stable state



```

100 unstable = count_unstable(all_signs)
101
102 # 5. While the number of unstable triangles is not zero, do the fo
103 # 5.1. Choose a triangle in the graph that is unstable.
104 # 5.2. Make that triangle stable.
105 # 5.3. Count the number of unstable triangles
106 while(unstable != 0):
107     G = move_a_tri_to_stable(G, tris_list, all_signs)
108     all_signs = get_signs_of_tris(tris_list, G)
109     unstable = count_unstable(all_signs)
110
111
112 # 6. Now that there is no unstable triangle in the network, it can

```

```

29
30 def move_a_tri_to_stable(G, tris_list, all_signs):
31     found_unstable = False
32     while(found_unstable == False):
33         index = random.randint(0, len(tris_list)-1)
34         if all_signs[index].count('+') == 2 or all_signs[index].count('-') == 2:
35             found_unstable = True
36         else:
37             continue
38
39     #move the unstable triangle to a stable state
40     r = random.randint(1,3)
41     if all_signs[index].count('+') == 2:
42         if r == 1:
43             if G[tris_list[index]][]['sign'] == '+':

```

3 cases possible to make it stable that is why : unme rand int chala to pick up randomly konsa transformation karna hai to make it stable !!

```

#move the unstable triangle to a stable state
#[1,2,3]
r = random.randint(1,3)
if all_signs[index].count('+') == 2:
    if r == 1:
        if G[tris_list[index][0]][tris_list[index][1]]['sign'] == '+':
            G[tris_list[index][0]][tris_list[index][1]]['sign'] = '-'
        elif G[tris_list[index][0]][tris_list[index][1]]['sign'] == '-':
            G[tris_list[index][0]][tris_list[index][1]]['sign'] = '+'
    elif r == 2:
        if G[tris_list[index][1]][tris_list[index][2]]['sign'] == '+':
            G[tris_list[index][1]][tris_list[index][2]]['sign'] = '-'
        elif G[tris_list[index][1]][tris_list[index][2]]['sign'] == '-':
            G[tris_list[index][1]][tris_list[index][2]]['sign'] = '+'
    elif r == 3:
        if G[tris_list[index][2]][tris_list[index][0]]['sign'] == '+':
            G[tris_list[index][2]][tris_list[index][0]]['sign'] = '-'
        elif G[tris_list[index][2]][tris_list[index][0]]['sign'] == '-':
            G[tris_list[index][2]][tris_list[index][0]]['sign'] = '+'

```

Just basic code implementation !!

```

107 # 5.3. Count the number of unstable triangles
108 while(unstable != 0):
109     G = move_a_tri_to_stable(G, tris_list, all_signs)
110     all_signs = get_signs_of_tris(tris_list, G)
111     unstable = count_unstable(all_signs)
112     unstable_track.append(unstable)
113
114 raw_input()
115 plt.bar([i for i in range(len(unstable_track))], unstable_track)
116 plt.show()
117
118
119 # 6. Now that there is no unstable triangle in the network, it can

```

```

processed_nodes = []
to_be_processed = [r]

for each in to_be_processed:
    if each not in processed_nodes:
        neigh = G.neighbors(each)

        for i in range(len(neigh)):
            if G[each][neigh[i]]['sign'] == '+':
                if neigh[i] not in first_coalition:
                    first_coalition.append(neigh[i])
                if neigh[i] not in to_be_processed:
                    to_be_processed.append(neigh[i])
            elif G[each][neigh[i]]['sign'] == '-':
                if neigh[i] not in second_coalition:
                    second_coalition.append(neigh[i])
                to_be_processed.append(neigh[i])

    processed_nodes.append(each)

```

Normal bfs , dfs wagera !!

```

160 print second
161
162 edge_labels = nx.get_edge_attributes(G, 'sign')
163 # 7. Display the network with coalitions
164 pos = nx.circular_layout(G)
165 nx.draw_networkx_nodes(G, pos, nodelist = first, node_color = 'red'
166 nx.draw_networkx_nodes(G, pos, nodelist = second, node_color = 'blue'
167
168 nx.draw_networkx_labels(G, pos, font_size = 10, font_family = 'sans-serif')
169
170 nx.draw_networkx_edges(G, pos)
171 nx.draw_networkx_edge_labels(G, pos, edge_labels = edge_labels, font_size = 10)

```

- 8) Consider Frank McCown's simulation of the Schelling model illustrated in the course. The simulation has parameters: Similar (threshold), empty (fraction of empty cells) and red/blue ratio. Choose the **INCORRECT** statements from the options.

- As similar increases, it takes more time for the system to stabilise
- The system will always stabilise no matter what the parameters are
- The system can stabilise even if similar = 100%
- Spatial segregation doesn't always induce communities/clustering in the grid

Yes, the answer is correct.

Score: 1

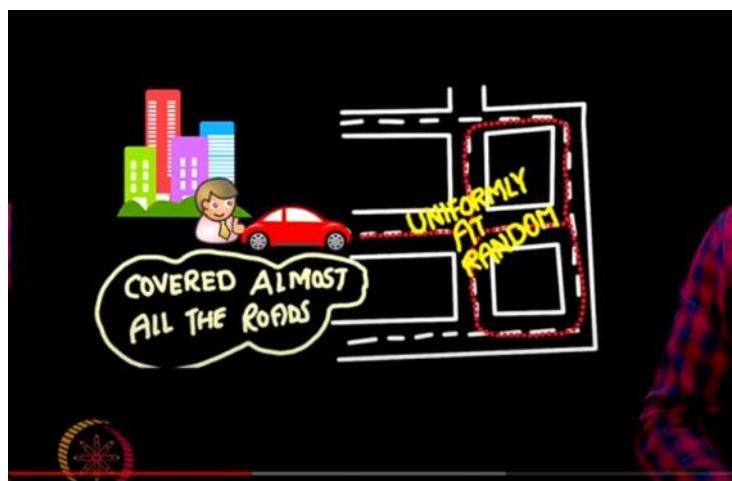
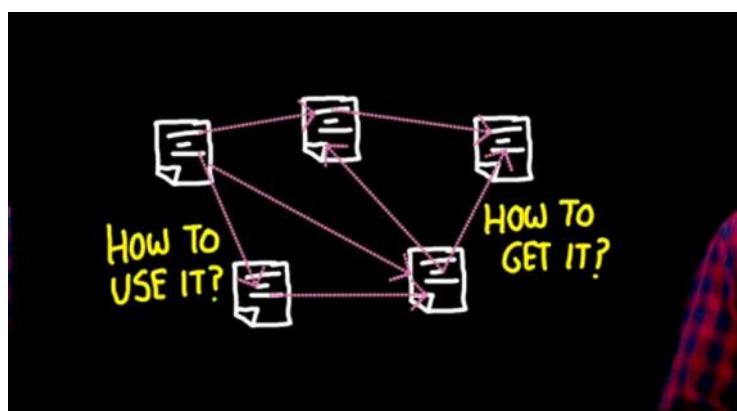
Accepted Answers:

The system will always stabilise no matter what the parameters are



Vvimp Problem !!

Week 6 :



Generate a Random Graph
Pick a Vertex Randomly
Start Walking on the Graph Randomly
When will one visit all the vertices?
Will one at all visit all the vertices, ever?

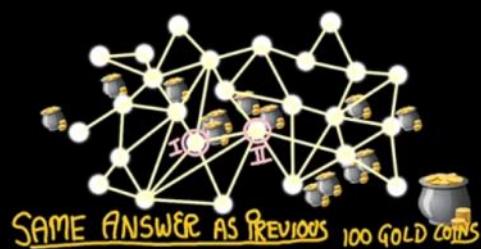
```

1 import random
2 import matplotlib.pyplot as plt
3 import networkx as nx
4 import numpy
5
6 def walk(n,p):
7     start=random.randint(0,n-1)
8     G=nx.erdos_renyi_graph(n,p)
9     S=set([ ])
10    v=start
11    count=0
12    while(len(S)<n):
13        Nbr=nx.neighbors(G,v)
14        v=random.choice(Nbr)
15        S.add(v)
16        count=count+1
17    return count
18
19 l=[]
20 for i in range(20,300):
21     z=[]
22     for j in range(10):
23         z.append(walk(10,0.3))
24     l.append(numpy.average(z))
25     print i,"-->",numpy.average(z)
26 plt.plot(l)
27 plt.show()
28

```

7:50 / 10:36

"checkrandom.py" 28L, 522C written

Collecting the Web Graph**RANDOM WALK ! CODE !!**

3:00 / 8:59

**Random Coin Dropping**

EQUAL SHARING
 \approx
 RANDOM DROPPING

Steps for Implementation

- ➊ Create/take a directed graph with 'n' nodes.
- ➋ Assign 100 points to each node.
- ➌ Keep distributing points until convergence.
- ➍ Get nodes' ranking as per the points accumulated.
- ➎ Compare the ranks thus obtained with the ranks obtained from the inbuilt Page Rank method.

◀ ▶ 🔍 3.26 / 4:33

Implementing PageRank Using Points Distribution Method-1

Social Networks

```

58
59 def main():
60     # Create/take a directed graph with `n` nodes.
61     G = nx.DiGraph()
62     G.add_nodes_from([i for i in range(10)])
63     G = add_edges(G, 0.3)
64
65     # Assign 100 points to each node.
66     points = initialize_points(G)
67
68     # Keep distributing points until convergence.
69     G, points = keep_distributing_points(G, points)
70
71     # Get nodes' ranking as per the points accumulated.
72     # Compare the ranks thus obtained with the ranks obtained from the j
73
74

```

```

19 def distribute_points(G, points):
20     prev_points = points
21     new_points = [0 for i in range(G.number_of_nodes())]
22
23     for i in G.nodes():
24         out = G.out_edges(i)
25         if len(out) == 0:
26             new_points[i] += prev_points[i]
27         else:
28             share = (float)(prev_points[i])/len(out)
29             for each in out:
30                 new_points[each[1]] += share
31
32     return G, new_points
33
34
35 def keep_distributing_points(G, points):
36     prev_points = points
37     while(1):
38         G, new_points = distribute_points(G, prev_points)
39

```

```

40     return G, new_points
41
42 def get_nodes_sorted_by_points(points):
43     points_array = np.array(points)
44     nodes_sorted_by_points = np.argsort(-points_array)
45     return nodes_sorted_by_points
46
47
48

```

NumPy -
<https://numpy.org/doc/stable/reference/generated/numpy.argsort.html>

numpy.argsort — NumPy v1.26 Manual

numpy.argsort# ... Returns the indices that would sort an array. Perform an indirect sort along the given axis using the algorithm specified by the kind keyword.

```
G, points = keep_distributing_points(G, points)

# Get nodes' ranking as per the points accumulated.
nodes_sorted_by_points = get_nodes_sorted_by_points(points)
print 'nodes_sorted_by_points', nodes_sorted_by_points
# Compare the ranks thus obtained with the ranks obtained from the inb

pr = nx.pagerank(G)
pr_sorted = sorted(pr.items(), key = lambda x:x[1], reverse = True)
for i in pr_sorted:
    print i[0],
```

Using inbuilt func in network x library !!

RESULTS DON'T MATCH WHY ? : BCZ OF SINKS IN THE NETWORK !!

**WHAT ARE SINKS ? : JINME SIRF : INCOMING EDGES HAI OUTGOING WALE NAHI
HOTE !!**

SO HOW TO HANDLE THEM ? SIMILAR TO TAXATION SYSTEM IN WORLD :SARE

**NODES SE : KUCH CERTAIN PERCENTAGE OF THERE COINS LE LETE HAI AND THEN :
GIVE THEM EQUALLY TO ALL THE NODES !!**

```
35
36 def keep_distributing_points(G, points):
37     prev_points = points
38     print 'Enter # to stop'
39     while(1):
40         G, new_points = distribute_points(G, prev_points)
41         print new_points
42
43         new_points = handle_points_sink(G, new_points)
44
45         char = raw_input()
46         if char == '#':
47             break
48         prev_points = new_points
49
50     return G, new_points
51
```

```
34
35 def handle_points_sink(G, points):
36     for i in range(len(points)):
37         points[i] = (float)(points[i])*0.8
38
39     n = G.number_of_nodes()
40     extra = ((float)(n)*100*0.2)/n
41     for i in range(len(points)):
42         points[i] += extra
43
44     return points
```

```
# 
nodes_sorted_by_points [0 5 9 2 8 3 7 4 6 1]
0 5 9 2 8 3 7 4 6 1
anamika@anamika-Tsnpiron-5423:~/NPTEL/WFFK_wise/
10:08 Video_code$
```

Now values will match !!

PAGE RANK IMPLEMENTATION USING RANDOM WALK :

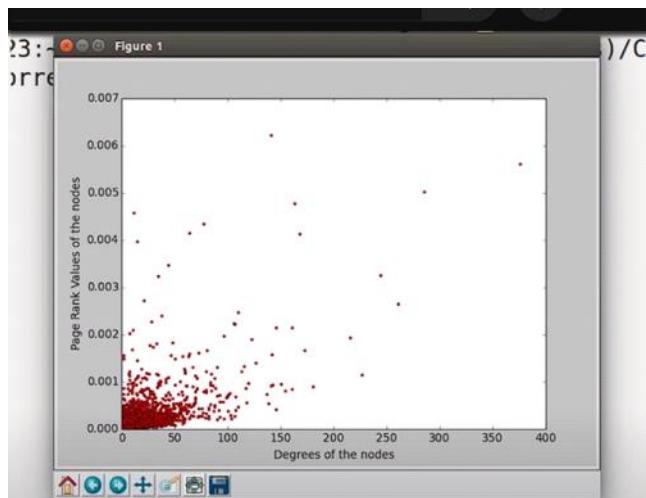
```
23
24 def random_walk(G):
25     nodes = G.nodes()
26     RW_points = [0 for i in range(G.number_of_nodes())]
27     r = random.choice(nodes)
28     RW_points[r] += 1
29     out = G.out_edges(r)
30             I
31     c = 0
32     while(c != 100000):
33         if(len(out) == 0):
34             focus = random.choice(nodes)
35         else:
36             r1 = random.choice(out)
37             focus = r1[1]
38             RW_points[focus] += 1
39             out = G.out_edges(focus)
40     |
```

REST PART IS SAME BAS DISTRIBUTION KA TAREEKA CHANGE HO JAYEGA !!

NOTE :

What is teleportation? It is the ans to the : sink edges : unse ham kahi nahi jaa sakte so : that is why we will select any random node in that case to teleport to !!

DEGREE RANK VS PAGE RANK !!



```

1 import networkx as nx
2 import matplotlib.pyplot as plt
3
4 def main():
5     G = nx.read_edgelist('citation.txt', create_using = nx.DiGraph())
6
7     deg = G.in_degree()
8     pr = nx.pagerank(G)
9
10    pr_values = []
11
12    for i in deg.keys():
13        pr_values.append(pr[i])
14
15    plt.plot(deg.values(), pr_values, 'ro', markersize = 3)
16    plt.xlabel('Degrees of the nodes')
17    plt.ylabel('Page Rank of the nodes')
18
19

```

- 2) In a game distributing gold coins across a network, each node begins with an equal **1 p** number of coins. Following one round of redistribution, the total number of coins across the network remains the same. Which network structure guarantees this outcome?

- Random network
- Scale-free network
- Ring network
- None of the above

Yes, the answer is correct.

Score: 1

Accepted Answers:

Ring network



VVGOD PROBLEM !!!

- 4) If nx represents networkx library then, for a graph G, what does nx.pagerank(G) returns?

- list of nodes in the ascending order of their page rank
- list of Page ranks of all nodes
- dictionary of node, Page rank pairs
- dictionary of edge, page rank pairs

Yes, the answer is correct.

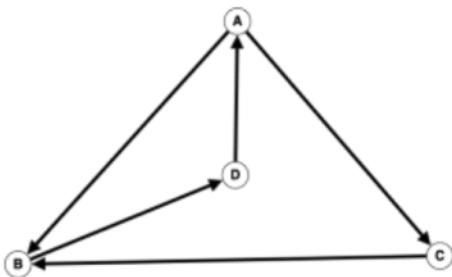
Score: 1

Accepted Answers:

dictionary of node, Page rank pairs

PLS DO FOR OTHER CODES AS WELL IF POSSIBLE !!

- 5) If the initial PageRank of each node is $\frac{1}{4}$ for the given graph below, what will be the PageRank of the nodes after 2 iterations?



- A=1/4, B=1/4, C=1/4, D=1/4
- A=1/4, B=3/8, C= 1/8, D=1/4
- A=1/4, B=1/2, C=1/8, D=1/4
- A=1/4, B=1/4, C=1/8, D=3/8

Yes, the answer is correct.

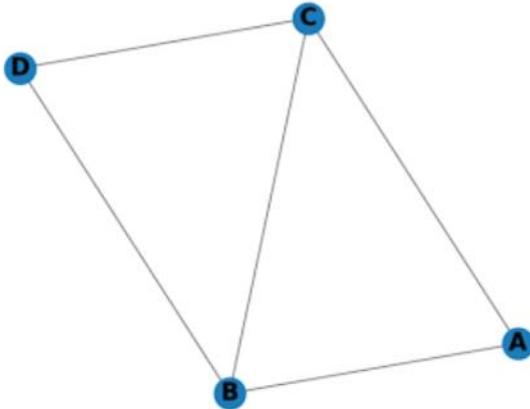
Score: 1

Accepted Answers:

A=1/4, B=1/4, C=1/8, D=3/8

EASY BUT IMP QUESTION !!

-
- 7) Consider the following graph for the gold coins distribution game. Each node in the graph G =A, B, C, D has 20, 30, 45, 30 corresponding number of gold coins initially. What is the number of gold coins after one iteration?



- 25, 40, 35, 20
- 40, 25, 35, 20
- 25, 40, 30, 20
- 40, 35, 30, 25

Isme last 20 nahi 25 hogा : misprint hai imp !!

- 8) How is the PageRank vector typically computed in the PageRank algorithm?

- Solving a system of linear equations
- Using a recursive algorithm
- Iteratively updating the vector until convergence
- All the above

No, the answer is incorrect.

Score: 0

Accepted Answers:

All the above

Random walk is a recursive process only na !!!

WEEK 7 { CASCADING BEHAVIOUR }

DEFINITION OF FOLLOW : { USED IN THE CONTEXT OF LECTURES }

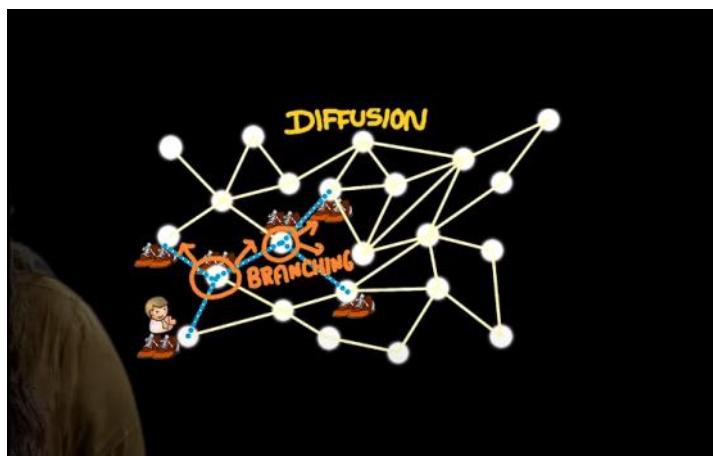
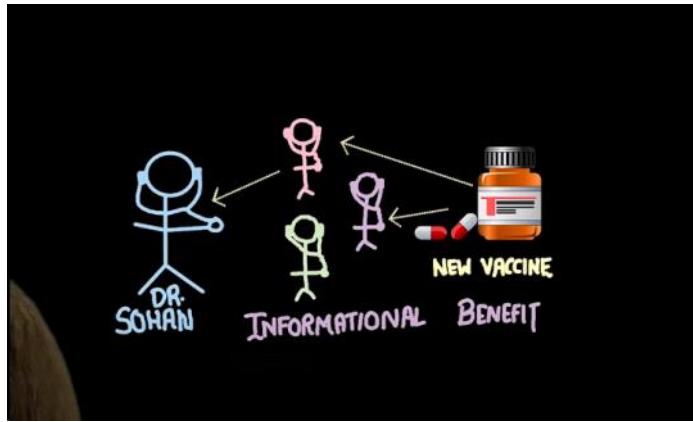
Before giving you answer to this question, I want to refine the definition of follow, not refine I want to explain in a little bit of more detail. What all kind of circumstances do you count in follow? Let us say my friend is doing something. I look at him he is doing this thing and I also start doing the same thing. It is a follow.

Let us say he is smoking cigarette. I looked at him, I feel smoking cigarette is cool. And I also start smoking cigarette, I followed him. Next my friend is using some product which is really nice. I looked at him, I like the product, I also started using it. Let us say an apple iPod; he is using an apple iPod, I liked it very much. I started using it, I followed him. A third kind of follow, rather a more involved a more intrinsic one is about the information we receive from other people. How information? My friend has information, which is of interest, some information which is of interest. It can be about a recent assignment we have to submit; it can be about news related to a Hollywood or Bollywood celebrity or something else some nice information.

So, when we talk to my friends. He tells me about this information, I like the information, I believe in the information. This is also a kind of follow; I followed him.



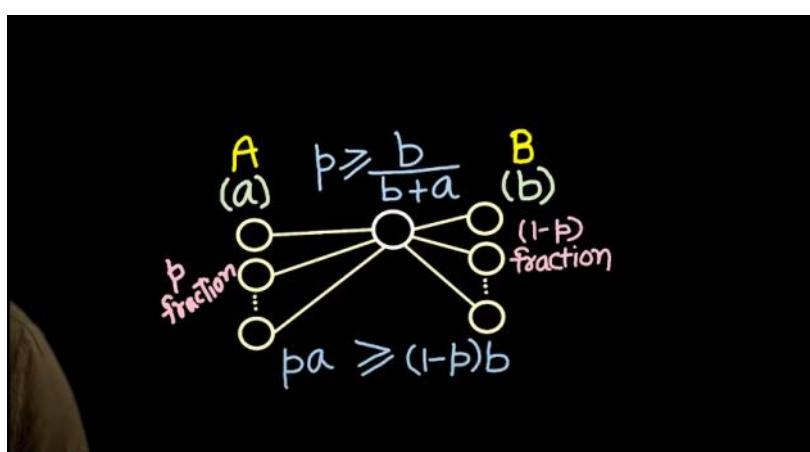
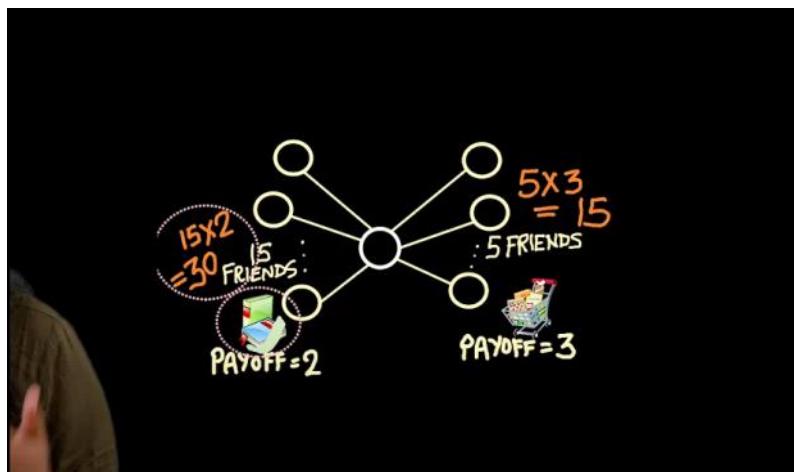
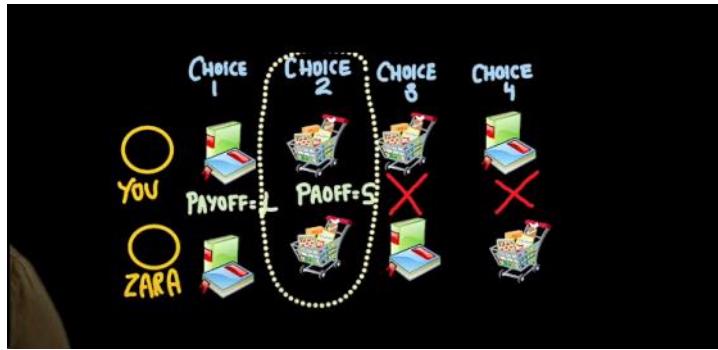
2nd reason to follow : to get usefull information !! { vvimp ! }



NEXT QUESTION : KITNA DUR TAK WITHIN THE NETWORK WILL THE GIVEN INFO DIFUSE ? : AND THE ANS TO IT IS : IT DEPENDS UPON THE TYPE OF INFORMATION LIKE : IF IT IS ABOUT A COURSE / STUDY : IT WILL DIFUSE LESS WHILE IF IT IS A GOSSIP OR SOMETHING THEN IT WILL DIFUSE TO A LOT OF EXTENT !!

WHAT WILL YOU FOLLOW AND WHAT NOT ? / WHAT ~~FACTORS GOVERN THE~~ AMOUNT OF DIFFUSION WITHIN A NETWORK ?

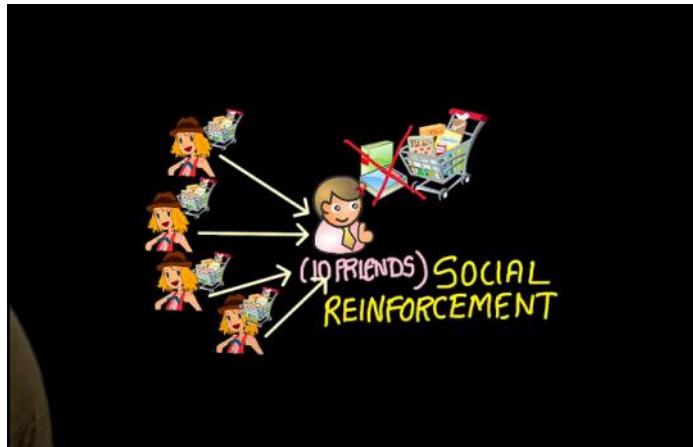




FOR GOING TO ACTIVITY A !!

$$p \geq \frac{b}{b+a}$$

→ p FRACTION OF YOUR FRIENDS SHOULD HAVE ADOPTED A.



1 You

What is social reinforcement ?

2 ChatGPT

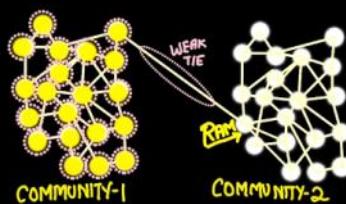
Social reinforcement refers to the process by which individuals modify their behavior based on the responses and feedback they receive from others in their social environment. It involves the use of social cues, approval, disapproval, praise, criticism, or other forms of social interaction to encourage or discourage certain behaviors.

In social reinforcement, individuals are more likely to repeat behaviors that are positively reinforced by others, such as receiving praise, attention, or rewards, while they are less likely to repeat behaviors that are met with disapproval, criticism, or social ostracism. Social reinforcement plays a crucial role in shaping social norms, transmitting cultural values, and maintaining social order within groups and societies.

Examples of social reinforcement include:

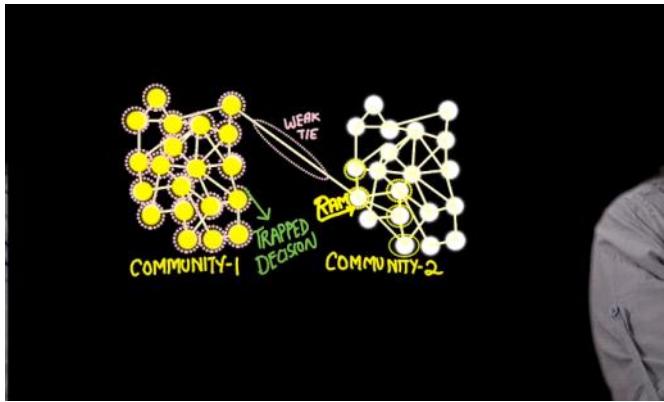
1. Praise and encouragement from peers or authority figures for desired behaviors, such as academic achievement or prosocial actions.
2. Social acceptance and inclusion in a group for conforming to group norms or expectations.
3. Peer pressure or social disapproval for engaging in behaviors that deviate from social norms or are considered undesirable.
↓
4. Rewards or recognition from a community or society for contributing to collective goals or values.

2.20



3:03 / 13:54

Impact of Communities on Diffusion

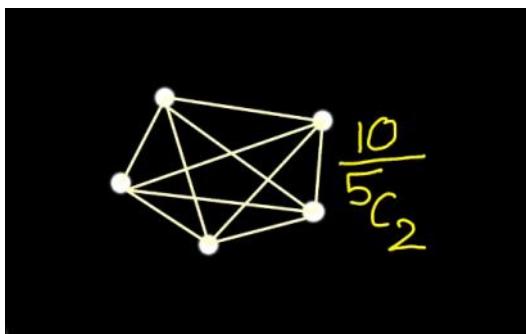
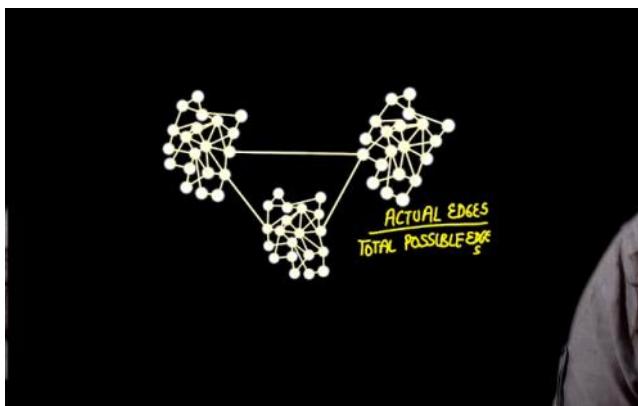


for you to adopt this new innovation. So, there is only the starting trouble for any product or any innovation to come in the market. We only need a few people to adopt it in the beginning and then rest of the diffusion is kind of easy, people look at each other and start following as we discussed and this innovation spreads in the network.

So, how do we overcome this starting trouble? So, one way of overcoming this starting trouble is to increase the payoff associated with your product. So, as payoff increases, people see that if they adopt it they are going to get a lot of advantage. So, probably initially people will be likely to adopt it more, but that might always not be possible. So, the second very nice option is to use the key people in the network.

So, instead of trying to advertise your product or your innovation to every single person in the population, you carefully choose a few key people and infect them with your idea or your innovation. What am I saying? I am saying that let us say you want a new bike

Influencer marketing jisse kehte hai basically !!



This is called Density of a Community !!

So, to help you answer this question, I tell you I will not tell you I will revise a small story most of you would be knowing. So, there was this king who was about to die and he had a big kingdom which needs to be looked after when he dies. And he had these 3 sons who always used to fight each other, and he wanted to teach them a lesson. So, when he was trying, he was on the deathbed, he asked his sons to come with a bundle of sticks, wood sticks.

And then they come with a bundle of wood sticks and he gave one wood stick to each of them and asks them to break it. The first person tries breaking the wood stick it easily gets broken, second wood stick easily gets broken third one also, but then he gives them all the wood sticks together as a bundle and asks them to break it. But then it does not get broken. So, there is an important lesson that there is power in unity. What it has to do with our question?

So, do you see here that this community here, if the people here kind of do not like each other or not many people are friends with each other, then if somebody will say something from outside, probably they will believe. And. they probably they will adopt this idea, but if they are good friends with each other it will be very difficult to convince

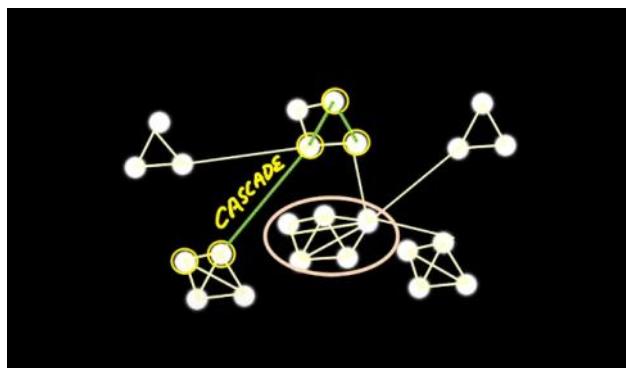
Page 11 / 11 - +

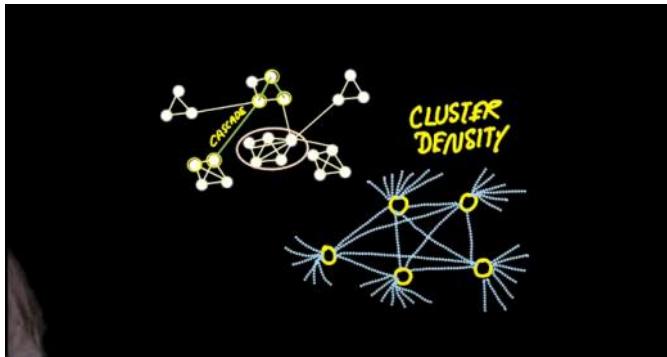
So, do you see here that this community here, if the people here kind of do not like each other or not many people are friends with each other, then if somebody will say something from outside, probably they will believe. And. they probably they will adopt this idea, but if they are good friends with each other it will be very difficult to convince them, it is very intuitive as well.

So, here are these bunch of 5 people who are really, really close friends and then they have decided to sit in library and do their assignment; no matter how hard you convince them to come outside with you and enjoy they probably will not come, but if their friendship was not that strong 1 or 2 of them would come outside with you. So, do you see here the role of the density of a community in diffusion? Higher the density the difficult it is for an idea to be injected in this community.

SUMMARY : HIGHER THE DENSITY : MORE DIFFICULT IT IS TO DIFFUSE A IDEA FROM OUTSIDE WORLD INTO THIS NETWORK !! { VIA WEAK TIES OFC }

SO PAYOFF AND DENSITY OF NETWORK ARE THE 2 FACTORS DISCUSSED TILL NOW THAT DECIDE THE AMOUNT TILL WHICH A IDEA WILL GET DIFFUSED INTO THE NETWORK !!





threshold value q . So, here we are assuming that every node is having the same threshold value q , I will come in on it later. So, every node in every cluster is having this threshold value q .

Now the next important thing as we know which needs to be captured is like we call the density of the community or the amount of unity.

So, here we defined something which is called the density of a cluster. What is the density of a cluster? So, I will give you a one liner let us see. So, the density of a cluster is we say that the density of a cluster is D , if you look at every node in this cluster and at least D fraction of these nodes friends is in this cluster itself. It might be a bit difficult let us look at it with an example.

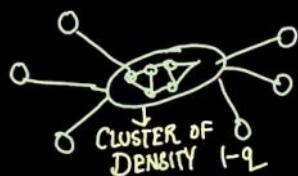
So, let us say that here is this cluster of 5 people. So, there are these 5 people in this cluster and let us say I say that the density of this cluster is 0.3 if I look at every person in this cluster and 30% of the friends of this person is in this same cluster. So, let us say that in this cluster everybody is having 10 friends.

So, I look at this node A out of these 10 friends 3 should be inside this cluster. For B out of these 10 friends 3 should be inside this same cluster and so on for the rest 3 of the nodes. So, that is what we mean by density of a cluster. So, I hope the definition is clear now.

Page 4 / 8 — Q +

Cascading behavior in social networks is when people influence each other's decisions and behavior. Cascades are a network phenomenon where small local shocks can lead to widespread innovations, strikes, fads, or power failures. ^

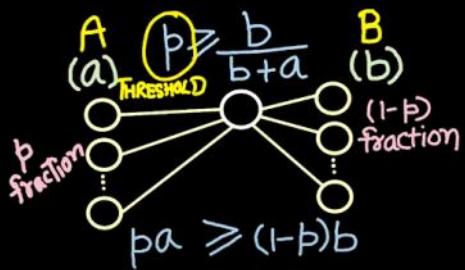
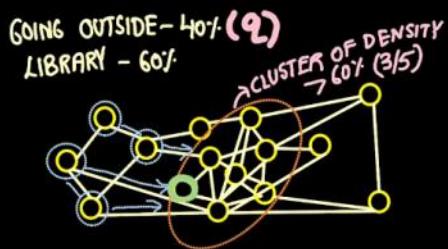
THRESHOLD OF ADOPTION = Q



Imp statement :

IF THE THRESHOLD OF ADOPTION OF ALL THE NODES == Q

THEN : THE FULL CASCADE CANT HAPPEN IF : THERE IS A CLUSTER WITH A CLUSTER DENSITY : $\geq 1-Q$



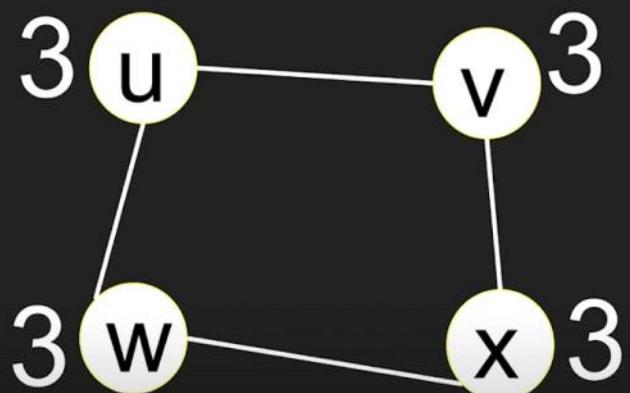
So idea wahi hai : agar is green colored node in above to above figure : see its friends : to unme se > 1-q i.e >60% i.e >threshold for library ne libarary jane ka faisla kiya hai therefore : acc to more payoff there : it will not go outside and similarly for other nodes of this cluster and that is why :the idea will not propogate further / fully in network !!

Cluster density = basically the fraction of friends of the given node that are part of the same cluster !!

Incomplete Cascade
 \Rightarrow Cluster of density
 $> (1-q)$

Basically : reverse bhi true hota hai ! { again proof is simple enough !!}

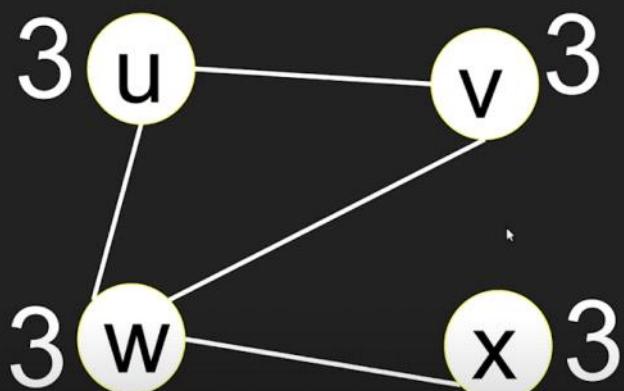
Deciding Whether collective action will be taken up based on the threshold values :



◀ ▶ ▶ ⏪ 10:37 / 11:23

Knowledge, Thresholds and the Collective Action

Reason for no collective action { revolt in this case for eg} is bcz : v and w don't know about the threshold of each other !!

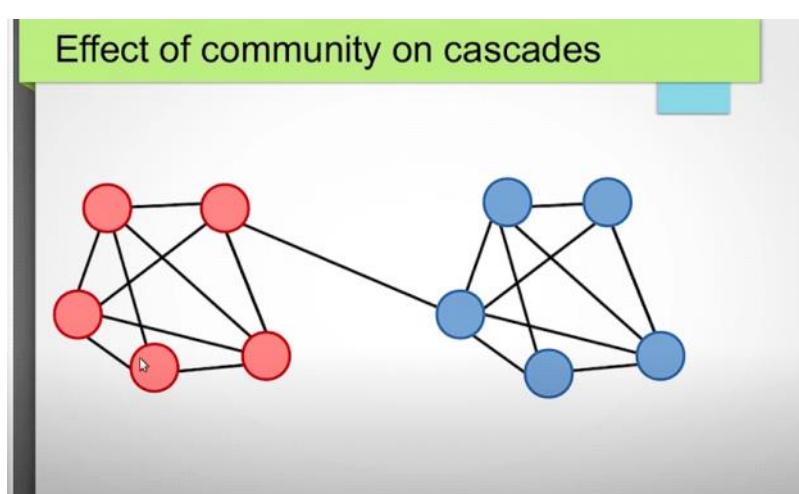
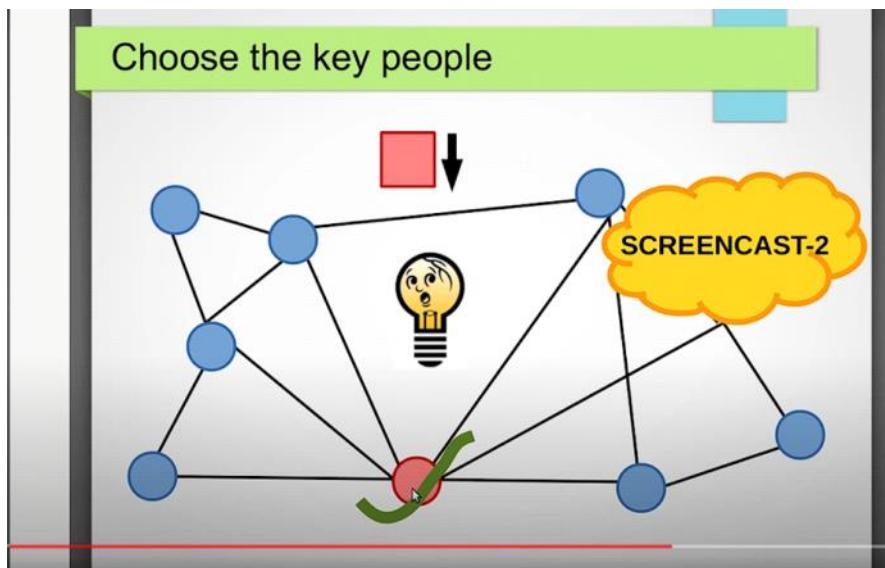
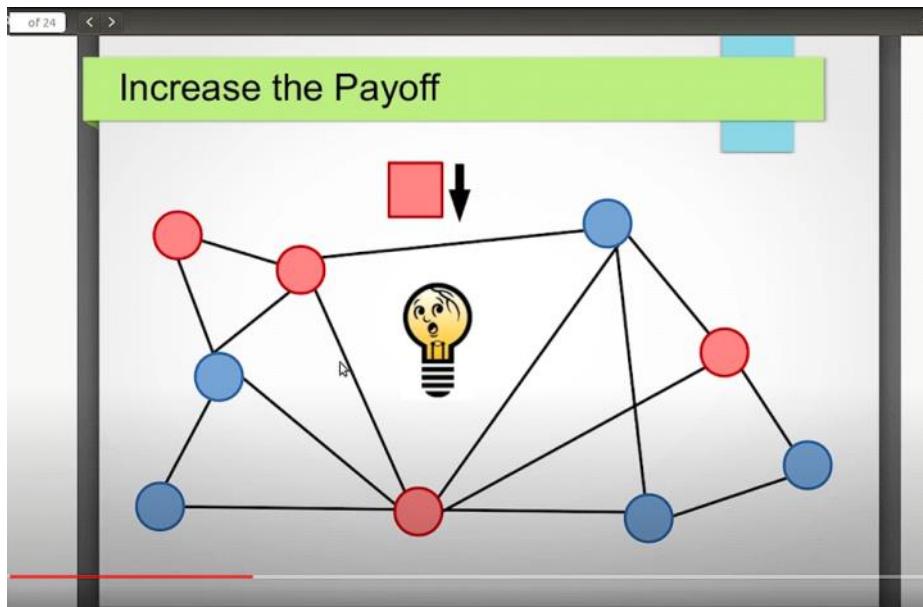


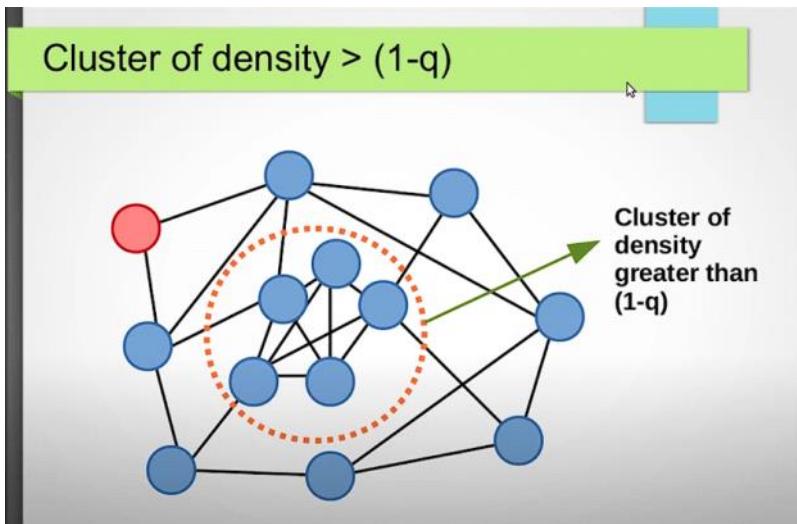
YES !! HERE COLLECTIVE ACTION WILL TAKE PLACE HERE !!

BASICALLY : BASED ON YOUR NEIGHBOURS ONLY : AAP : DECISION TO TAKE ACTION OR NOT LOGE !!!

MAJOR IDEAS OF WEEK 7 :

How to diffuse / cascade an idea over the network :





```

import networkx as nx
G=nx.erdos_renyi_graph(10,0.5)
nx.write_gml(G,'random_graph.gml')

```

0.5 is the probability with which an edge should be present !!

Erdos renyi graph { vvvimp !! }

```

G=nx.read_gml('random_graph.gml')
set_all_B(G)
list1=[3,7]
set_A(G,list1)
colors=get_colors(G)
nx.draw(G,node_color=colors, node_size=800)
plt.show

```

```

def set_A(G,list1):
    for each in list1:
        G.node[each]['action']='A'

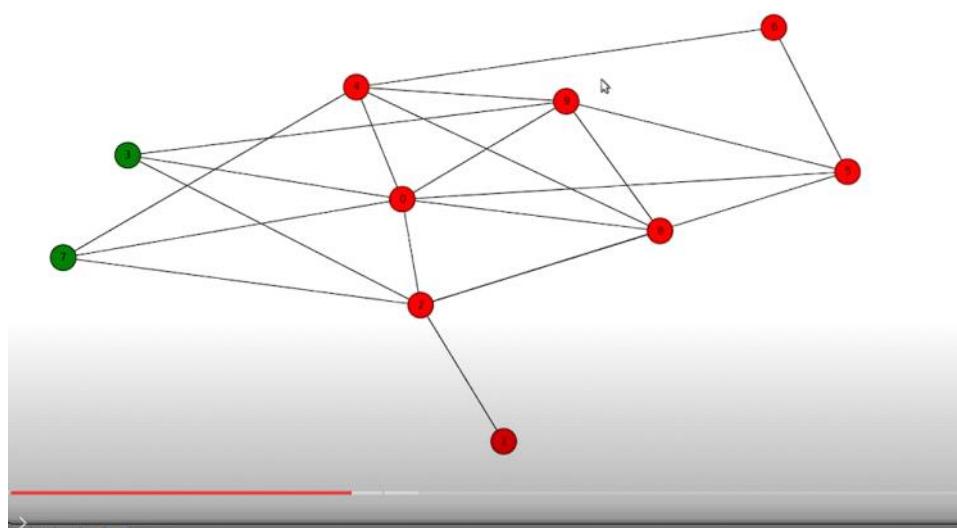
def get_colors(G):
    list1=[]
    for each in G.nodes():
        if G.node[each]['action']=='B':
            list1.append('red')
        else:
            list1.append('green')
    return list1

G=nx.read_gml('random_graph.gml')

```

Attribute Action >

Python Tab Width: 8 Ln



```

*Base_code.py x

def recalculate_options(G):
    dict1={}
    # Payoff(A)= a = 4
    # Payoff (B) = b = 3
    a=4
    b=3
    for each in G.nodes():
        num_A= find_neigh(each, 'A',G)
        num_B= find_neigh(each, 'B',G)
        payoff_A=a*num_A
        payoff_B=b*num_B
        if payoff_A>=payoff_B:
            dict1[each]='A'
        else:
            dict1[each]='B'

    return dict1

def find_neigh(each,c,G):
    num=0
    for each1 in G.neighbors(each):
        if G.node[each1]['action']==c:
            num=num+1
    return num

```

```

def reset_node_attributes(G,action_dict):
    for each in action_dict:
        G.node[each]['action']=action_dict[each]

```

```

base_code.py x [list1]
set_A(G,[list1])

colors=get_colors(G)

nx.draw(G,node_color=colors, node_size=800)

plt.show()

action_dict=recalculate_options(G)

reset_node_attributes(G,action_dict)

colors=get_colors(G)

nx.draw(G,node_color=colors, node_size=800)

plt.show()

```

Saving file '/home/yayati/base_code.py'...

```

num=num+1
return num

def recalculate_options(G):
    dict1={}
    # Payoff(A)= a = 4
    # Payoff (B) = b = 3
    a=10
    b=5
    for each in G.nodes():
        num_A= find_neigh(each,'A',G)
        num_B= find_neigh(each,'B',G)
        payoff_A=a*num_A
        payoff_B=b*num_B
        if payoff_A>=payoff_B:
            dict1[each]='A'
        else:

```

2:24 / 14:30 Python Tab Width: 8

Coding the First Big Idea - Increasing the Payoff

[Take Notes](#) [Public Note](#)

Social Networks
NPTEL-NOC IITM - 97/171

BY INCREASING THE PAYOFF WE WERE ABLE TO CASCADE THE BEHAVIOUR THROUGHOUT THE NETWORK !!

```

import networkx as nx
import random

def create_first_community(G):
    for i in range(0,10):
        G.add_node(i)
    for i in range(0,10):
        for j in range(0,10):
            if i<j:
                r=random.uniform(0,1)
                if r<0.5:
                    G.add_edge(i,j)

```

```
for i in range(11,20):
    G.add_node(i)
for i in range(11,20):
    for j in range(11,20):
        if i < j:
            r = random.uniform(0,1)
            if r < 0.5:
                G.add_edge(i,j)

G = nx.Graph()
create_first_community(G)
create_second_community(G)

nx.draw(G)
plt.show()
```

4) In the context of a cascade of adoptions of a behavior/product 'X' within a network, 1 which statement is NOT POSSIBLE? Consider a chain reaction of switches from 'Y' to 'X'.

- Eventually all nodes adopt 'X' over time.
- The cascade initiates but halt before all nodes transition to 'X'.
- At a point, every node reverts to 'Y'.
- Only the initial seed node transitions to 'X'.

No, the answer is incorrect.

Score: 0

Accepted Answers:

At a point, every node reverts to 'Y'.

1) Consider the modeling of information cascade in a graph G, where everyone starts **0 points** with B and there is a small set S of early adopters of A. Given following two statements.

S1: Nodes in S keep using A no matter what payoffs tell them to do

S2: Nodes outside S keep using B no matter what payoffs tell them to do.

Choose the correct option.

- Both S1 and S2 are true
- S1 is true but S2 is false
- S2 is true but S1 is false.
- Both S1 and S2 are false.

No, the answer is incorrect.

Score: 0

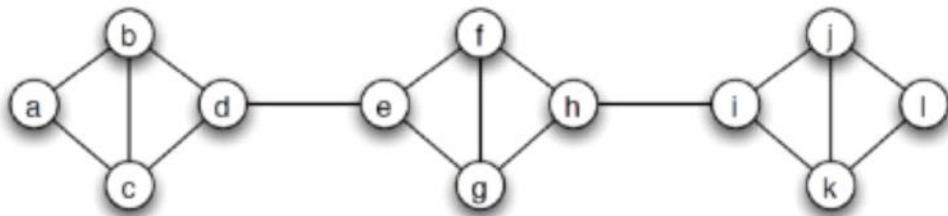
Accepted Answers:

S1 is true but S2 is false

Both these questions lead to a conclusion ki jo : pehle infect honge : wo

revert back nahi karenge chahe kuch bhi ho jaye !!

6) What is the cluster density of each of the four node cluster?



- 1/3
- 2/3
- 1
- 4/3

Yes, the answer is correct.

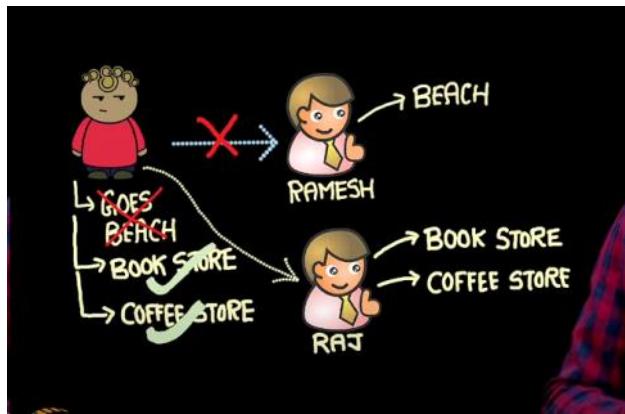
Score: 1

Accepted Answers:

2/3

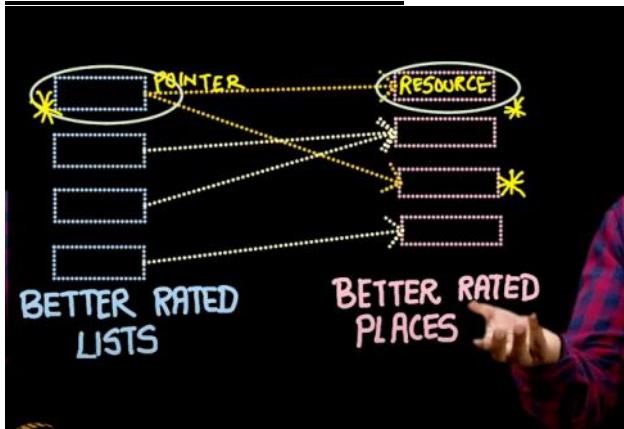
In case of unequal fraction of friends: min wala consider karlo simple !!

WEEK 8 :





WHATEVER SUGGESSION A PERSON GIVES IF I LIKE IT I NOT ONLY GIVE STAR TO THAT PLACE BUT ALSO TO THAT : PERSON WHO RECOMMENDED THAT PLACE AND THEREFORE IT IN FUTURE HE RECOMMENDS ME ANOTHER NEW / UNEXPLORED PLACE : IT WILL BE VALUED MORE BASED ON THE FACT THAT THE RECOMMNDATION IS COMING FROM A PERSON JISKI PURANI SUGGESSIONS MUJHE PASAND AAYI THI !!

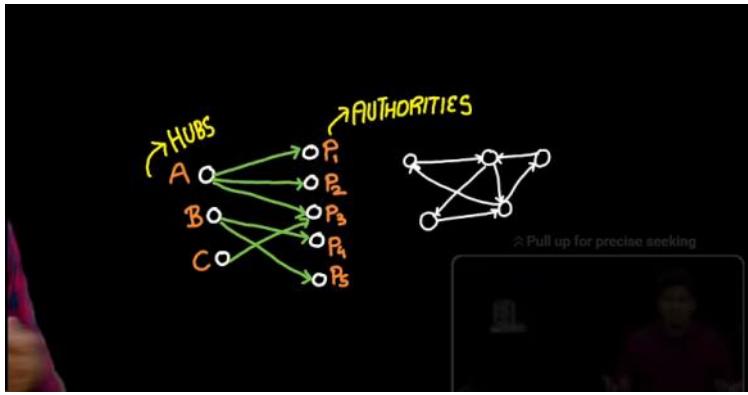


Principle of Repeated Improvement

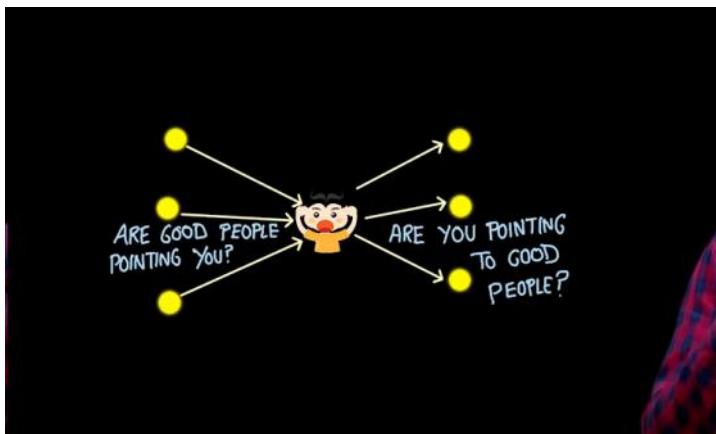
See : simple sa funda : ek ki improvement means usse linked cheez ki improvement and vice versa !!

So : if : a person recommended a place that I find good : that is give it a * / star
 Then : it will not only lead to rating improvement of the place but that of the : person also

And agar koi nayi jagah wo karega suggest to uske * badenge if recommend karne ke stars / rating me improvement hogi !!!

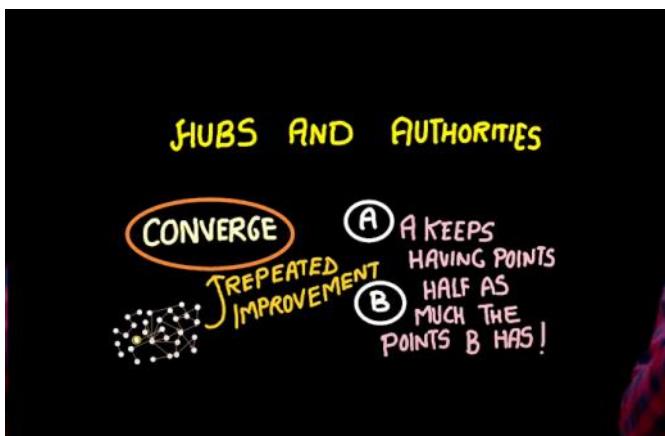


Recomenders : are basically called hubs !! And the recommendations are called authorities !!



THAT IS WHY : PEOPLE IN A NETWORK CAN ACT BOTH AS HUBS AND AUTHORITIES !!

And if you can observe : something similar can be : applied to find page ranks also !!! : i.e rank of a page will increase if : good pages point to it and : if it points to good web pages / informational ones ... !!!



SO idea is that : as you keep allocating the points / stars to nodes via this repeated improvement method then the ratio of points they have will converge after some time : linking it with web pages : idea is that : rank of page can be deterministically determined via this !!

2.70

PageRank

NPTEL

PageRank Revisited - An example

Social Networks

PageRank

| | A | B | C |
|---------------|---------------|---------------|---------------|
| I | $\frac{1}{3}$ | $\frac{1}{3}$ | $\frac{1}{3}$ |
| II | $\frac{1}{6}$ | $\frac{1}{3}$ | $\frac{1}{2}$ |
| III | $\frac{1}{6}$ | $\frac{1}{2}$ | $\frac{1}{3}$ |
| ↓ Converges ↓ | | | |

NPTEL

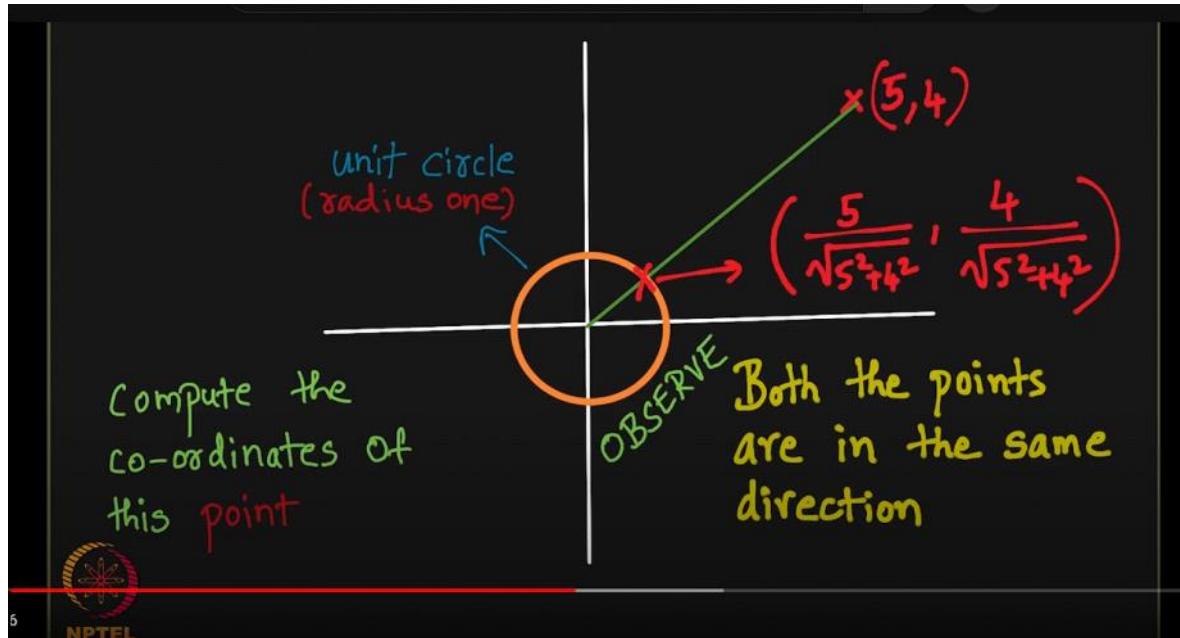
PageRank Revisited - Convergence in the Example

THESE VALUES DO CONVERGE { YE HAMNE PREV WEEKS ME BHI DEKHA THA !! }

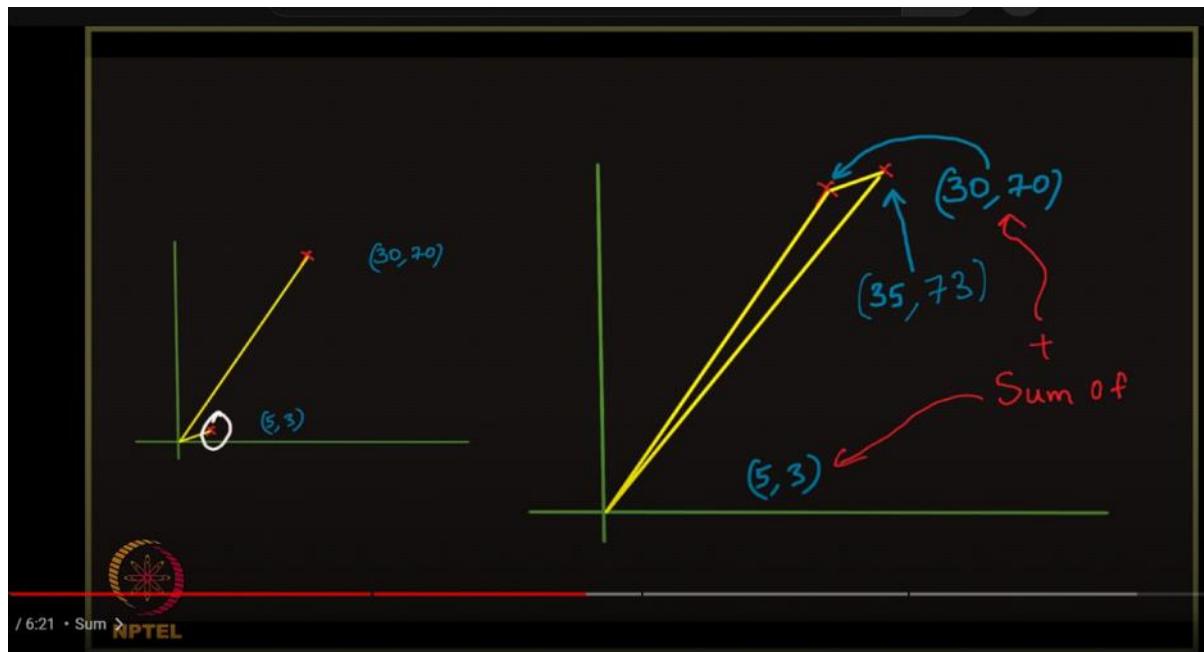
5:18 / 5:26

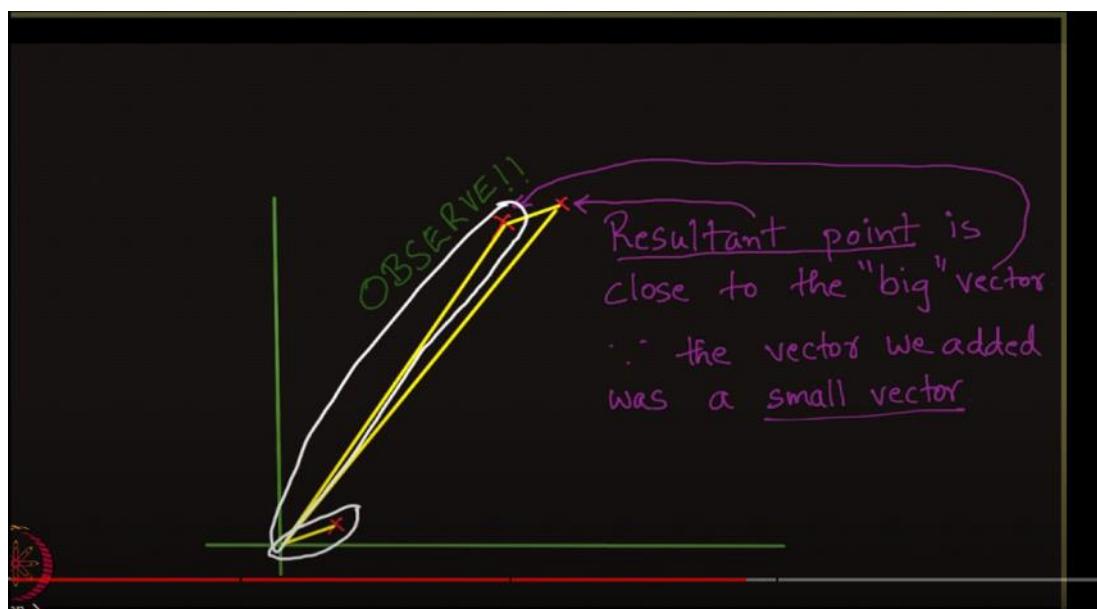
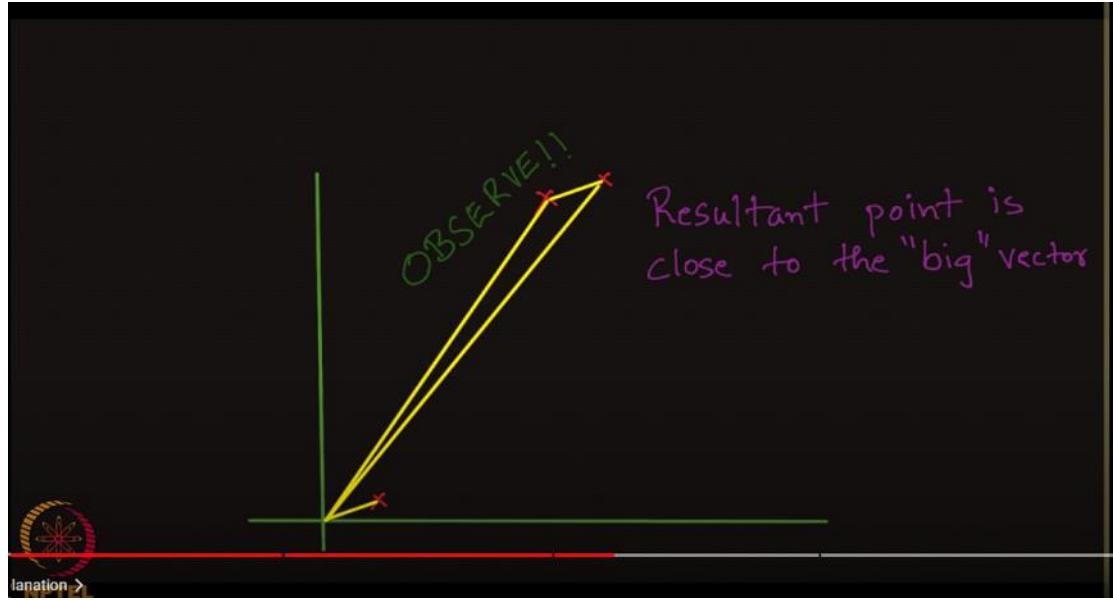
Sheet1

| | A | B | C | D | E | F |
|----|--------------|--------------|----------------|---|---|---|
| 57 | 0.2000000005 | 0.3999999985 | 0.400000001 | | | |
| 58 | 0.1999999993 | 0.400000001 | 0.399999998 | | | |
| 59 | 0.2000000005 | 0.3999999998 | 0.399999998 | | | |
| 60 | 0.1999999999 | 0.3999999998 | 0.400000004 | | | |
| 61 | 0.1999999999 | 0.4000000004 | 0.399999998 | | | |
| 62 | 0.2000000002 | 0.3999999998 | 0.400000001 | | | |
| 63 | 0.1999999999 | 0.4000000001 | 0.400000001 | | | |
| 64 | 0.2 | 0.4000000001 | 0.399999999 | | | |
| 65 | 0.2 | 0.3999999998 | 0.400000001 | | | |
| 66 | 0.2 | 0.4000000001 | 0.4 | | | |
| 67 | 0.2 | 0.4 | 0.4 CONVERGES! | | | |
| 68 | 0.2 | 0.4 | 0.4 | | | |
| 69 | 0.2 | 0.4 | 0.4 | | | |
| 70 | 0.2 | 0.4 | 0.4 | | | |
| 71 | 0.2 | 0.4 | 0.4 | | | |
| 72 | | | | | | |
| 73 | | | | | | |
| 74 | | | | | | |
| 75 | | | | | | |
| 76 | | | | | | |
| 77 | | | | | | |
| 78 | | | | | | |



Repeated multiplication of 2 vectors and normalization : lead to convergence
{ result 1 }





Moral?

When we add a "big" vector to a "small" vector
The resultant is close to the direction of the "big" vector.

Moral?

When we add a "big" vector
to a "small" vector

Big/Small in
amplitude

Distance from
origin!

The resultant is close to the
direction of the "big" vector.

PRE -REQ - ②

Big + Small = Something close to
big, both in direction
as well as amplitude

WHAT DOES ONE OBSERVE?

No matter what vector we choose,
we land with the same vector (direction)



i.e matrix mul me 1st vector fix karke 2nd vector kuch bhi le to resultant vector is
in same direction !!

Recollect!

Heard of Eigen vectors/values
of a matrix?

$$\textcircled{1} \quad A v = \lambda v$$

(2. 2×2 A')

③ Any vector
 $v = \alpha v_1 + \beta v_2$

v_1 v_2

Linearly Independent



How/why?

A matrix Some vector Eigen values

$$A \cdot v = A(\lambda_1 v_1 + \lambda_2 v_2) =$$

Eigen vectors

$$= \lambda_1 A(v_1) + \lambda_2 A(v_2) = \lambda_1^2 v_1 + \lambda_2^2 v_2$$

◀ ▶ ⏪ ⏹ 7:14 / 16:52

Convergence in Repeated Matrix Multiplication- The Details

Social Networks

$$A^K(v) = \underbrace{\lambda_1^K v_1}_{\text{Amplitude}} + \underbrace{\lambda_2^K v_2}_{\text{Direction}} \quad (\text{Say } \lambda_1 > \lambda_2)$$

Note: $2^2, 3^2 \rightarrow$ One more than twice
 $2^3, 3^3 \rightarrow$ More than thrice
 $2^6, 3^6 \rightarrow$ Several folds
 So much that $2^{100}/3^{100} \approx 0$.



$$A^K(v) = \lambda_1^K v_1 + \lambda_2^K v_2 \quad (\text{Say } \lambda_1 > \lambda_2)$$

Amplitude Direction

$$\Rightarrow A^K(v) = \boxed{\lambda_1^K v_1} + \boxed{\lambda_2^K v_2}$$

Big Small



$$\Rightarrow A^K(v) = \lambda_1^K v_1 + \lambda_2^K v_2$$

Big Small

$$\Rightarrow A^K(v) \approx \lambda_1^K v_1$$

It is in the direction of v_1

{ PRE REQUISITE 2 }

$$\Rightarrow A^K(v) = \lambda_1^K v_1 + \lambda_2^K v_2$$

Random vector Big Small

$$\Rightarrow A^K(v) \approx \lambda_1^K v_1$$

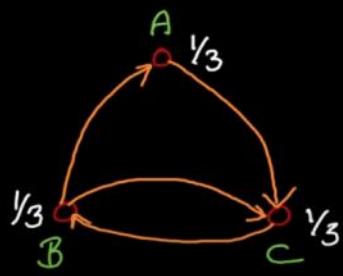
It is in the direction of v_1

Independent of v



THAT IS WHY : IT CONVERGES TO A SAME VALUE !! { VVVIMP }

Process is a matrix multiplication



$$A \begin{bmatrix} A & B & C \\ 0 & y_2 & 0 \\ 0 & 0 & 1 \\ 1 & y_2 & 0 \end{bmatrix} \begin{bmatrix} y_3 \\ y_3 \\ y_3 \end{bmatrix} = \begin{bmatrix} y_6 \\ y_3 \\ y_3 + y_6 \end{bmatrix}$$

Can this process be captured as a matrix multiplication process.



- 1) For a given network, start from $y_3, y_3, y_3 \rightarrow$ we observe it converges
- 2) We noted that this iterations can be seen as matrix multiplication.
To be shown next
- 3) Why does it converge?
- 4) PageRank & all the above?

Such a matrix
is called a
Markov matrix.

Note

$$\begin{pmatrix} 0 & 0 & y_2 & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 1 & 0 & y_2 \\ \frac{1}{2} & 0 & 0 & 0 \end{pmatrix}$$

Every column
sums to 1

Has a very interesting property

Highest Eigen value is 1



$$\Rightarrow A^k v = \lambda_1^k v_1 + \lambda_2^k v_2 + \dots + \lambda_n^k v_n$$

Less than 1



$$\begin{pmatrix} 0 & 0 & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 1 & 0 & \frac{1}{2} \\ \frac{1}{2} & 0 & 0 & 0 \end{pmatrix}$$

Has a very interesting property
Highest Eigen value is 1

$$\Rightarrow A^k v = \lambda_1^k v_1 + \lambda_2^k v_2 + \dots + \lambda_n^k v_n$$

Less than 1

$$\Rightarrow A^k v = v_1 \rightsquigarrow \text{Eigen vector corresponding to 1}$$


$$\begin{pmatrix} 0 & 0 & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 1 & 0 & \frac{1}{2} \\ \frac{1}{2} & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \frac{1}{4} \\ \frac{1}{4} \\ \frac{1}{4} \\ \frac{1}{4} \end{pmatrix} = \begin{pmatrix} \frac{1}{8} \\ \frac{3}{8} \\ \frac{3}{8} \\ \frac{1}{8} \end{pmatrix}$$

Observe
 $A\alpha = \alpha$

| | | | | | | | |
|---|---------------|---------------|---|---|---|---|------------------|
| a | $\frac{1}{4}$ | $\frac{1}{8}$ | - | - | - | - | $\frac{19}{100}$ |
| b | $\frac{1}{4}$ | $\frac{3}{8}$ | . | . | . | . | $\frac{1}{3}$ |
| c | $\frac{1}{4}$ | $\frac{3}{8}$ | - | . | . | . | $\frac{19}{50}$ |
| d | $\frac{1}{4}$ | $\frac{1}{8}$ | - | - | . | . | $\frac{9}{100}$ |

Same Values

QUIZ?

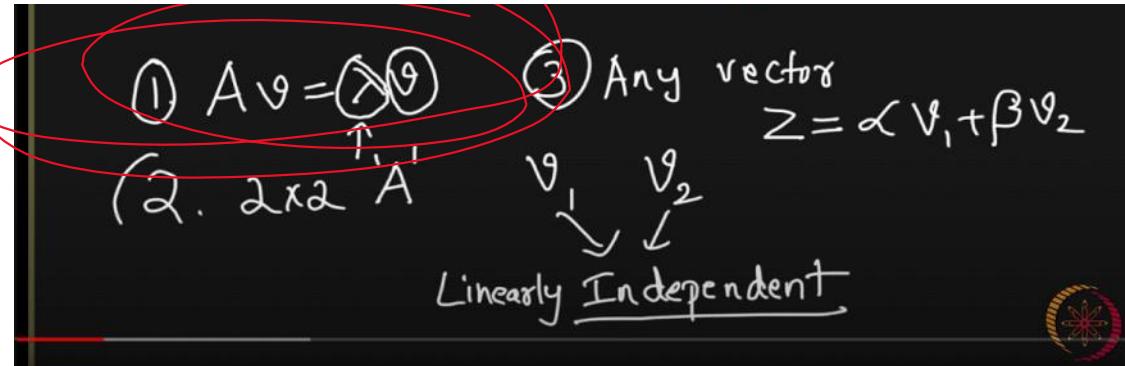


So basically idea is that : if it converges means : $A * (\alpha) = \alpha$

Recollect!

Heard of eigen vectors/values
of a matrix?

① $A \cdot v = \lambda v$ ③ Any vector



SO OBSERVE THAT : $A^{**K} * (\vec{v})$ { where \vec{v} is any random vector }

Becomes = \vec{v}_1 { lambda1 = 1 }

So : acc to this property : $A * (\vec{v}_1)$ { i.e next iteration } = lambda1 * (\vec{v}_1) = \vec{v}_1
 { VVVIMP }

AI overviews are experimental. Learn more ::

Listen

Here are some properties of Markov matrices:

Non-negative entries
 All entries in a Markov matrix are non-negative because probabilities cannot be negative.

Row sums equal to one
 Each row in a Markov matrix sums to one, which reflects the total probability of transitioning from a given state to all possible states. ^

SUPER DUPER IMP MARKOV MATRIX KE BAARE ME !!

the eigenvector, let's break it down.

- Matrix:** A matrix is a mathematical construct consisting of rows and columns of numbers. Matrices are used in various mathematical operations, including transformations in geometry, solving systems of linear equations, and representing data in computer science and engineering.
- Eigenvectors:** Eigenvectors are special vectors associated with a square matrix. When a matrix is multiplied by one of its eigenvectors, the resulting vector is a scaled version of the original eigenvector. The scaling factor is known as the eigenvalue. Formally, if A is a square matrix, \vec{v} is an eigenvector of A , and λ is the corresponding eigenvalue, then $A\vec{v} = \lambda\vec{v}$.
- Direction:** In linear algebra, vectors have both magnitude and direction. Changing the direction of a vector means altering the orientation of the vector without changing its length or magnitude. For example, if you have a vector pointing north, changing its direction might make it point northeast or northwest, but its length remains the same.

7) Which of the following statements are correct?

Statement I - Applying a matrix on its eigenvectors only changes the direction of the eigenvector.

Statement II - Eigenvectors of a matrix are linearly dependent on each other.

- I only
- II only
- Both
- None

No, the answer is incorrect.

Score: 0

Accepted Answers:

Both

VVIMP FUNDA !!

IN RESULTS KE BASE PE HI HAI YE !! :

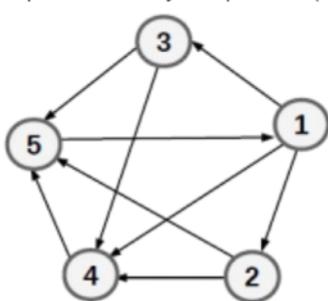
Recollect!

Heard of eigen vectors/values
of a matrix?

$$\begin{array}{ll} \textcircled{1} \quad A\vec{v} = \textcircled{2} \vec{v} & \textcircled{3} \text{ Any vector } \\ \textcircled{2}. \quad 2 \times 2 \quad A' & \vec{v}_1 \quad \vec{v}_2 \\ & \downarrow \quad \downarrow \\ & \text{Linearly } \underline{\text{Independent}} \end{array}$$



10) What is the score value of authority(a) and hub (h) respectively for the node 4 in the following figure after applying 1-step hub-authority computation (i.e. when k is 1)? Assume initial hub and authority of each node as 1.



- a(1) = 1, h(1) = 3
- a(1) = 3, h(1) = 1
- a(1) = 4, h(1) = 2
- a(1) = 2, h(1) = 4

Yes, the answer is correct.

Score: 1

Accepted Answers:

Vvvvimp result :

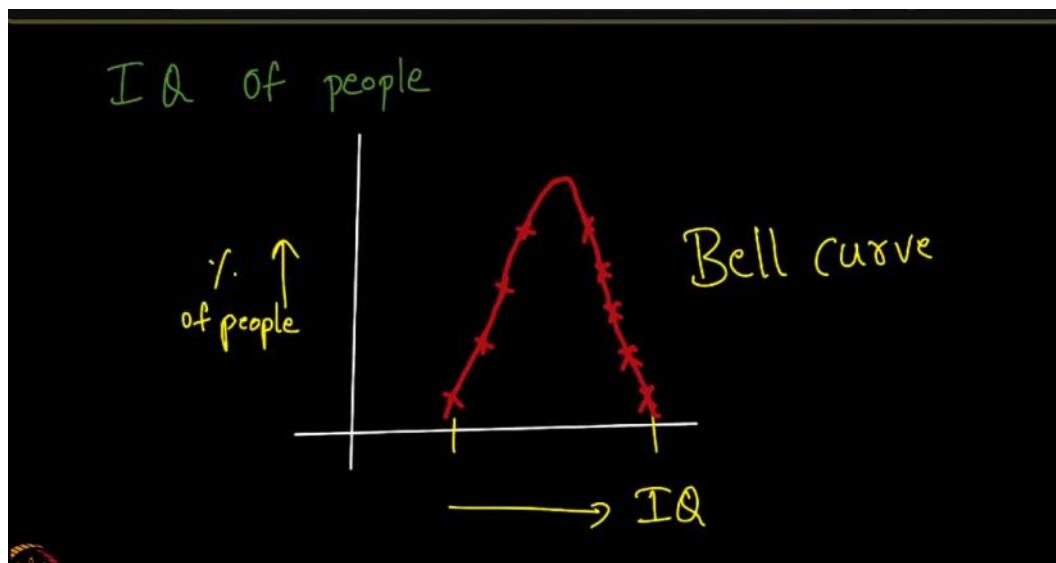
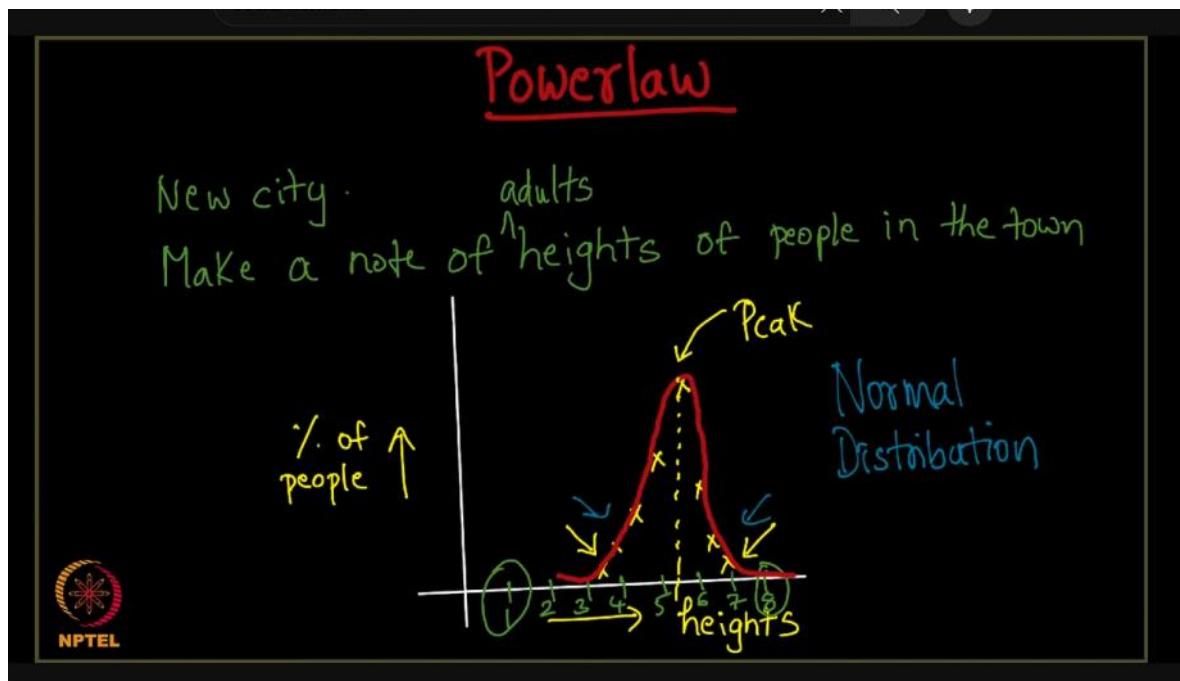
HUB SCORE = SUM OF AUTHORITY SCORE OF ALL THE POINTS

THAT IT POINT TO !! { OUT GOING LINKS }

AUTHORITY SCORE =SUM OF HUB SCORE OF ALL THE POINTS

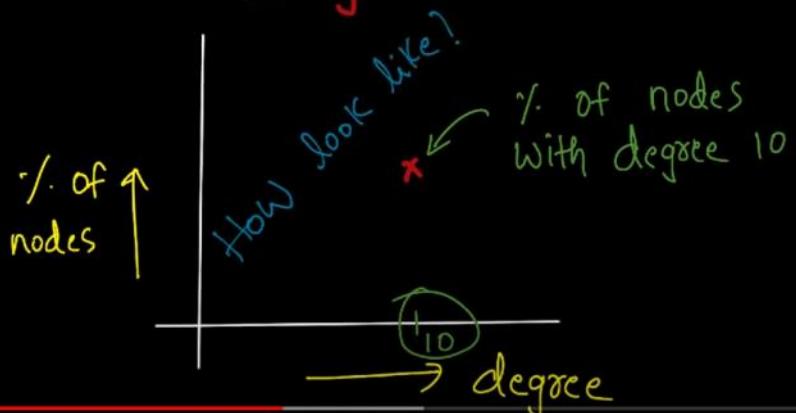
THAT POINT TO IT !!! {IN COMING LINKS }

WEEK 9 { VVIMP } :



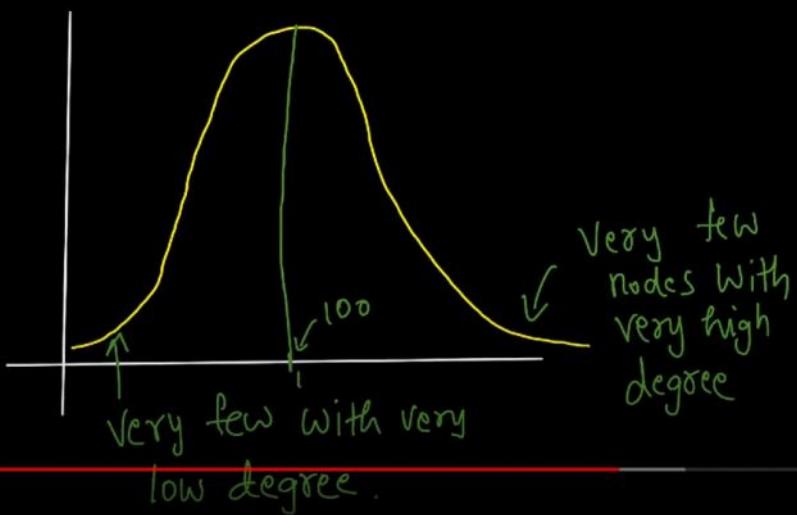
Observe in networks

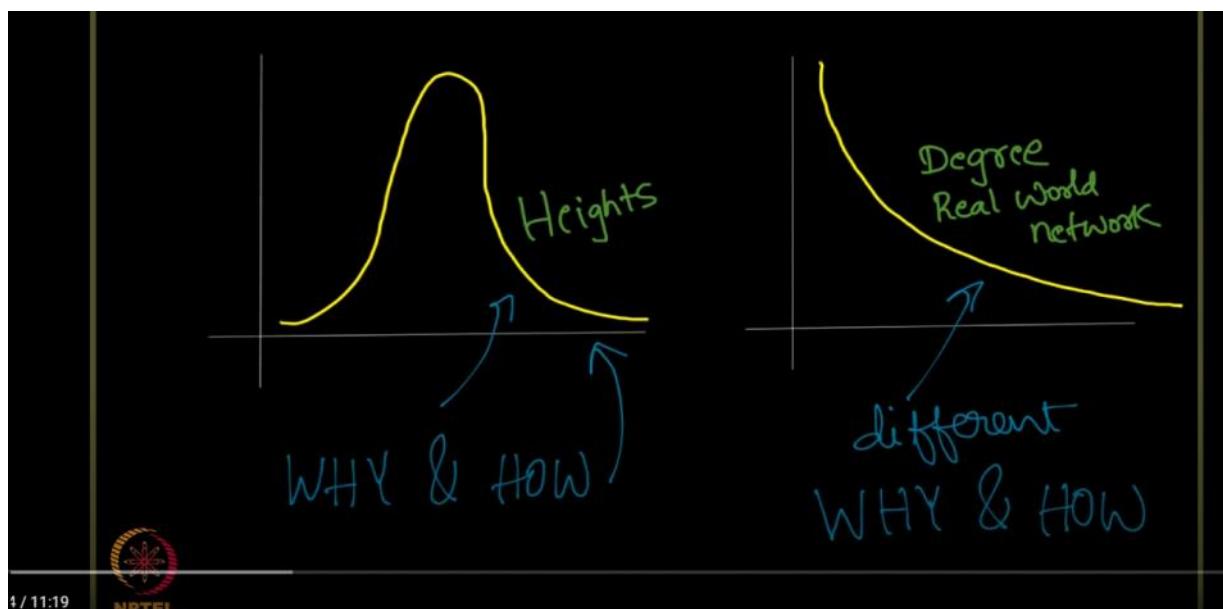
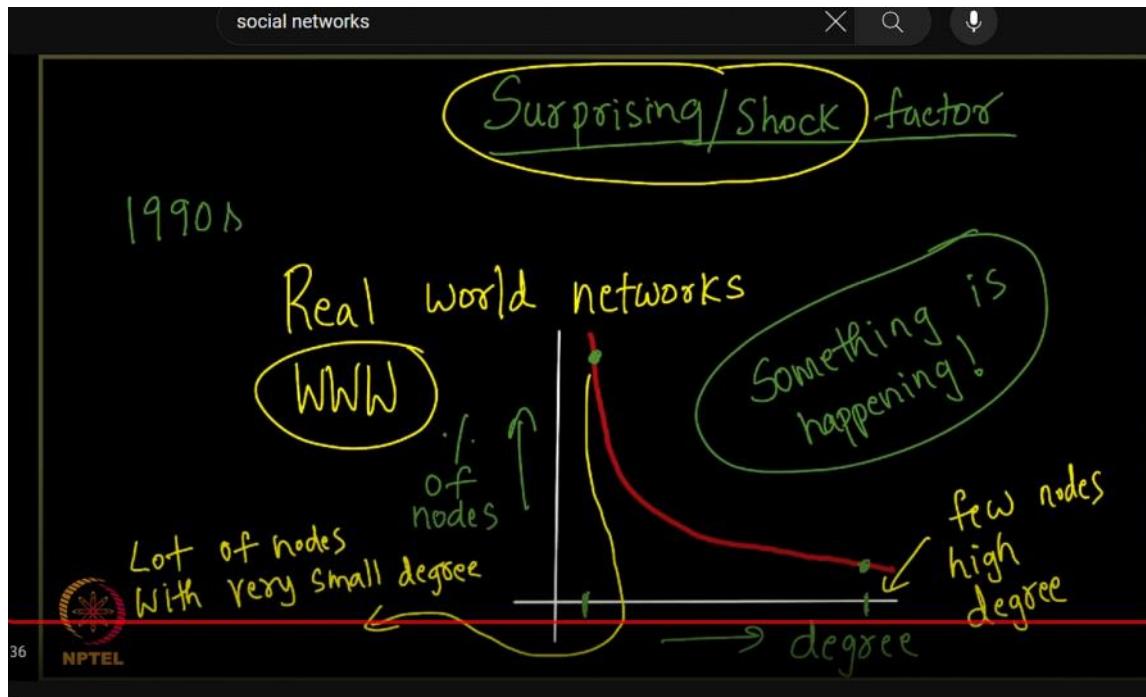
Degree distribution



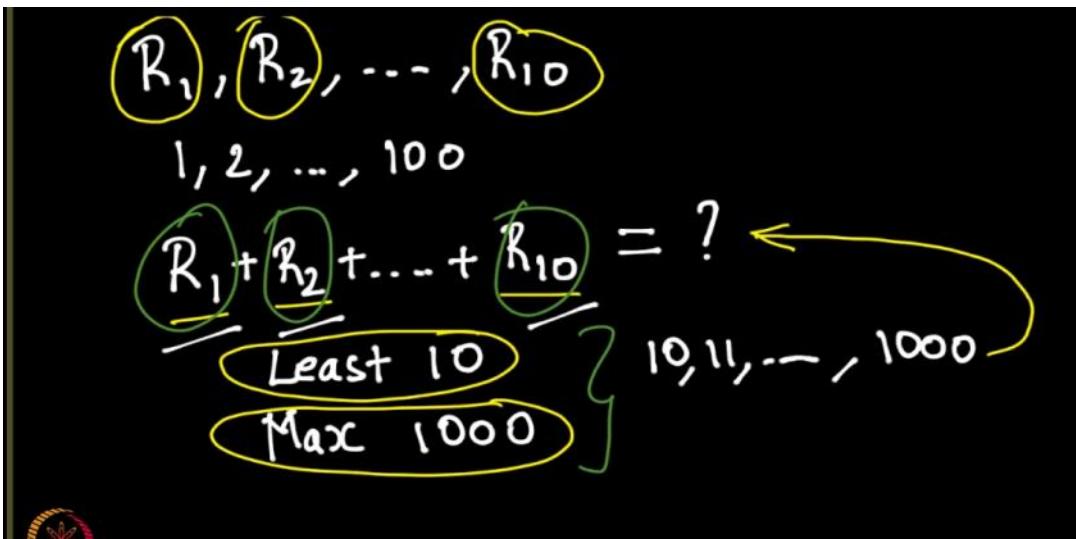
social networks

$\mathcal{G}(1000, 0.1)$

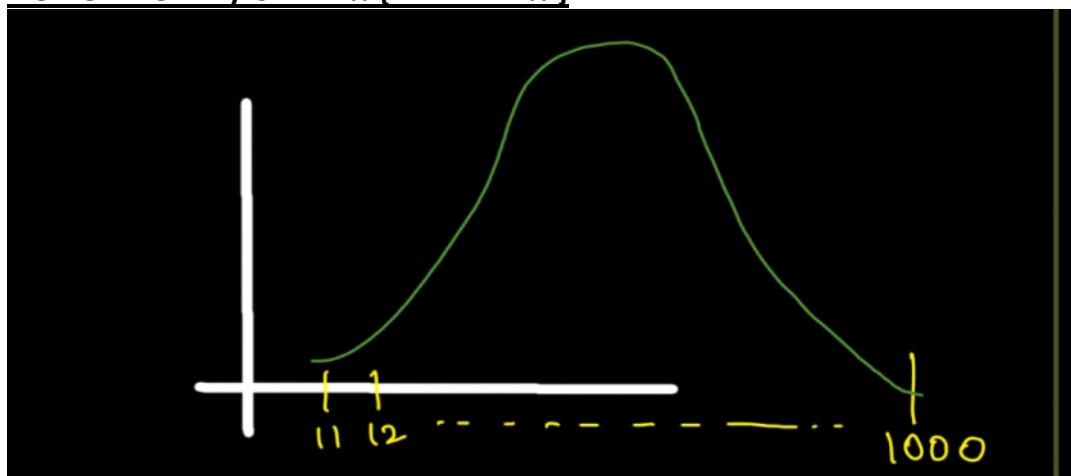




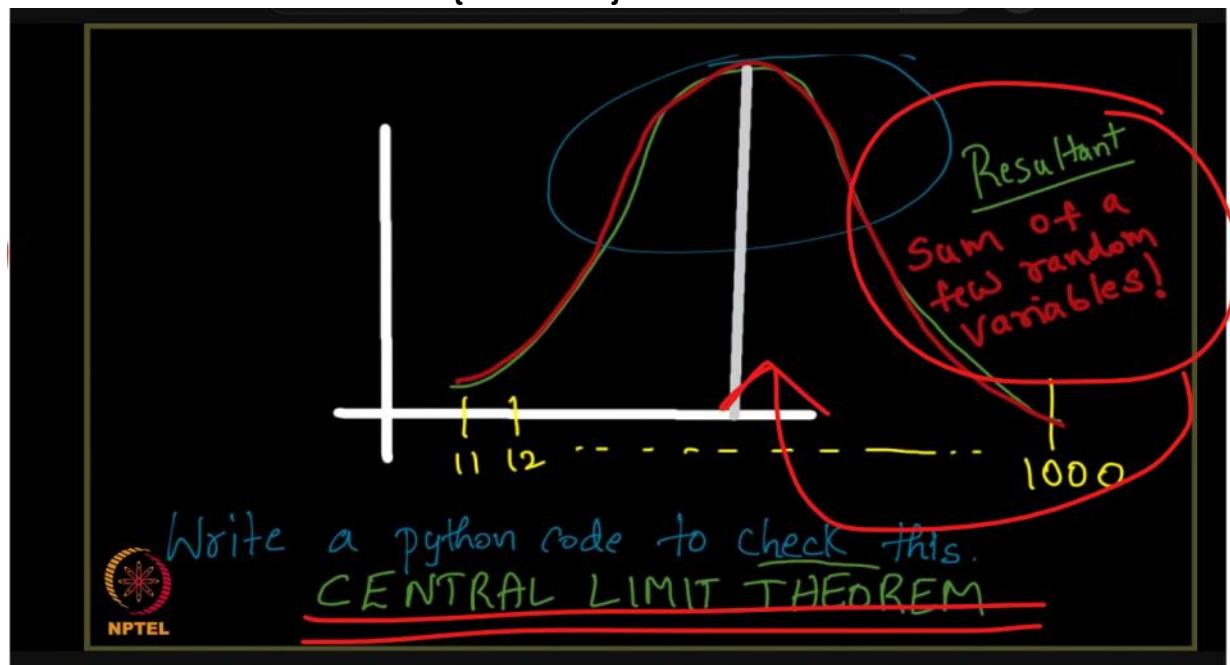
- 1) I pick randomly $\{1, 2, \dots, 100\} \rightarrow R_1$
- 2) You pick randomly $\{1, 2, \dots, 100\} \rightarrow R_2$
- ⋮
- 10) Picks $\{1, 2, \dots, 100\} \rightarrow R_{10}$



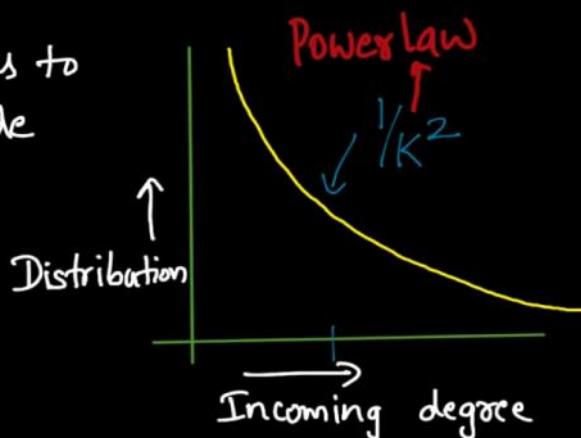
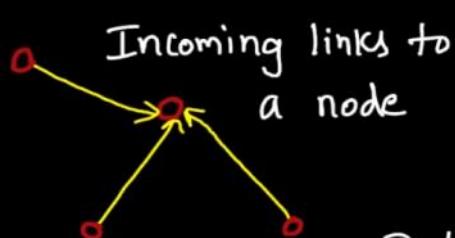
BUT BUT BUT : TWIST IS THAT : CHANCES OF APPEARING OF SUM = 10 , 11 IS NOT UNIFORM / SAME !! { VVIMP !! }



RATHER IT IS A BELL CURVE !!! { VVIMP !! }



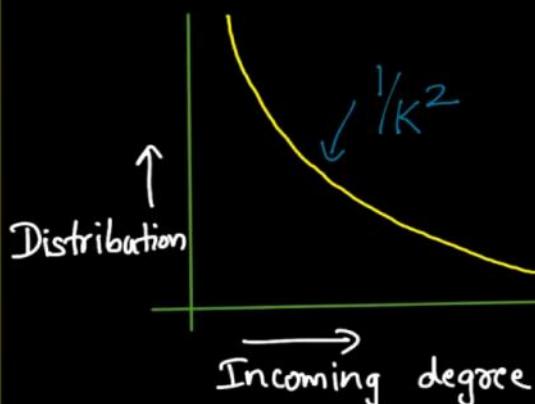
Graph of weblinks



NPTEL

Search

Difference:
Normal vs power law



Continuous

There are
nodes with
very high incoming
degree

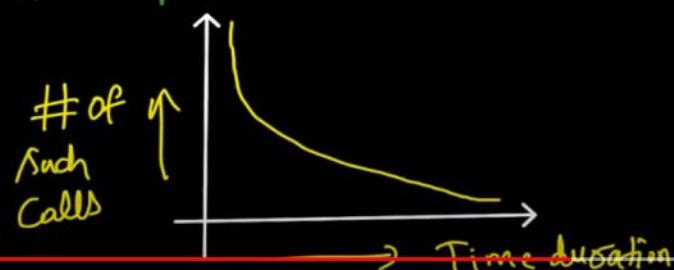


NPTEL

WWW-graphs

Power law is observed in many situations:

- 1) Distribution of telephone conversation duration.



TEI

Power law ↘ is observed in many situations:

- 1) Distribution of telephone conversation duration.
- 2) ↗ song downloads

Power law in WWW graph

Network: Powerlaw

$$f(k) = \frac{1}{k^2} \quad [2] \downarrow [3]$$

How to see if a network exhibit power law or not ?

$$f(k) = \frac{1}{k^\alpha}$$

for some α

$$f(k) = \frac{1}{k^\alpha}$$

$$\log f(k) = \log 1 - \log k^\alpha$$

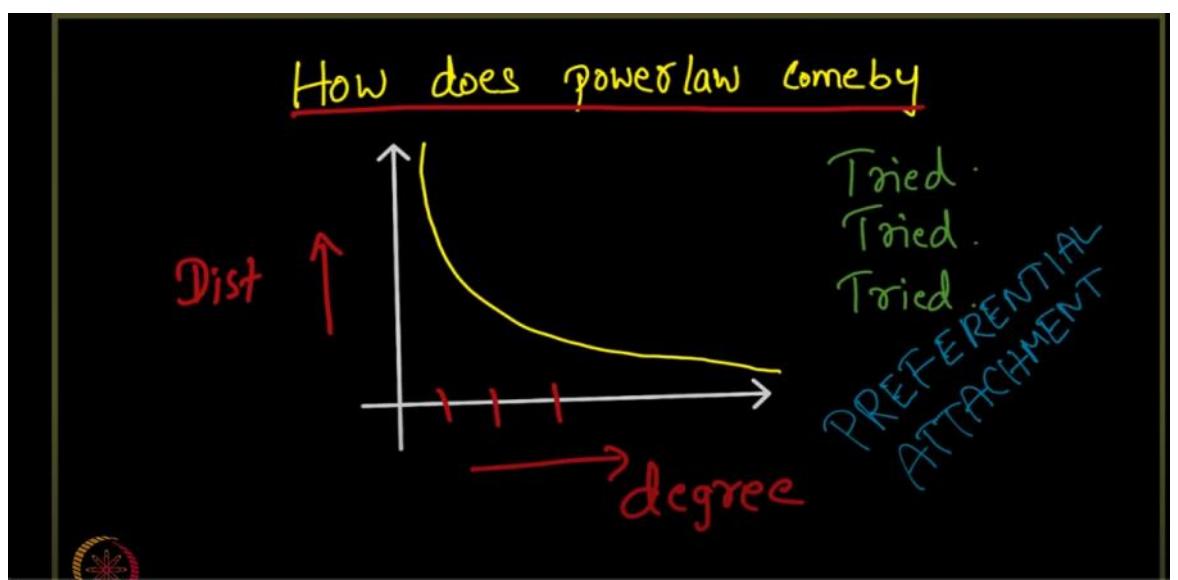
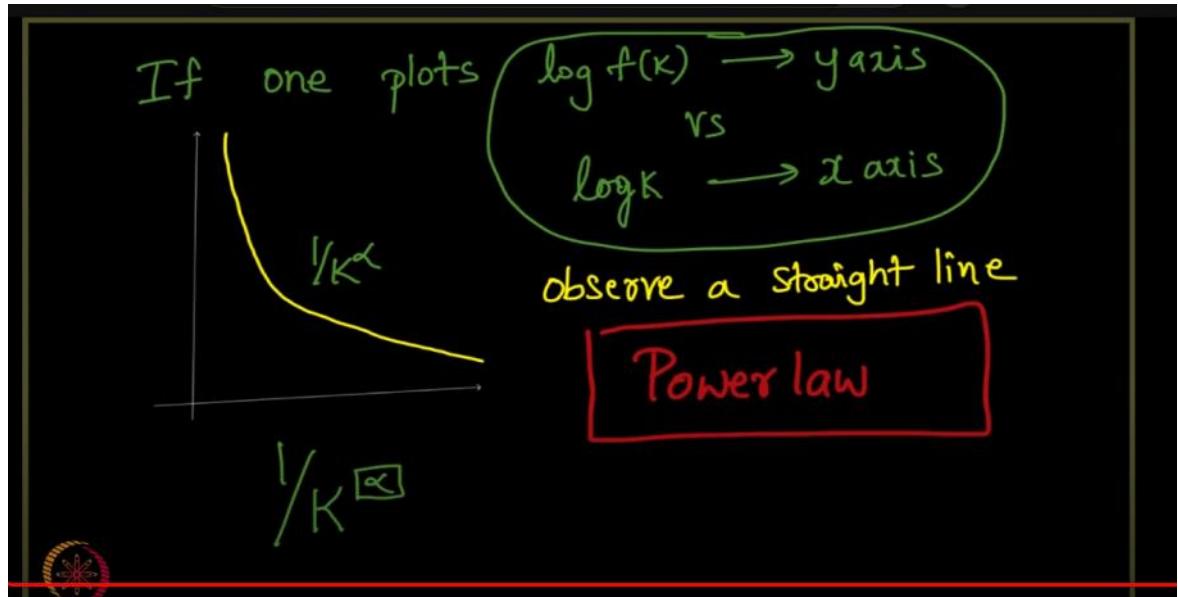
$$\Rightarrow \log f(k) = 0 - \alpha \cdot \log k$$

$$\log f(k) = (-\alpha) \cdot \log k$$

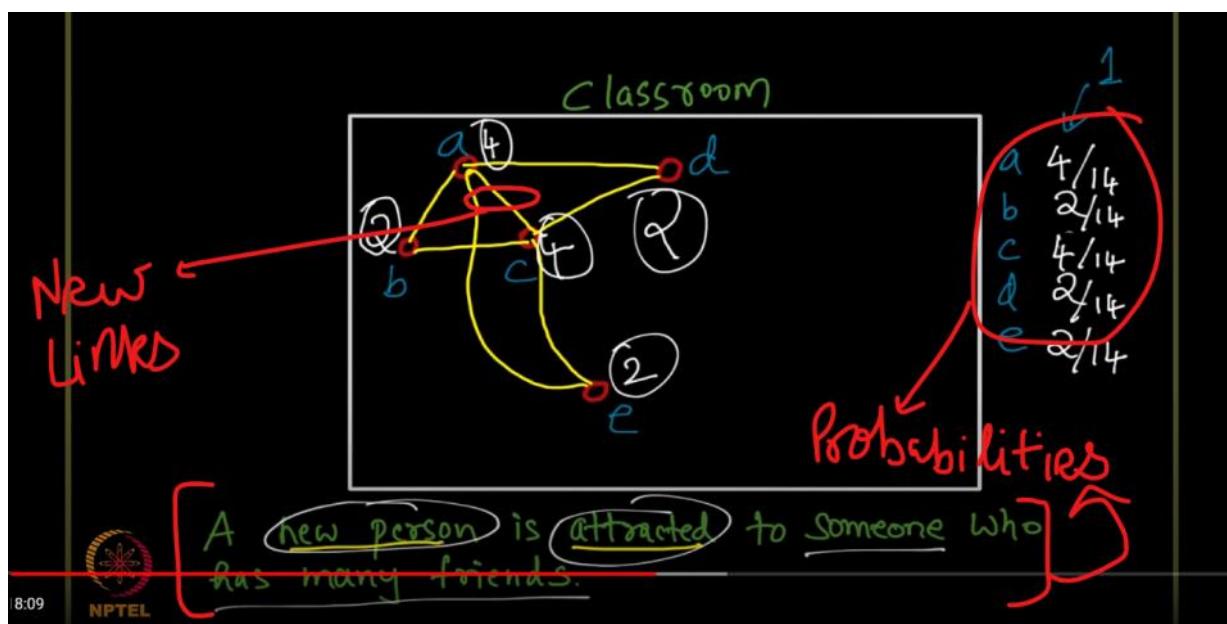
$$y = (-\alpha) \cdot x$$

9:33

Plotting log log graph of a given plot and you get a straight line then : that distribution follows a power law !!!



Imp hypothesis : PREFERENTIAL ATTACHMENT !!



Ideal model

| Node | Degree |
|------|--------|
| a | 5/18 |
| b | 2/18 |
| c | 4/18 |
| d | 3/18 |
| e | 2/18 |
| f | 2/18 |

A new person is attracted to someone who has many friends.

18:09 NPTEL

If I continue doing this for 1000 nodes, how does the resultant network look like?

(Power law)

A node with higher degree will attract more edges to it.

"Rich get richer".

2.60 17:37 / 18:09 NPTEL

Rich Get Richer Phenomenon

ISME AAPKA : VVHIGH NO OF ELEMENTS WILL HAVE : LOW DEGREE / WILL BE POOR AND HIGHER DEGREE ONES WILL : BE GETTING MORE HIGHER AND HIGHER { THIS GRAPH EXTENDS AS SHOW BEFORE }

THAT IS WHY : INDEGREE DISTRIBUTION FOLLOWS : A POWER LAW !!

RICH GETTING RICHER PHENOMENON : ALSO CALLED AS : PREFERNTIAL ATTACHMENT !!

Implementing Rich-getting-richer Phenomenon
(Barabasi-Albert Model)-1

Rich-get-richer phenomenon (i.e. Barabasi-Albert Model)- The Main Idea

- ① We start with m_0 nodes, the links between which are chosen arbitrarily, as long as each node has at least one link.
- ② The network develops following two steps:
 - ① **Growth:** At each timestep we add a new node with m ($\leq m_0$) links that connect the new node to m nodes already in the network.
 - ② **Preferential attachment:** The probability that a link of the new node connects to node i depends on the degree of node i .

Steps for Implementation

- Take n , i.e. total number of nodes, from the user.
- Either take m , i.e. the number of edges to be connected to the new node, from the user, or decide it yourself based on n . In our implementation we will decide it ourselves as follows:
 - (m_0 is the initial number of nodes that should have atleast one link. m should be less than or equal to m_0).
 - We will take m_0 to be any random number between 2 to $n/5$. (You can use any measure.)
 - We will take m to be one less than m_0 .
- Add the rest $n-m_0$ nodes. Add edges to these $n-m_0$ nodes based on 'preferential attachment'.

Steps to be followed for Preferential Attachment

For all $n - m_0$ nodes, repeat the following:

- Add the node.
- To add the edges, do the following:
 - Preprocessing:
 - Get a dictionary of degrees (since preferential attachment will happen based on degrees).
 - Maintain a dictionary of probabilities. (The probabilities have to be assigned based on degrees. More the degree, more the probability. Precisely, $\text{probability}[i] = \text{Degree}[i]/\text{Sum}(\text{degrees of all the nodes})$).
 - Maintain a list of lists for maintaining cumulative node probabilities.(This is for choosing a node based on probabilities)
For Example,
Probabilities = [0.2, 0.3, 0.5]
—> *Cumulative Probabilities = [0.2, 0.5, 1.0]*.
 - While edges_added are not equal to m :
 - Choose a random number from 0 to 1.
 - Whichever node has cumulative probability more than this random number, the edge will be connected to that node, if not already added.

If $r = 0.4$

—> Node 2 will be chose out nodes 1, 2 and 3.

HOW TAKING CUMMILATIVE PROBABILITY HELPS US ?

IDEA : BASICALLY IDEA IS THAT : APKA : JITNI JYADA PROBAB HOGI : UTNI BADI

WINDOW HOGI CUMILATIVE ME AND THEREFORE : R KA USE < HO JANE KA

CHANCE JYADA SE JYADA HOGA !!

Implementing a Random Graph (Erdos-Renyi Model)-I

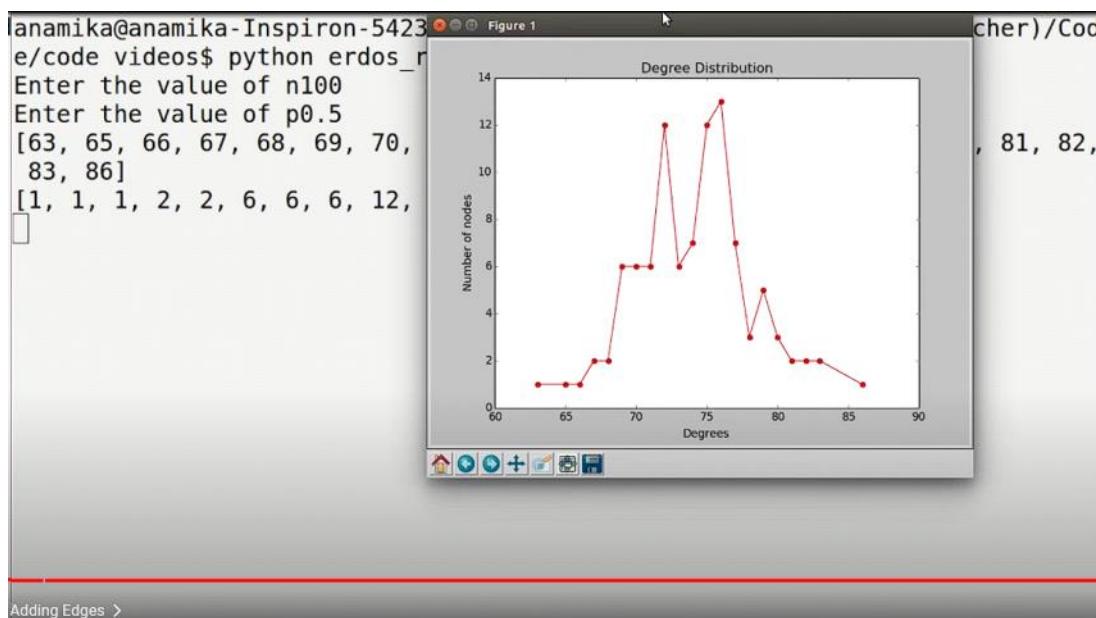
Anamika Chhabra

Steps for Implementation

- ① Take n , i.e. total number of nodes, from the user.
- ② Take p , i.e. the value of probability from the user.
- ③ Create an empty graph. Add n nodes to it.
- ④ Add edges to the graph *randomly*.

Steps to be followed for adding edges *Randomly*

- ① Take a pair of nodes.
- ② Get a random number r .
- ③ If r is less than p : Add this edge, else ignore.
- ④ Repeat step 1 for all possible pair of nodes.



Degree distribution for such graph !!

Idea : bcz we were randomly adding a edge and not preferentially attaching therefore a normal distribution was seen !!!

Erdos_renyi_graph we saw earlier also : (n,p) was the properties that we passed into the fnc !! {VVIMP !!}

Forced Versus Random Failure of nodes (*Attack Survability*)

Anamika Chhabra

```
1 import networkx as nx
2
3 def remove_a_random_node(G):
4     nodes = G.nodes()
5     r = random.choice(nodes)
6     G.remove_node(r)
7     return G
8
9
10 def remove_a_high_deg_node(G):
11     degrees = G.degree()
12     nodes = sorted(degrees.items(), key = lambda x:x[1], reverse = True)
13     #where the tuple is having two things: (node, degree)
14     G.remove_node(nodes[0][0])
15     return G
16
17 print 'nodes removed when randomly chosen', nodes_removed_random
```

```
raw_input()

print '###Selective removal###'
G2 = G.copy()
print 'Is G2 connected?', nx.is_connected(G2)
nodes_removed_selective = 0

while(nx.is_connected(G2)):
    G2 = remove_a_high_deg_node(G2)
    nodes_removed_selective += 1
    print 'Is G2 connected?', nx.is_connected(G2)
print 'nodes removed when selectively chosen', nodes_removed_selective

main()
```

Imp Result : { FOR REAL WORLD NETWORKS }

WHENEVER WE HAVE A GRAPH AND WE REMOVE THE NODES RANDOMLY THEN IT TAKES MORE NO OF ITERATIONS TO MAKE THE GRAPH CONNECTED COMPARED TO A : PREFERENTIAL DELETION OF HIGHEST DEGREE NODE !!

NOTE : THAT IS WHY : IF A ATTACKER NEEDS TO BREAK A NETWORK / MAKE A NETWORK DISCONNECTED THEN : IT SHOULD DO PREFERNETIAL ATTACK AND NOT RANDOM SELECTION !!

What are hubs in a network?

Hubs are those nodes in a graph : jinke removal se : graph become disconnected !!

IMP RESULT { FOR RANDOM NETWORK }:

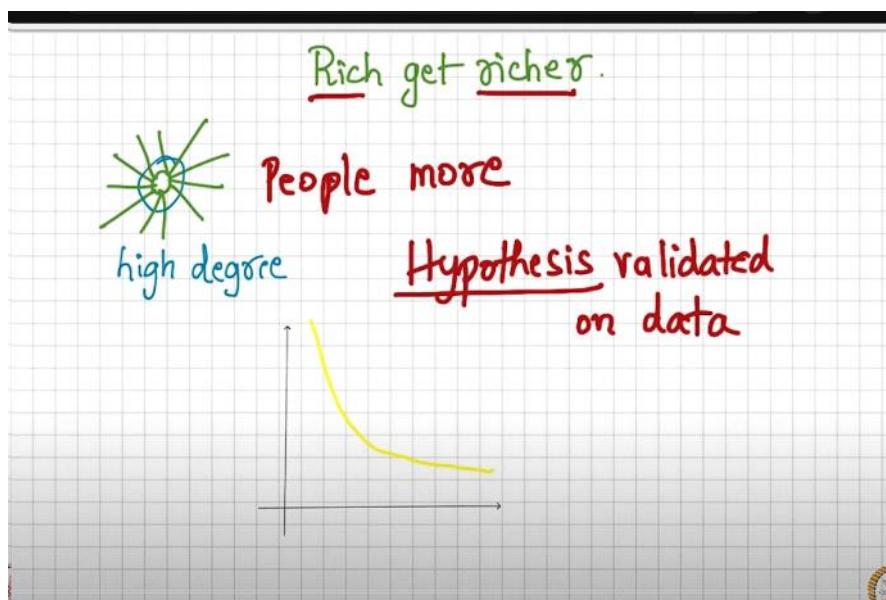
SO FOR IT , IT DOESN'T MAKE MUCH DIFFERENCE TO : REMOVE NODES RANDOMLY OR : USING PREFERENTIAL DETACHMENT : REASON BEING : RANDOM NETWORKS GENERALLY DON'T EXHIBIT PRESENCE OF HUBS WHILE REAL WORLD NETWORKS DO EXHIBIT THE PRESENCE OF HIGH DEGREE NODES WHOSE REMOVAL LEAD TO MAKING GRAPH DISCONNECTED !!

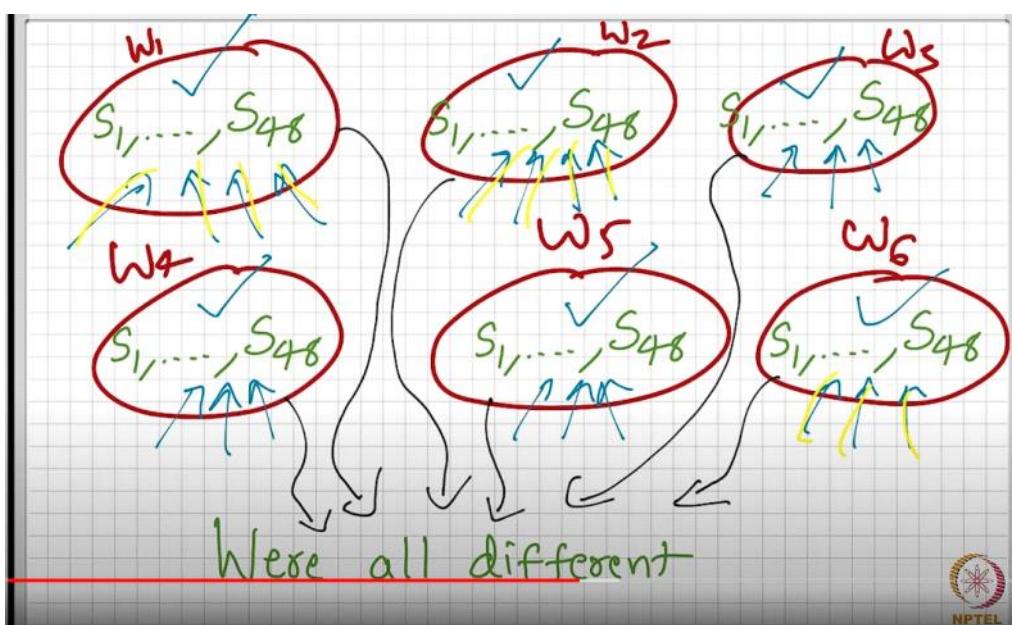
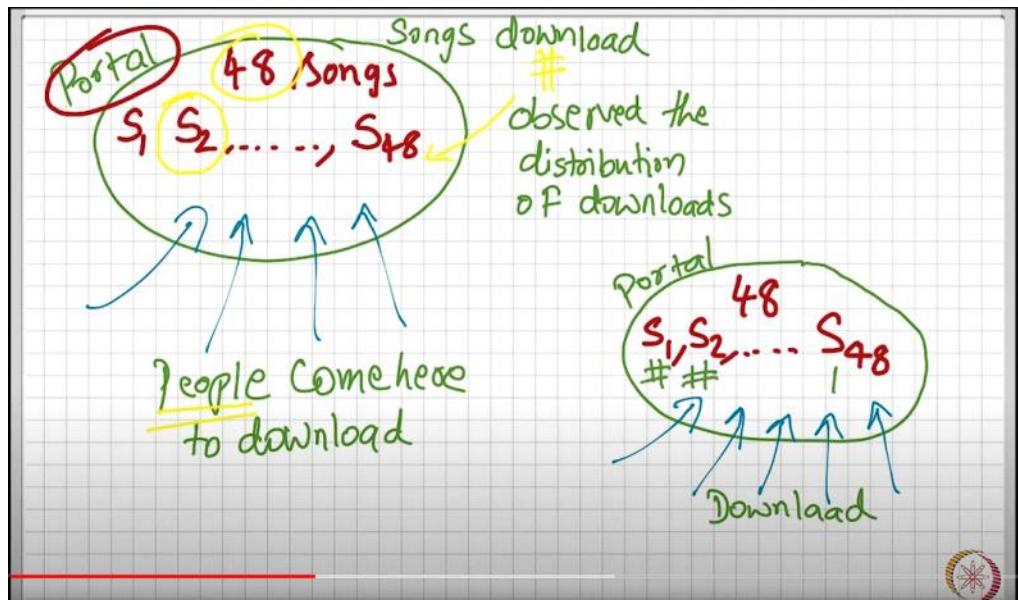
- 6) Consider a random graph with 600 nodes and edges between any two nodes with a $1 p$ probability of 0.2, where can one expect the peak of the degree-distribution graph?

- 300
- 120
- 320
- 500

Q) Consider the given graph. There is an incoming node D. To which node will this node point?
IMP : PEAK OF DISTRIBUTION IN RANDOM GRAPH CALCULATED USING MULTIPLICATION OF N AND P !!

WEEK 10 :



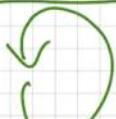


Rich get richer

Download information.

Highly downloaded songs biased new people!

"Rich" get richer



Rewind history?

1st

ranked
people aren't
same

EXPERIMENT

Basically question was how does someone became rich in 1st hand and the reason for that was : luck !!

And uske baad : popularity of them : biased incoming people and then : rich gets richer !!

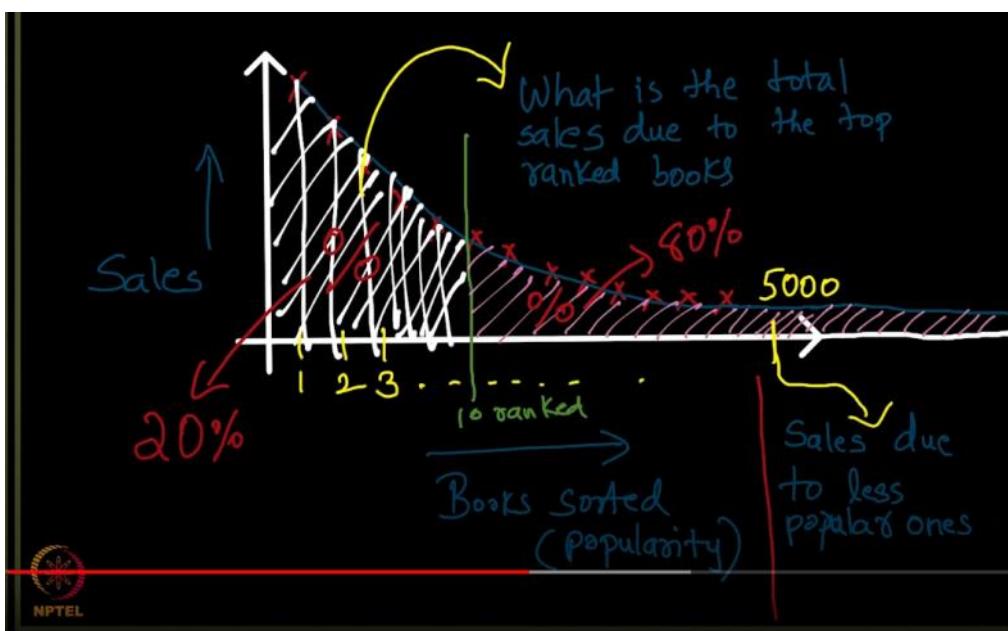
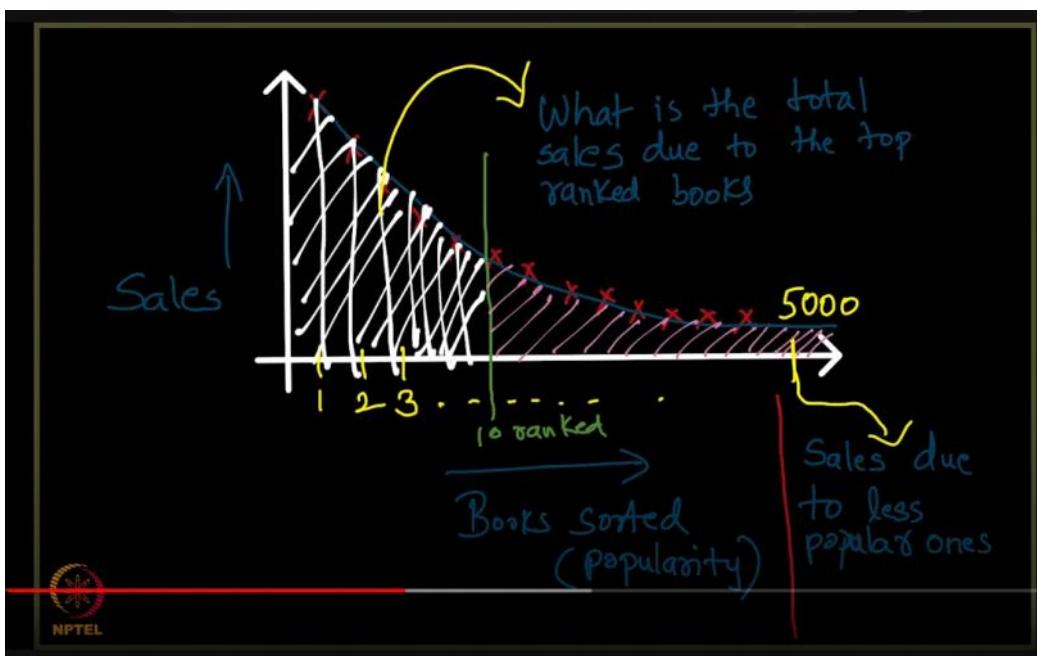
1) How can one detect the right talent by subtracting out the popularity obtained by RGR phenomenon.

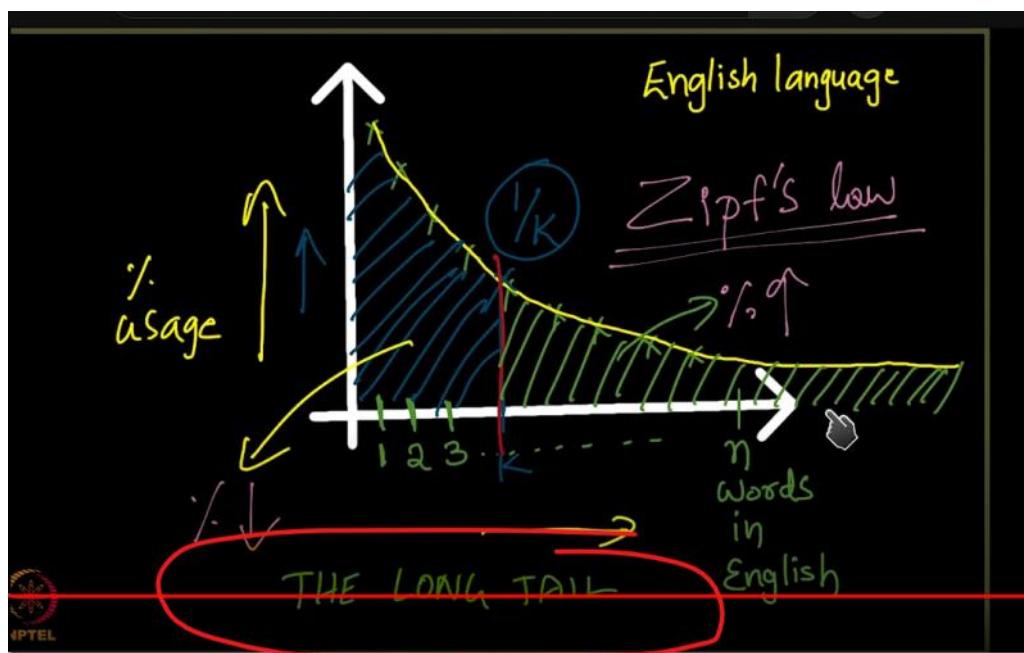
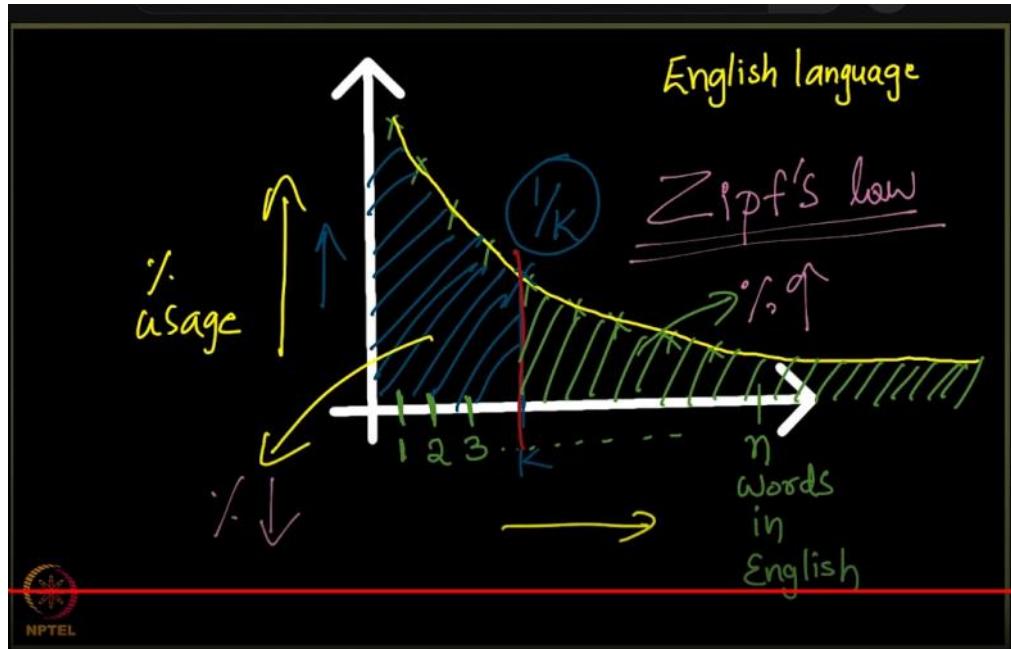
2) RGR
How does one judge the true value of an entity

1) RGR the reason for emergence power law.

2) RGR adds noise to popularity !
How to remove.

More Powerlaw

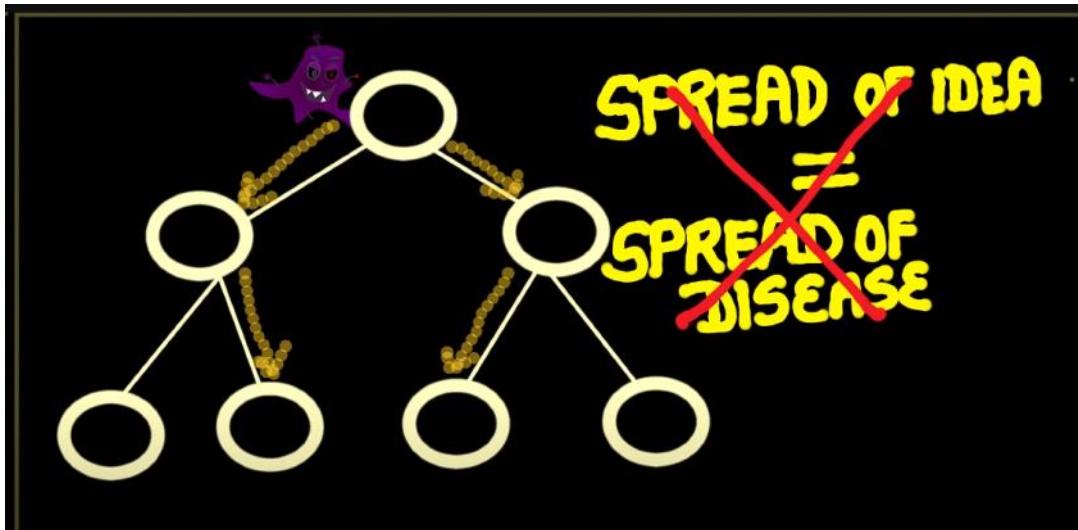
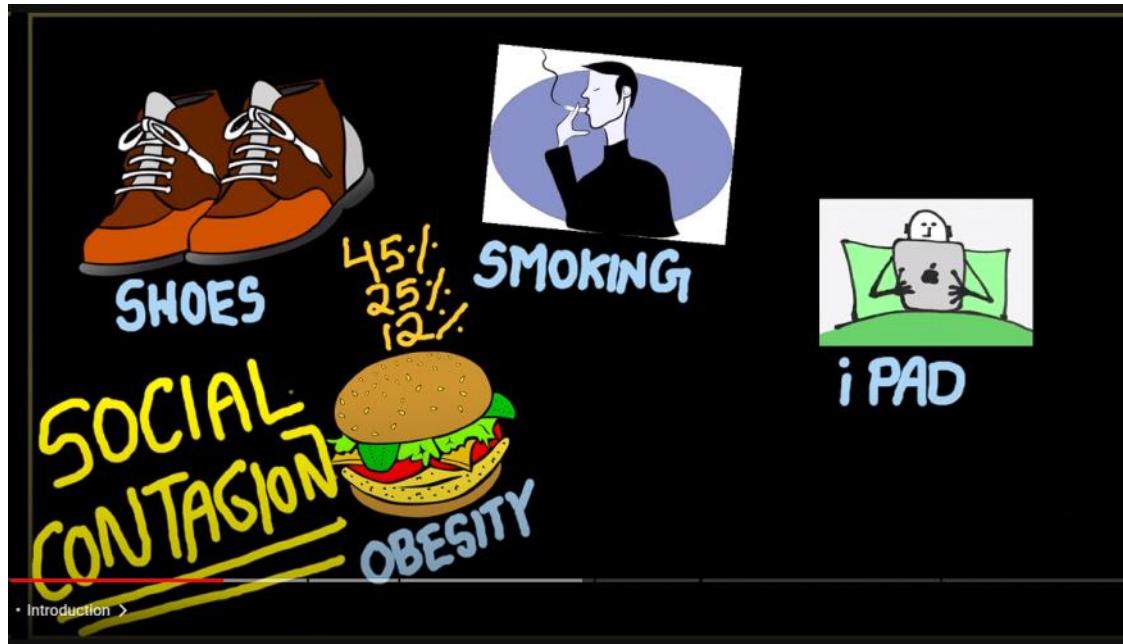


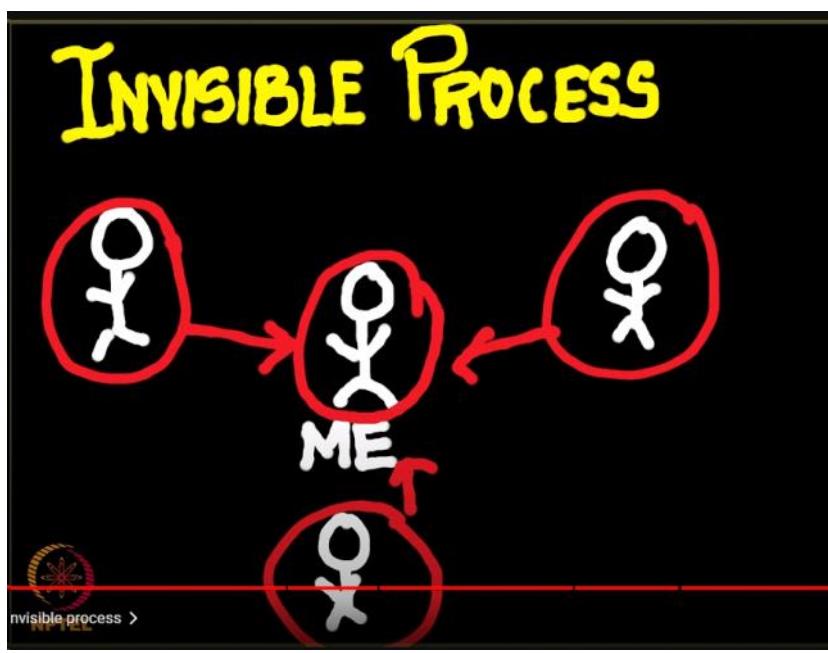
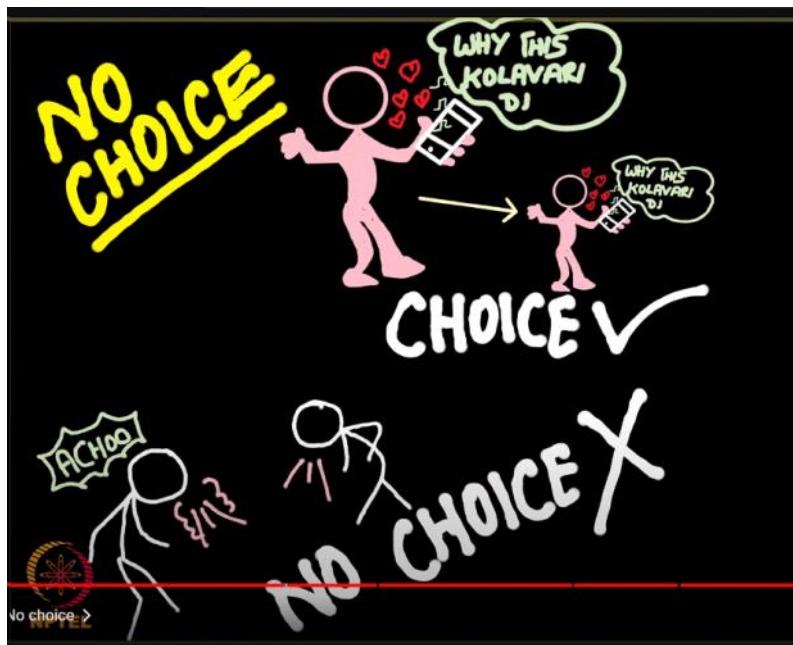


THE LONG TAIL PHENOMENON !!

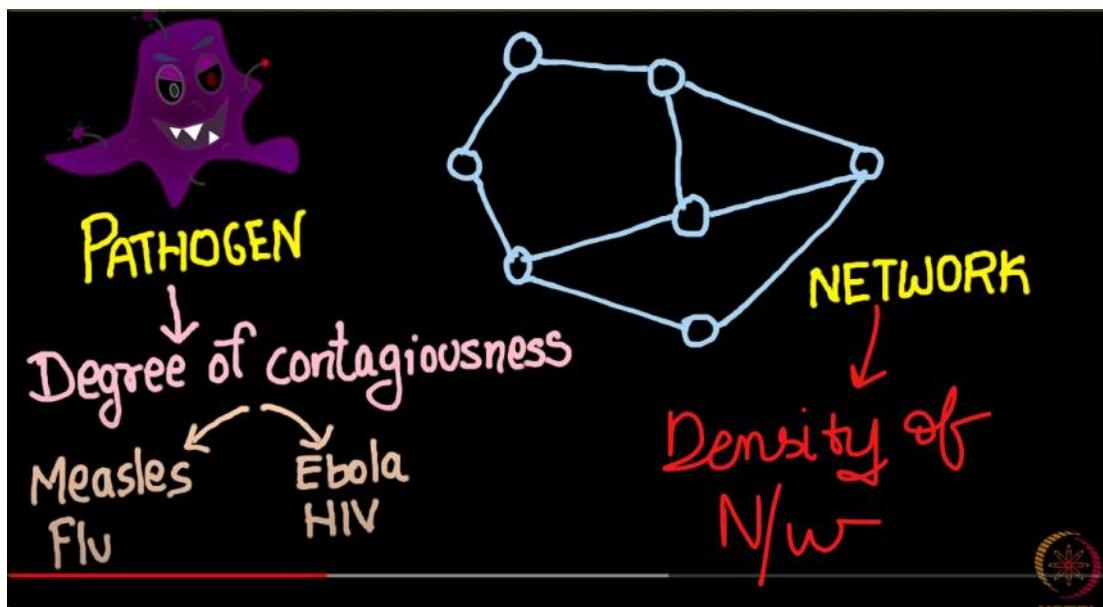
OBESITY : 3 LEVELS TAK CONTAGIOUS HOTI HAI :



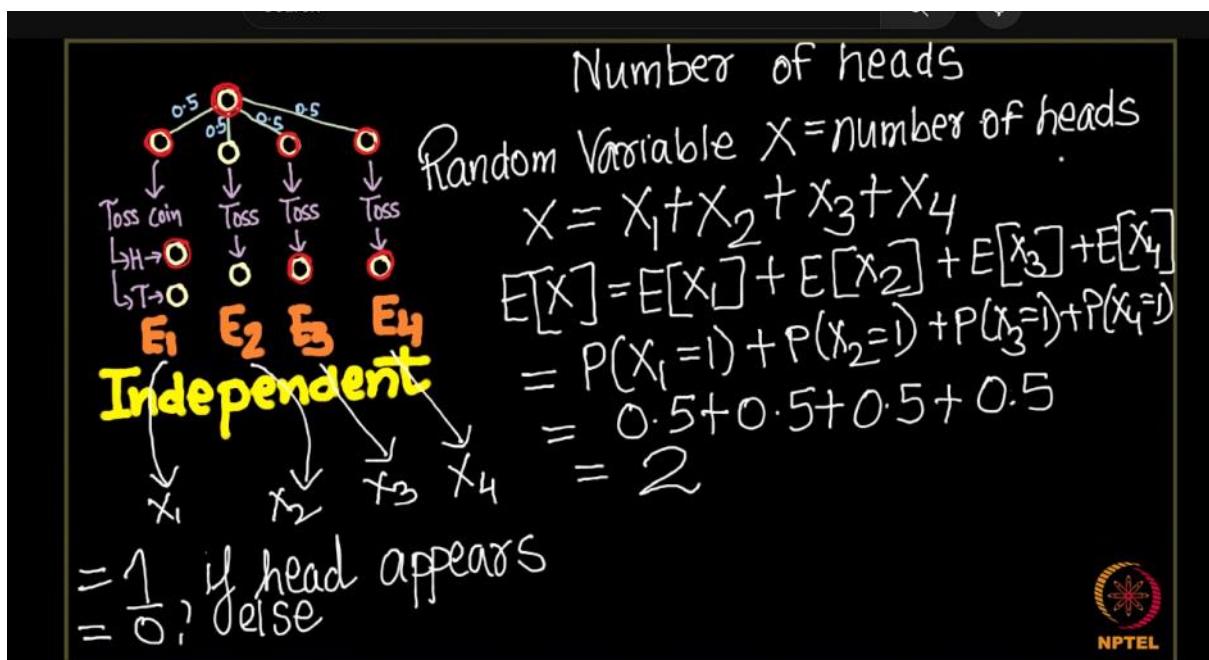
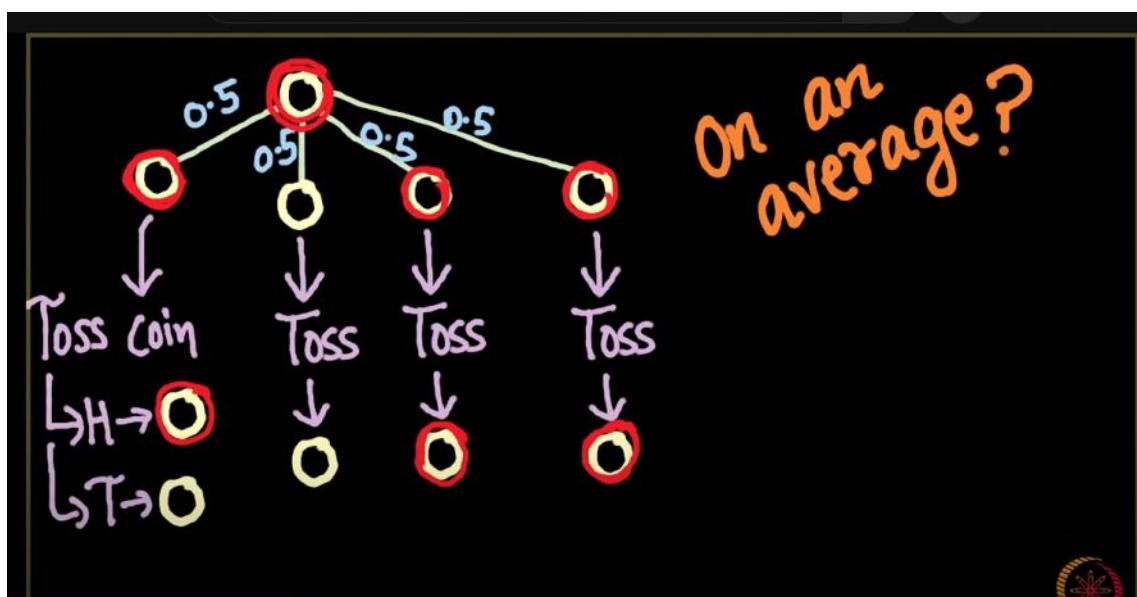
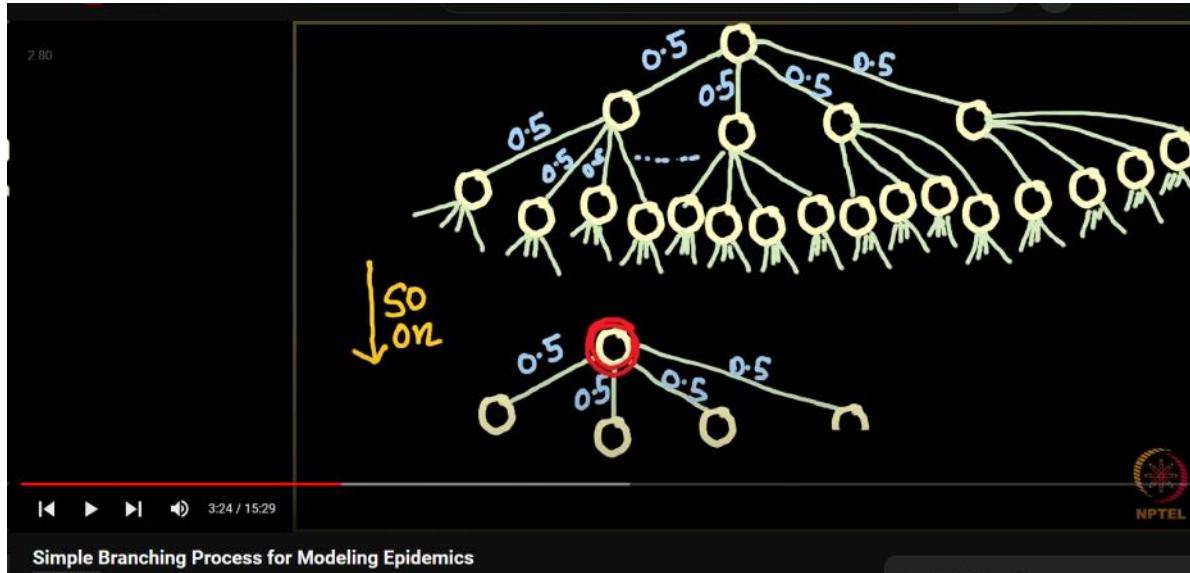


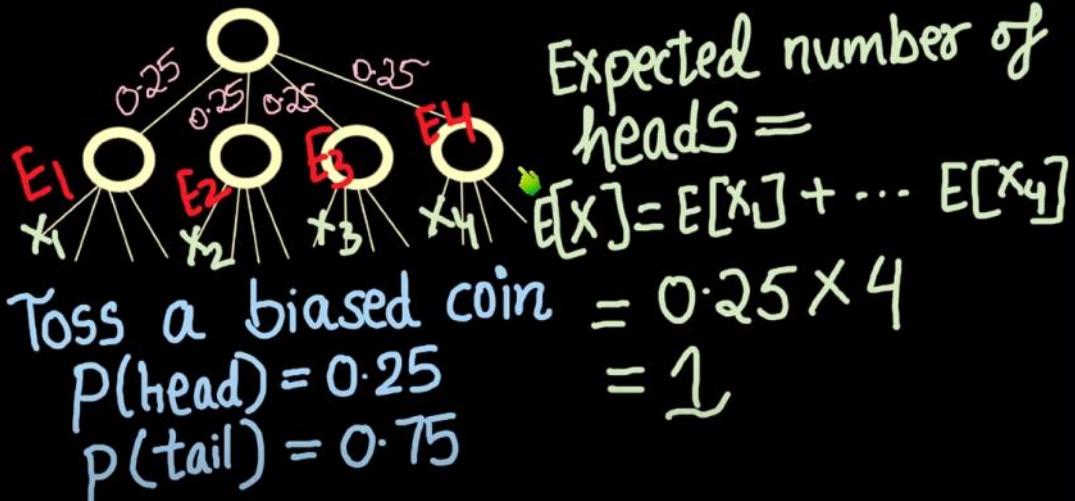


FROM WHOM I ADOPTED THE DISEASE IS HARD TO TELL !!



These 2 things are required to model the epidemics !!





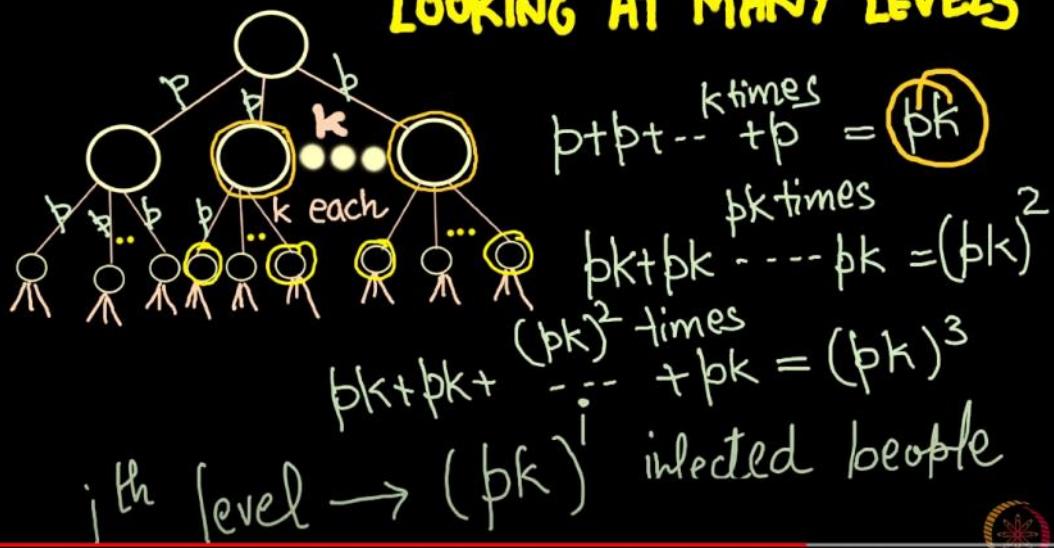
Generalise



LOOKING AT MANY LEVELS

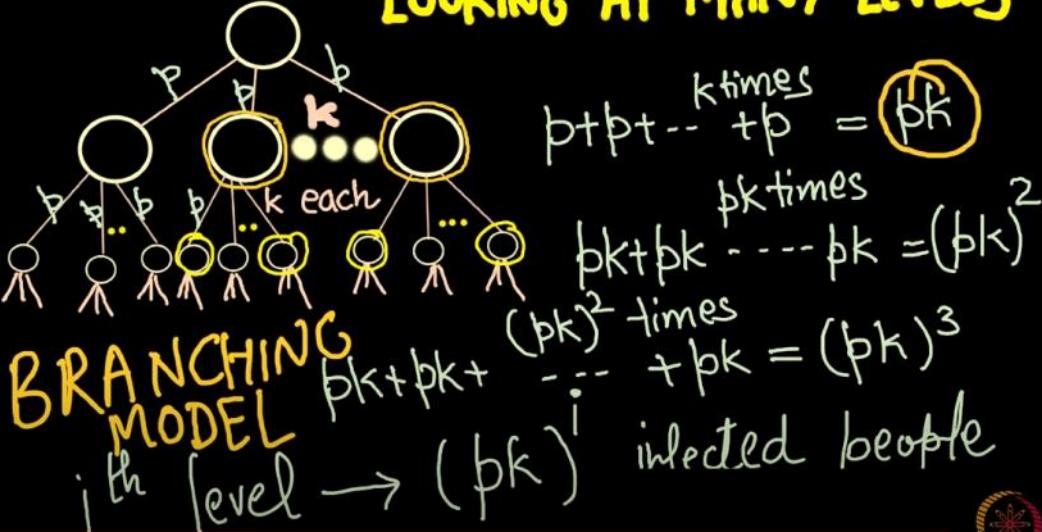


LOOKING AT MANY LEVELS



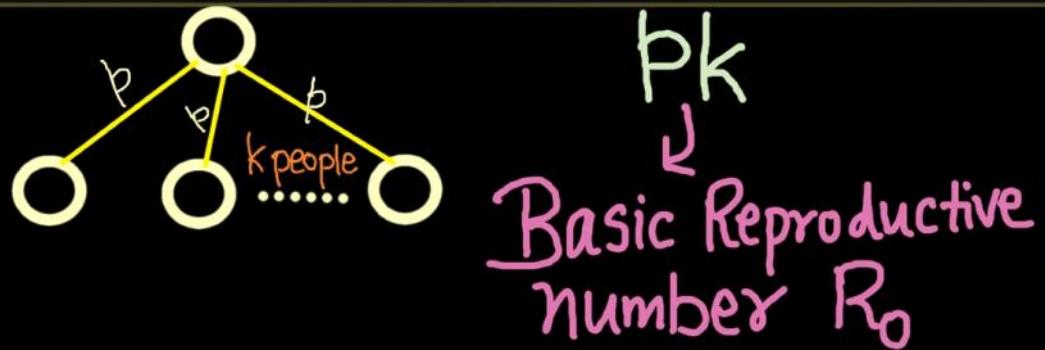
EXPECTED NO OF PEOPLE AT ith level !!

LOOKING AT MANY LEVELS

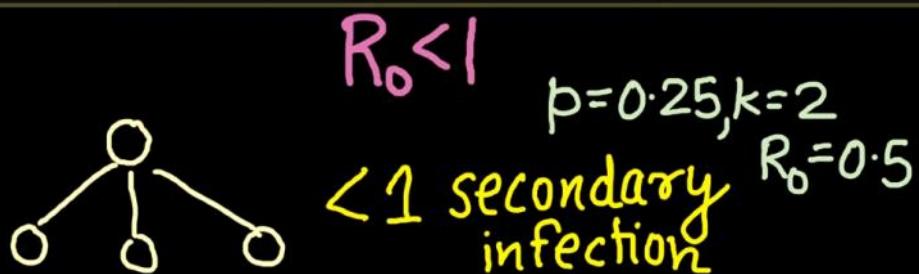


CALLED BRANCHING MODEL !!

IF a person at too much down level is also infected by this then we call such a disease as epidemic !!

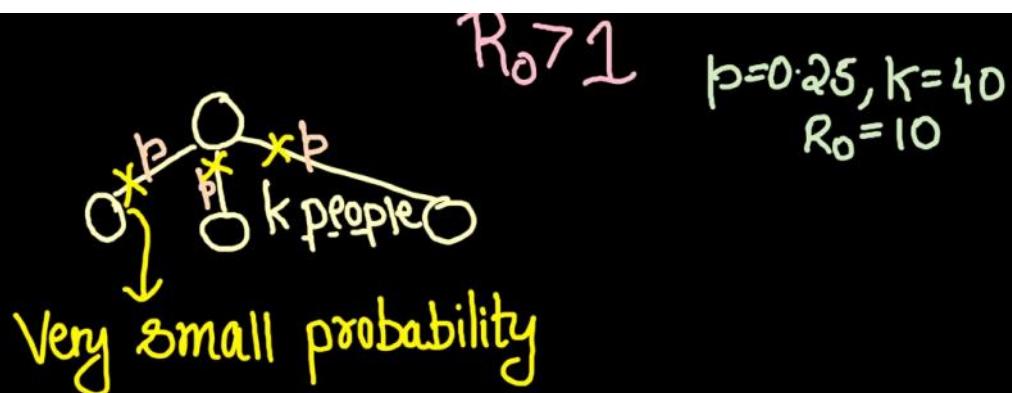


Tells you the expected no of secondary infections a guy would do !!



Infection ultimately dies away

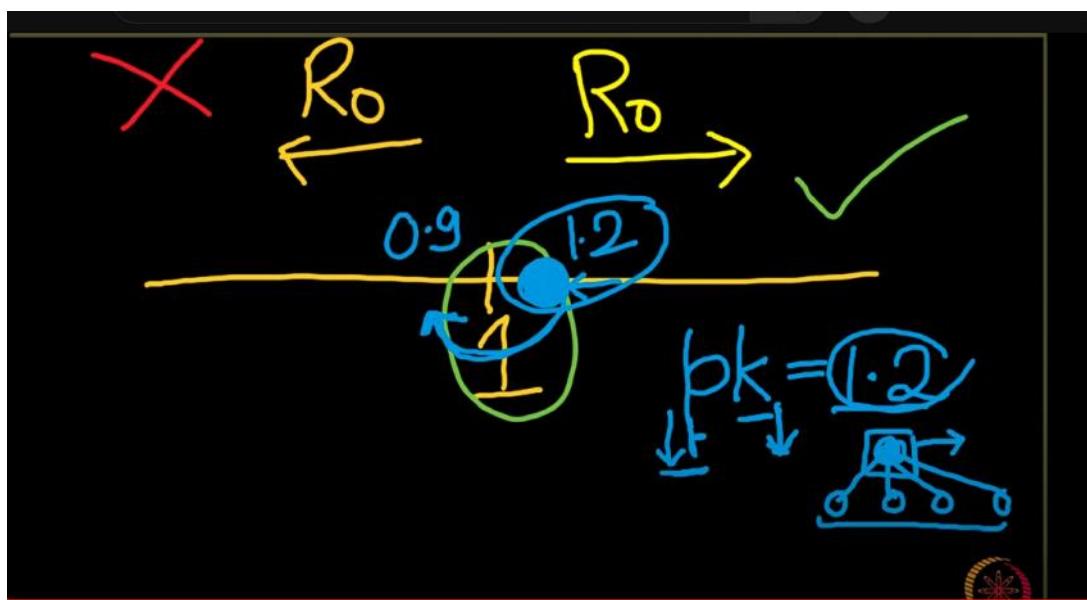
Therefore using this Reproductive number we are able to guess if a disease will become epidemic or not !!



Very small probability it will die away !!

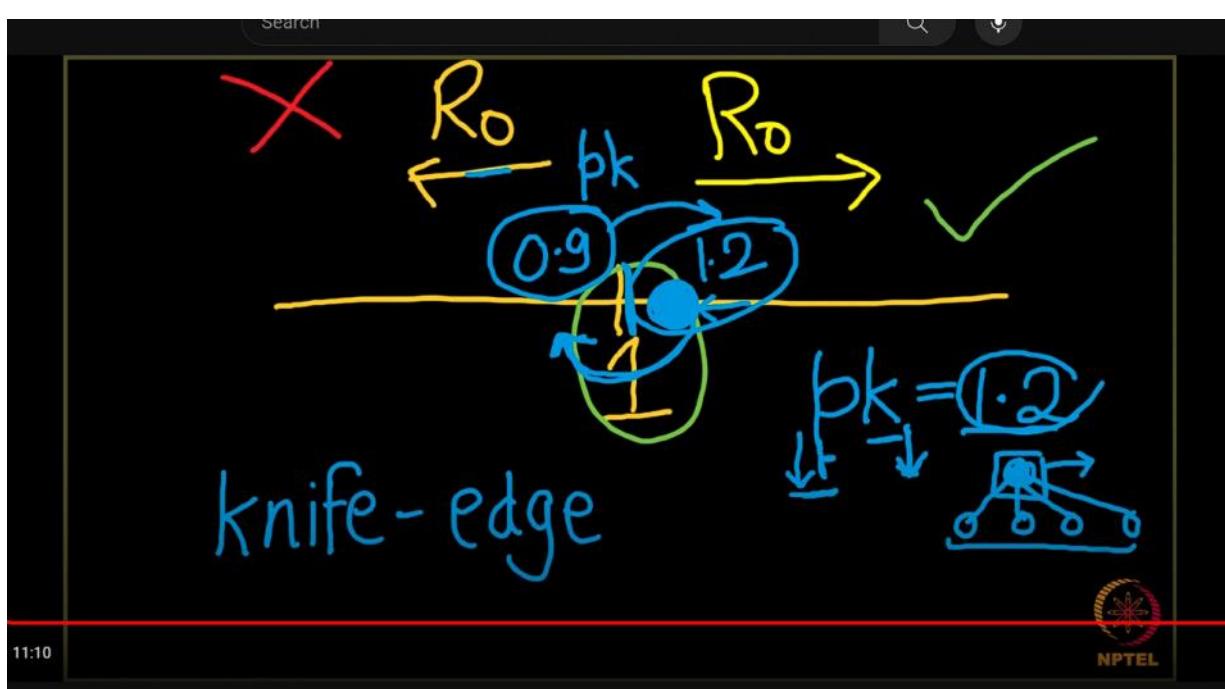
$R_0 < 1$: Dies away with
 $p\gamma = 1$

$R_0 > 1$: Persists in n/w
with +ve probability



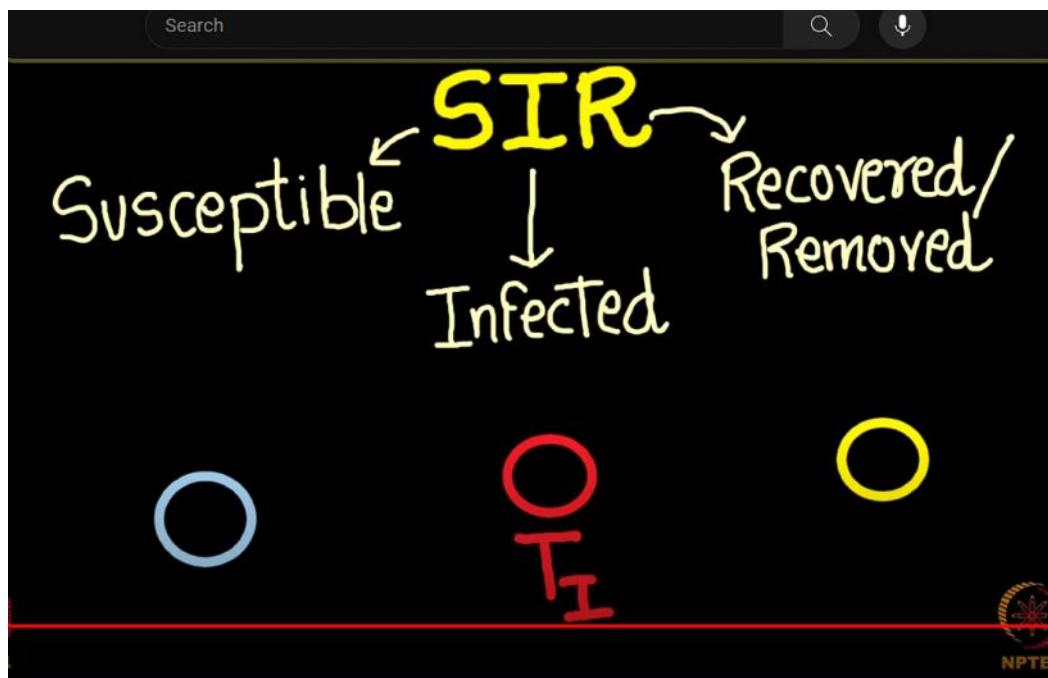
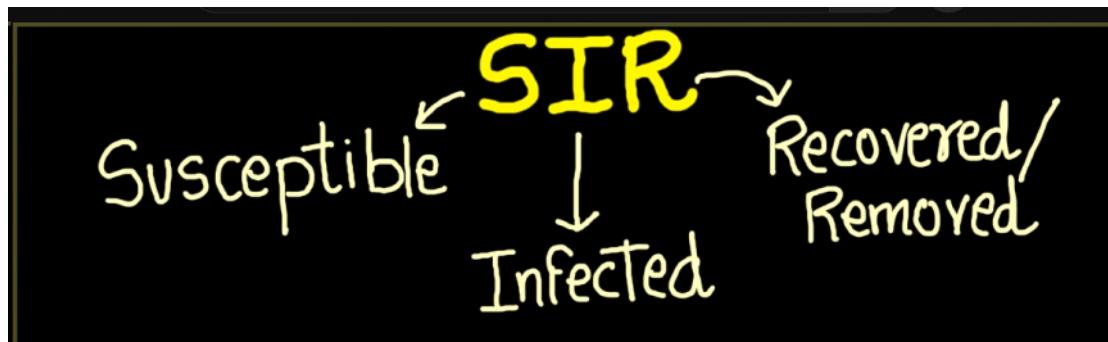
Decreasing k : means quarantine and all !!

Decreasing p : means logo ko haath dhone ko kaho and all !!



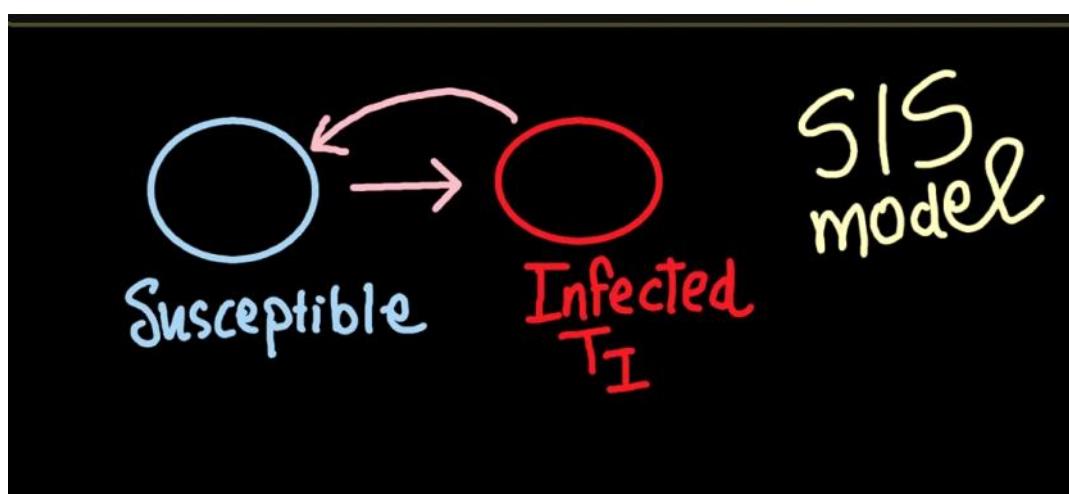
This property of R₀ is called KNIFE EDGE PROPERTY !!

Modeling epidemics on complex networks

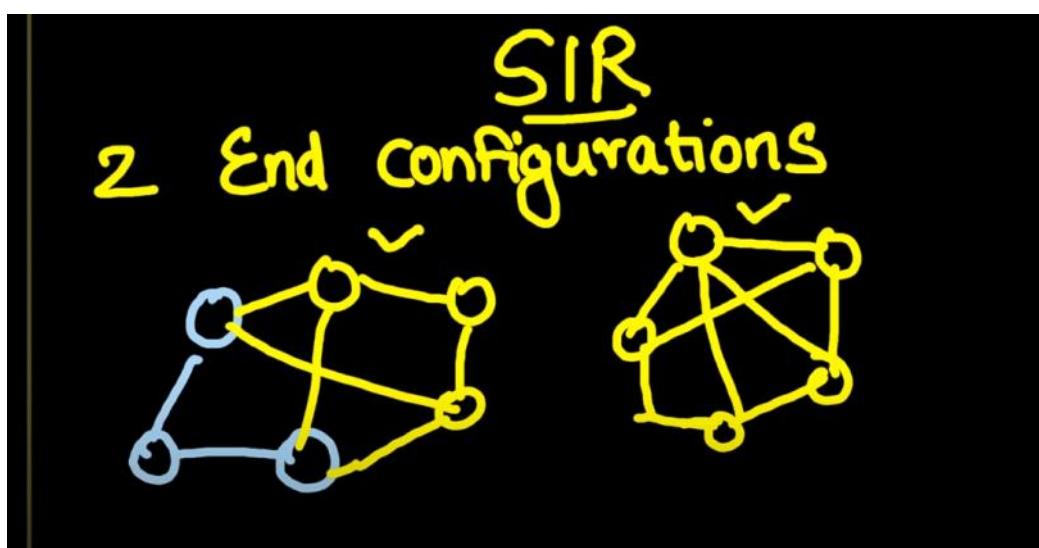
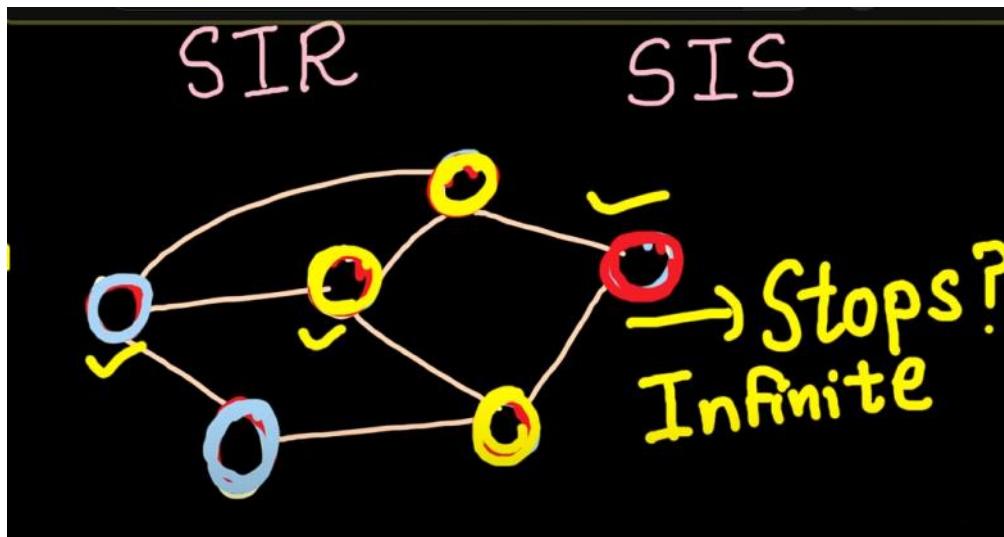


T_I = time for which you will be infected !!

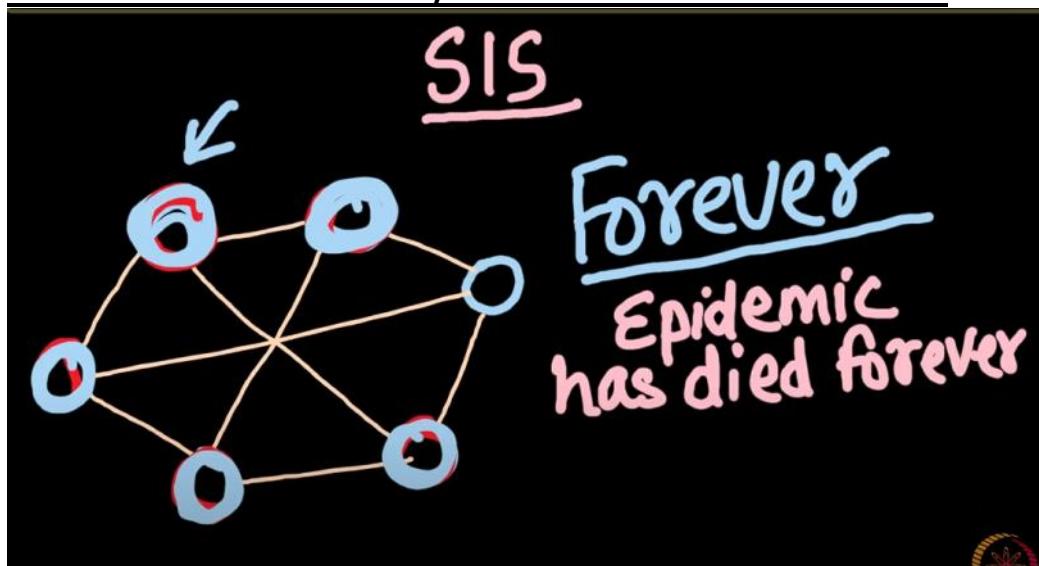
This model is used in case of diseases like : MEASLES ! JOKI EK BAAR AAP RECOVER KAR GAYE TO FIR LIFE TIME TAK USE IMUNE HO JATE HO !!



This model is used in case of diseases like : COMMON COLD !



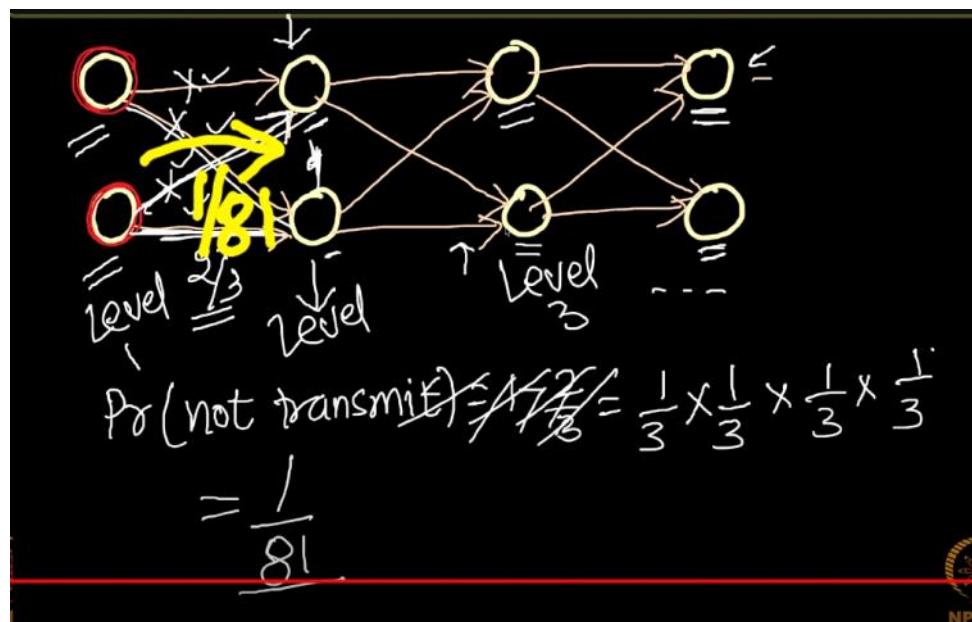
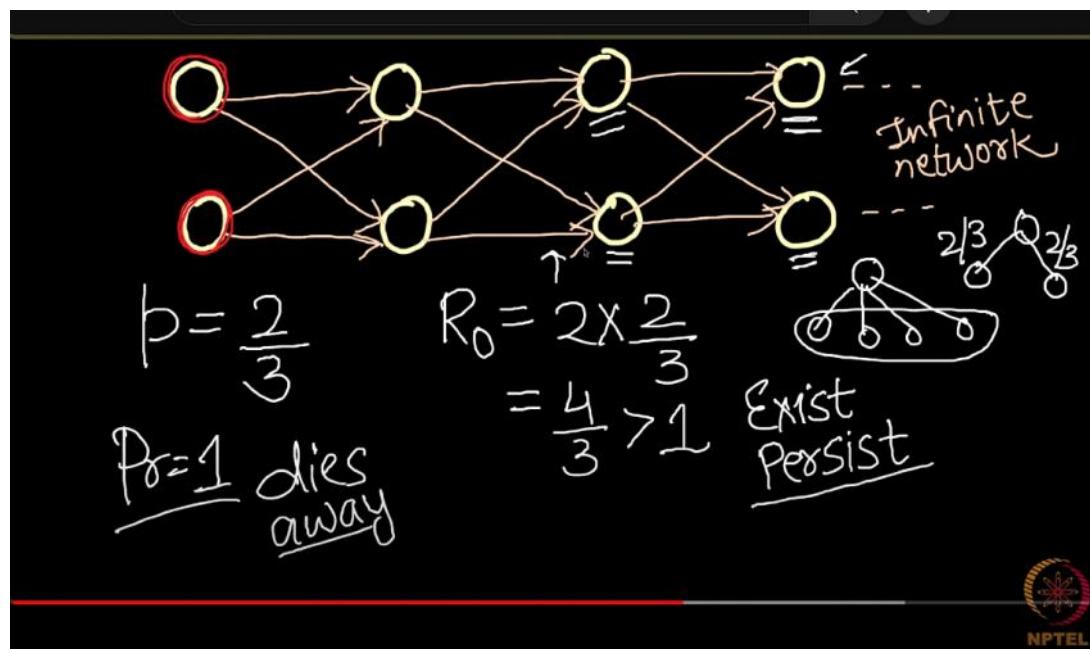
YELLOW MEANS RECOVERED / REMOVED FROM THE NETWORK !

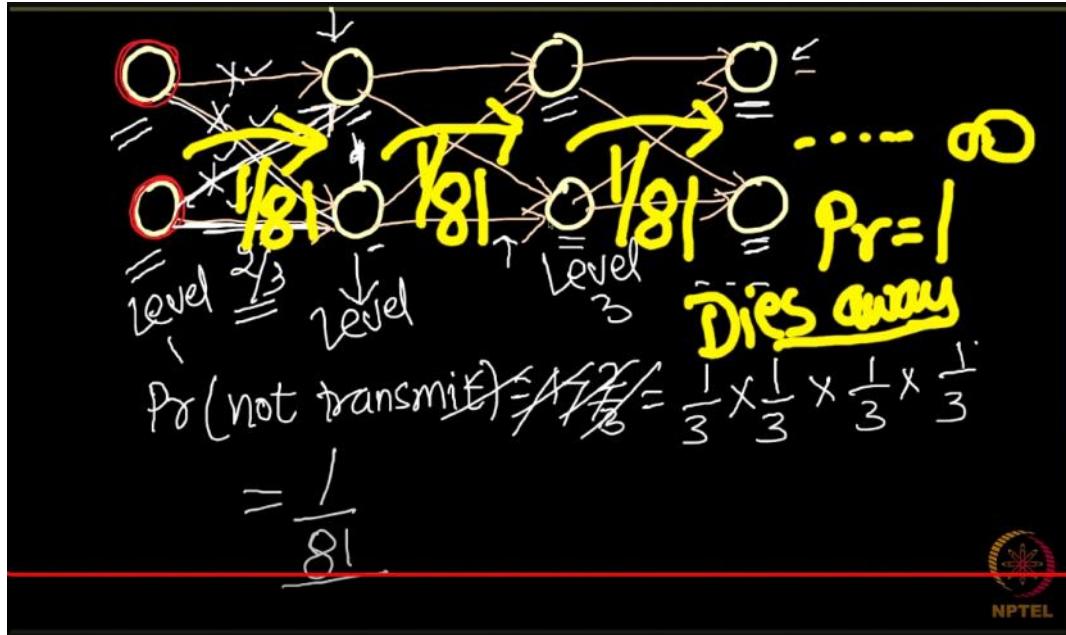


SIR \rightarrow End

SIS \rightarrow Keep Running forever

Basic Reproductive Number Revisited for Complex Networks

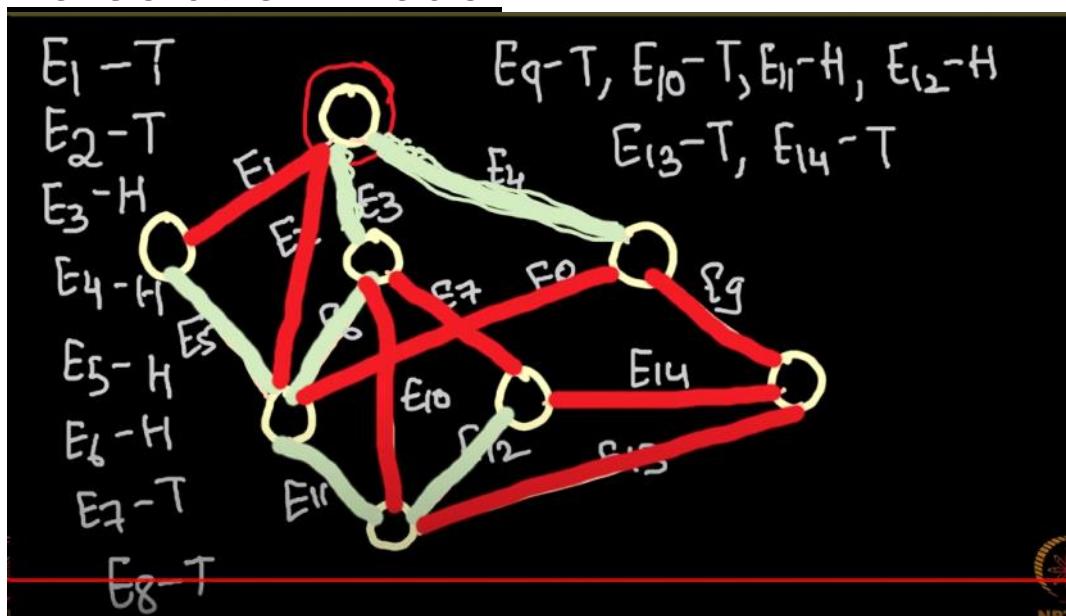




RESULT :

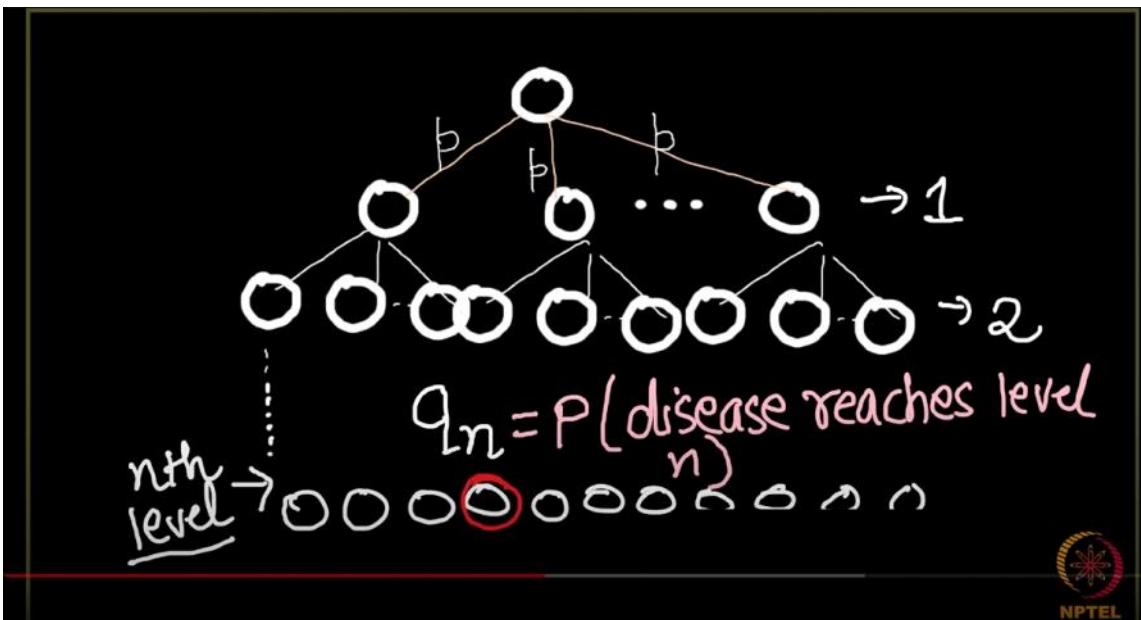
IN SIR AND SIS MODEI : BASIC REPRODUCTIVE NUMBER WALA CONCEPT DOESN'T HOLD TRUE !!

Percolation model



IF THERE IS A OPEN PIPE PATH FROM SRC TO DESTINATION : THEN : WE SAY THAT YES THAT NODE WILL BE INFECTED : SO IDEAS IS THAT : WE SAW IT FROM A DIFF PERSPECTIVE : REMOVING THE NOTION OF TIME !! : WHICH WE DID IN SIR AND SIS MODEL !!

PROOF FOR BASIC REPRODUCTIVE NUMBER :



$$q_n = P(\text{disease reaches level } n) \\ = P(\text{disease persists level } n)$$

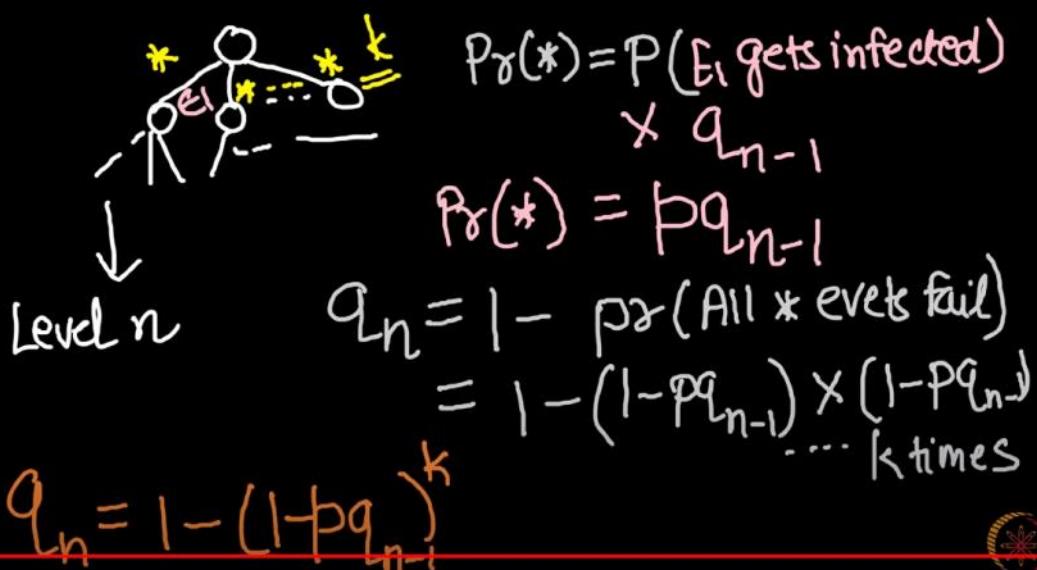
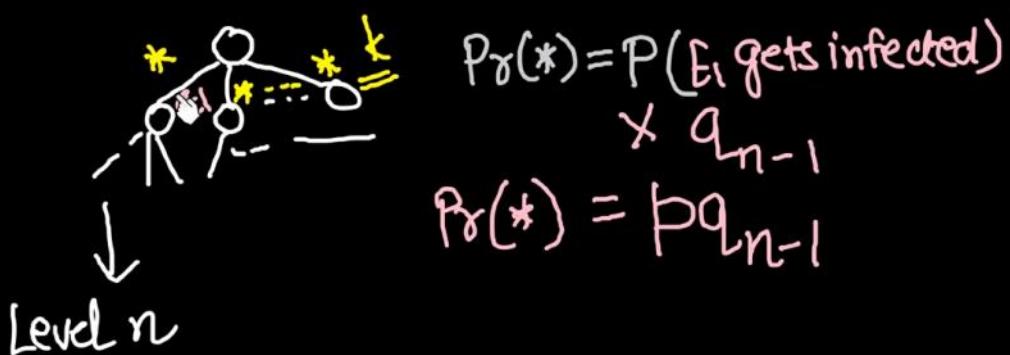
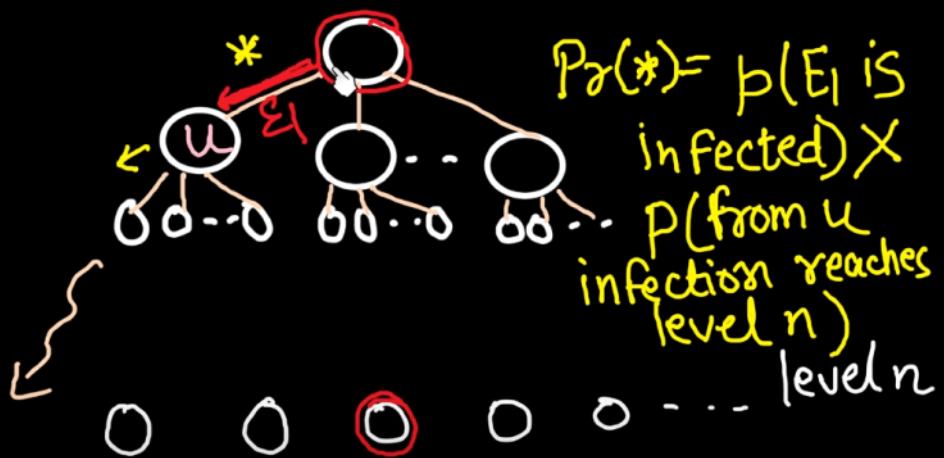
(*) $q_n = 1$ (Epidemic)
 $n \rightarrow \infty \rightarrow 0$ (Died away)

To Prove

| | | |
|-----------|-----------|--------------|
| $R_0 < 1$ | $q^* = 0$ | for sure |
| $R_0 > 1$ | $q^* > 0$ | +ve persists |

(*) $q_n = 1$ (Epidemic)
 $n \rightarrow \infty \rightarrow 0$ (Died away)





If $R_0 < 1$ $q^* = 0$

$R_0 > 1$ $q^* > 0$

$$q_n = 1 - (1 - p q_{n-1})^k$$

$$q^* = \lim_{n \rightarrow \infty} q_n$$

$$q_0 = 0$$

$$q_1 = 1 - (1 - p q_0)^k$$

$$q_2 = 1 - (1 - p q_1)^k$$

$$q_n = 1 - (1 - p q_{n-1})^k$$

$$y = f(x) = 1 - (1 - px)^k$$

$$q_1 = f(q_0)$$

$$q_2 = f(q_1) = f(f(q_0))$$

$$q_3 = f''(q_0) \text{ } \underset{\infty \text{ times}}{\dots}$$

$$q^* = f^{\infty}(f^{\infty}(f^{\infty}(q_0)))$$

$$q^* = f(f(f(\dots)))$$



$$f(x) = 1 - (1 - px)^k$$

$$f'(x) = -k(1 - px)^{k-1}(-p)$$

$$= pk(1 - px)^{k-1}$$

$$x=0$$

$$f'(0) = pk(1 - 0)^{k-1}$$

$$= pk$$

↑ Pull up for precise seeking

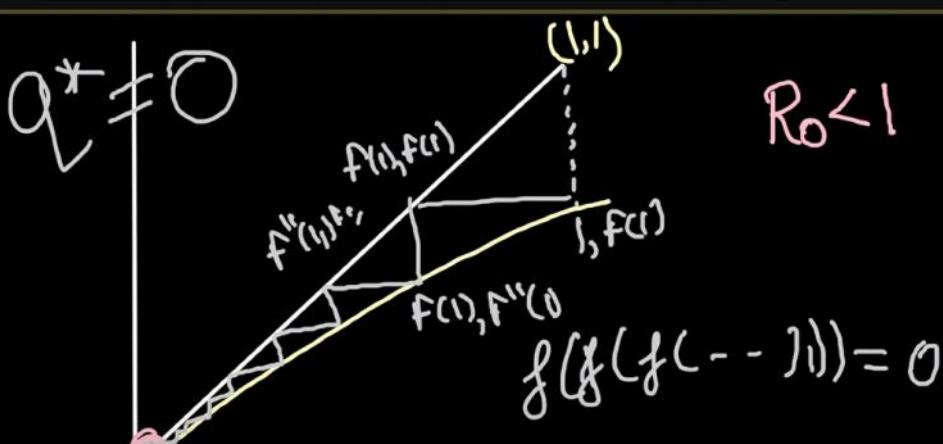
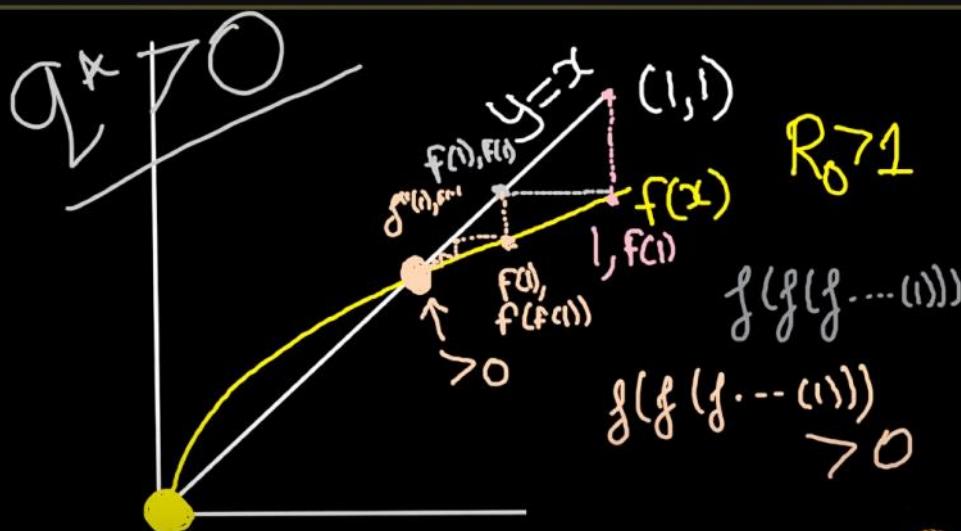
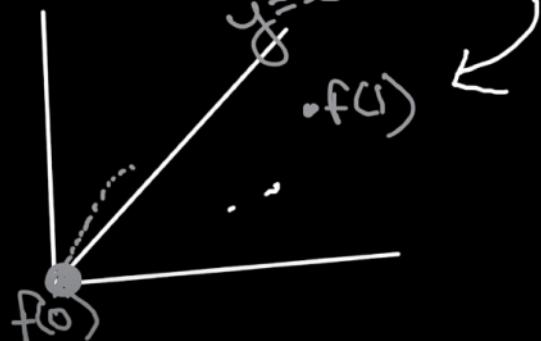
$$\begin{aligned} f(0) &= 1 - (1 - p \cdot 0)^k \\ f'(0) &= -k(1 - p \cdot 0)^{k-1}(-p) \\ &= pk(1 - 0)^{k-1} \\ f'(0) &= pk(1 - 0)^{k-1} \end{aligned}$$

Slope
5:44



$$f(x) = 1 - (1 - px)^k$$

Slope (origin)
= R_0



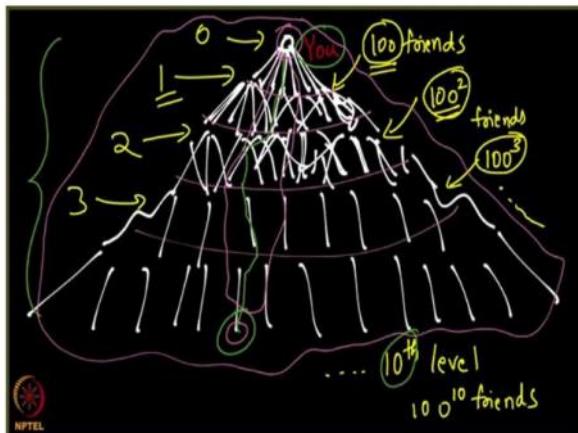
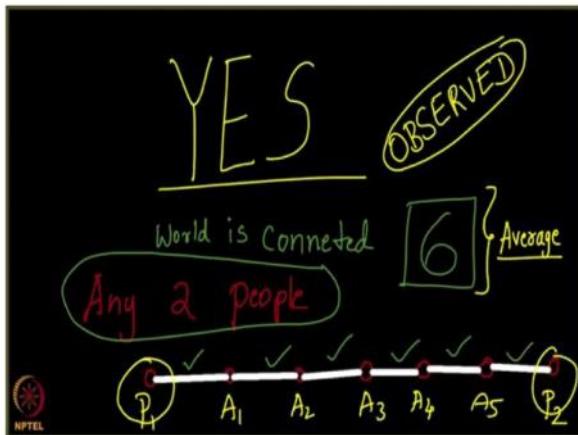
WEEK 11 :

Major points :

1. Milgram Experiment : deviced the local searching algo !! For finding ki kisko pass kare letter !! : quoated that : on an avg 6 hops me person 1 to person 2 tak pahuncha jaa sakta hai

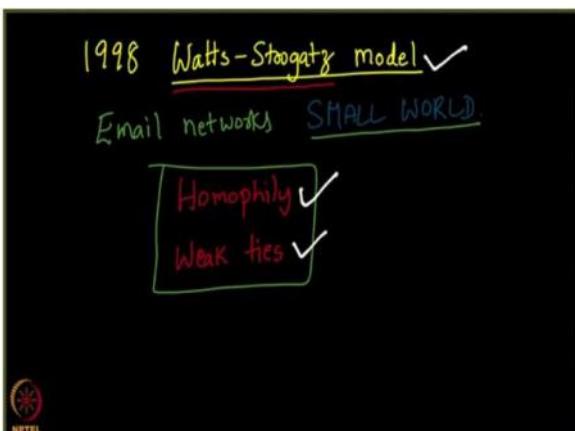
2. Generative Model , swartz and .. : homophily and weak tie reason for : small world

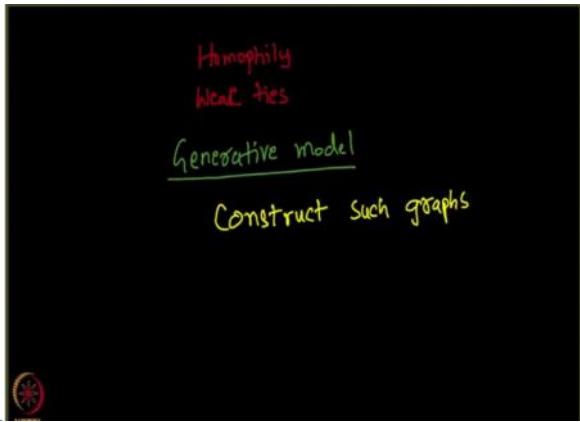
Generative model : vicinity me dalo edges : then rewire with a probability : $1/d^{**}k$: BEST VALUE OF k == 2 !!



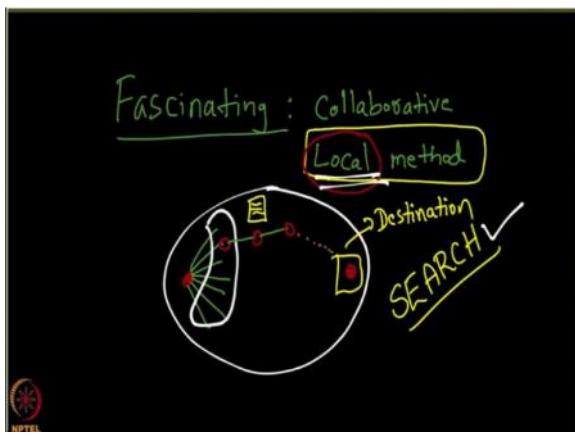
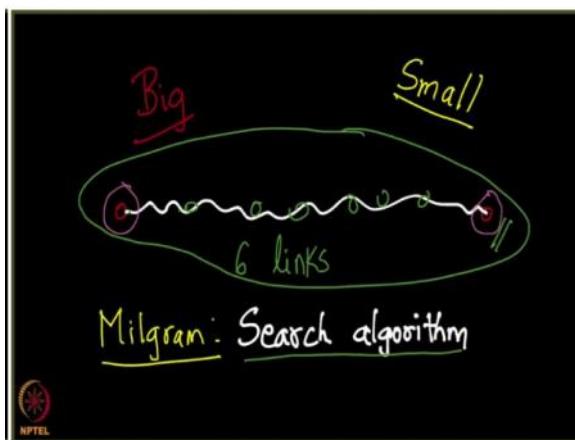
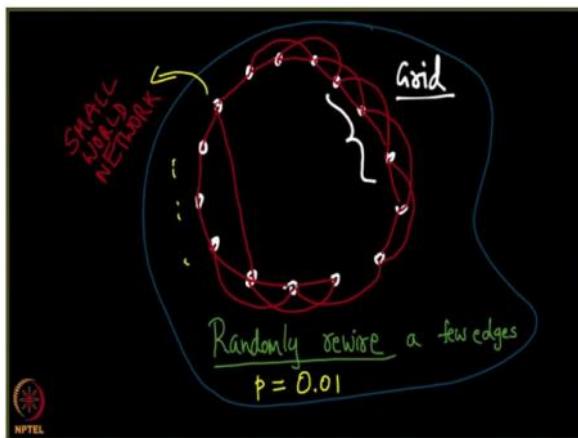
assume this is you and you have some say 100 friends. So, you have some

Intution for that : 6 hops jaisa less number hi : $100^{**}6$ logo ko : cater kar leta hai at 6th level which is huge !!





GENERATIVE MODEL IS A MODEL THAT HELPS YOU GENERATE THOSE GRAPHS THAT EXHIBIT THIS SMALL WORLD EFFECT !!



This is what we called as DECENTRALIZED SEARCH !!

- ! 2) In a 2-DWatts-Strogatz model, what is the ideal value of clustering exponent for an **1 point** efficient decentralized search?

- 1
- 2
- 1
- 2

VVIMP !!

- 4) In the generalized Watts-Strogatz model, let the distance between two nodes u and v be represented as $d(u, v)$, and the clustering exponent be represented by q , the probability that u and v are connected by a weak tie is proportional to

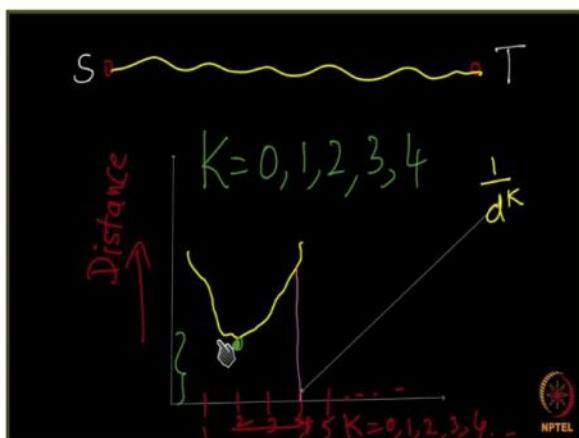
- $d(u, v)$
- $d(u, v)^q$
- $d(u, v)^{-q}$
- $d(u, v)^{-1}$

- 6) Which of the following correctly represents a Watts-Strogatz model on n nodes in **2 dimensional space**? **1 point**

-
- n nodes arranged in 2-D lattice where the connections between the nodes are all random.
-
- n nodes arranged in a 2-D lattice where every node is connected to every other node.
-
- n nodes arranged in a 2-D lattice where every node is connected to the nodes on its left, right, top, bottom and diagonally opposite.
-
- n nodes arranged in a 2-D lattice where every node is connected to the nodes on its left, right, top, bottom and diagonally opposite, and, some edges are randomly laid in the network between any two nodes.

- ! 9) Random rewiring in small world generative model refers to **1 point**

- Addition of an extra edge in the network
-
- Deletion of a random edge in the network and addition of a new edge in the network
- None of the above



Rewiring karne pe probability of existance of weak tie btw any 2 nodes u and v are = $1/d^{}k$ where best value of k == 2 !!**

the next node which is closest to be on the grid ok; of all the neighbours you choose that node that is closest to T on the grid. That is what we have been discussing all this while right. And this particular algorithm will tell you, this particular programming exercise will tell you that a K equals 2 is the best possible value for K.

K equals 2 for grid and K equals 1 for 1 dimension.

But, the idea here is simple idea here is that you use K to sort of decide on whether you will make friends with nodes that are further away right. And, when you make friends with nodes that further away a lot more then you are in trouble; a lot less you are in trouble. The perfect balance is when you choose K equals 2 ok. In fact, we will be showing you programming exercise; we will be explaining this in detail. It will become probably clear to you, then. As of now just know that this is curve is how it looks like and for 2 the distance is the shortest.

8) Choose the correct statement 1 point

- Watts-Strogatz model resembles a ring in 1 dimension and a grid in 2 dimension.
- Watts-Strogatz model resembles a grid in 1 dimension and a ring in 2 dimension.
- Watts-Strogatz model resembles a ring both in 1 dimension as well as 2 dimension.
- Watts-Strogatz model resembles a grid both in 1 dimension as well as 2 dimension.

WEEK-12 :

2:10

Small World Phenomenon

Programming Screencast

Covers



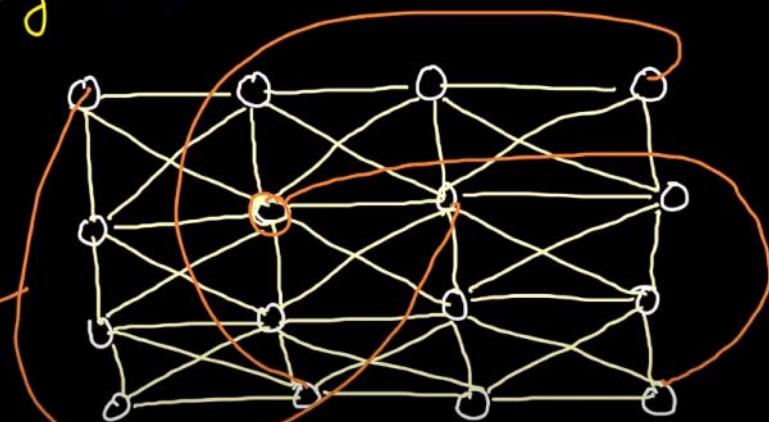
1. Making a small world network
2. Myopic Search



Making a small world network

2-D

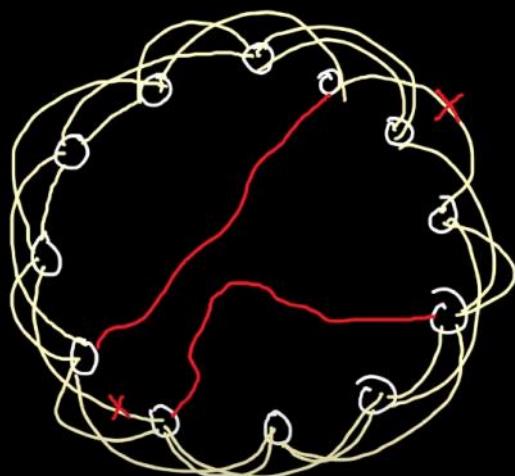
Homophily
Weak tie



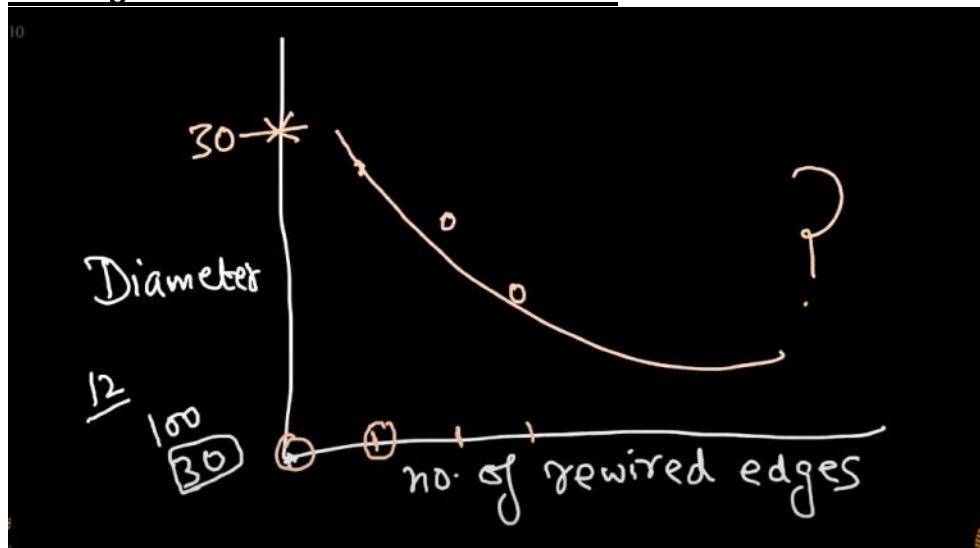
1D

Rewiring

- ① Delete
- ② Add

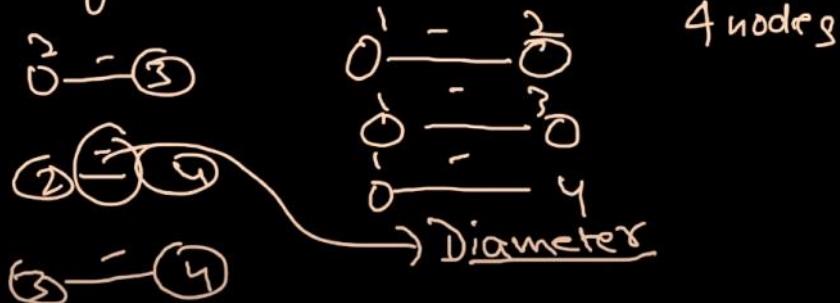


1d : ring me visualize karte hai nodes ko !!



IMP THING : IF WE DO REWIRING MORE AND MORE : THE DIAMETER OF THE GIVEN GRAPH DECREASES !!

Longest shortest distance



```
def add_edges(G):
    list_nodes=G.nodes()
    n=G.number_of_nodes()
    print list_nodes
    for i in range(0,len(list_nodes)):
        #print list_nodes[i], list_nodes[i+1],
list_nodes[i-1], list_nodes[i+2], list_nodes[i-2]
        G.add_edge(list_nodes[i], list_nodes[i-1])
        G.add_edge(list_nodes[i], list_nodes[i-2])
        target= list_nodes[i+1]
        if target>n-1:
            target=target-n
        G.add_edge(list_nodes[i], target)
        target= list_nodes[i+2]
        if target>n-1:
            target=target-n
        G.add_edge(list_nodes[i], target)
```

```
*ring.py (~) - gedit
print list_nodes
for i in range(0,len(list_nodes)):
    print list_nodes[i], list_nodes[i+1], list_nodes[i-1], list_nodes[i+2], list_nodes[i-2]
    G.add_edge(list_nodes[i], list_nodes[i-1])
    G.add_edge(list_nodes[i], list_nodes[i-2])
    target= list_nodes[i+1]
    if target>n-1:
        target=target-n
    G.add_edge(list_nodes[i], target)
    target= list_nodes[i+2]
    if target>n-1:
        target=target-n
    G.add_edge(list_nodes[i], target)
# i+1/i+2 -> n , we need to set this value to 0
#           -> n+1, we need to set them to 1
```

```

def add_long_link(G):
    v1=random.choice(G.nodes())
    v2=random.choice(G.nodes())
    while(v1==v2):
        v1=random.choice(G.nodes())
        v2=random.choice(G.nodes())
    G.add_edge(v1,v2)

```

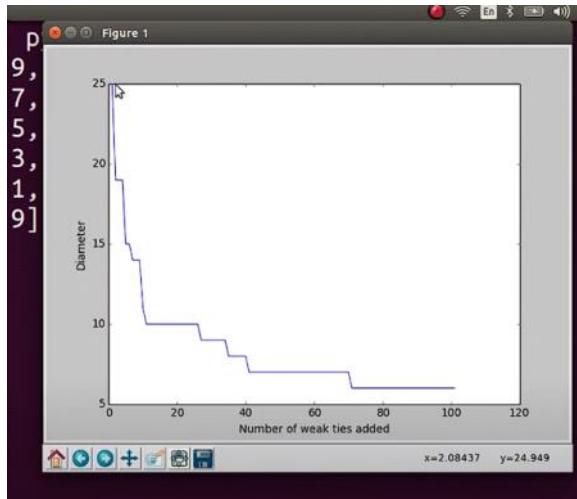
Rewiring !!

Plotting change in diameter

```

#add_long_link(G)
x=[0]
y=[nx.diameter(G)]
t=0
while(t<=100):
    add_long_link(G)
    t=t+1
    x.append(t)
    y.append(nx.diameter(G))
plt.xlabel('Number of weak ties added')
plt.ylabel('Diameter')
plt.plot(x,y)
plt.show()

```

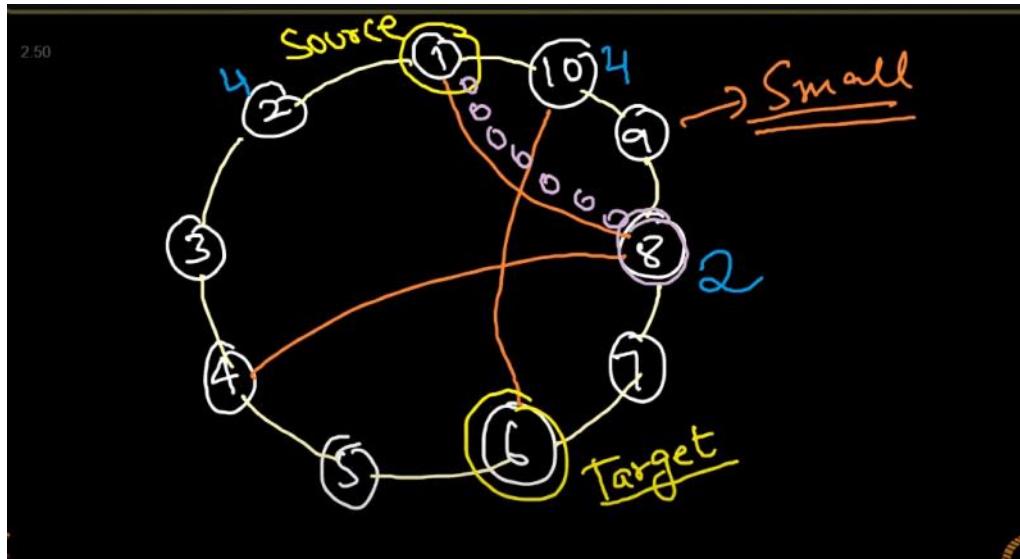


Diameter == 6 : means : at max 6 hops me : I to r pahuncha jaa sakta hai where I and r are any 2 person in the network / world !!

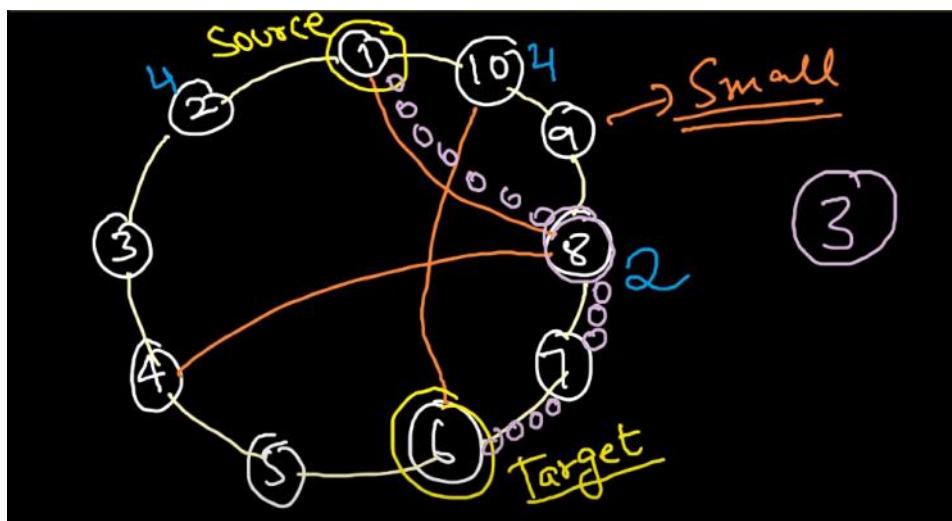
Programming illustration- Myopic Search : Introduction

MYOPIC SEARCH IS SAME AS DECENTRALIZED SEARCH !! { VVVIMP }

Note : while implementing it we need to take care of that thing in mind that : the nodes will not have info about the weak ties of its neighbours !! It will just have info about the strong ties / close wale neighbours of it's neighbour !!



For eg : here the $dis[8]$ from 6 will be calculated based on : the weak ties that is why : It will be == 2 and for 10th node it will be == 4 and so on !!



THIS PLOT WE WILL BE MAKING !! FOR MYOPIC SEARCH !!

```

def find_best_neighbor(G,c,v):
    dis=G.number_of_nodes()
    for each in G.neighbors(c):
        dis1=len(nx.shortest_path
(H,source=each,target=v))
        if dis1<dis:
            dis=dis1
            choice=each
    return choice

```

**GRAPH H ONLY CONTAINS : STRONG TIES : WEAK TIES : ISME NAHI HAI TO
SHORTEST PATH SIRF: STRONG TIES KE BASE PE NIKLEGA !!**

```

def myopic_search(G,u,v):
    path=[u]
    current=u
    while(1):
        w=find_best_neighbor(G,current,v)
        path.append(w)
        current=w
        if current==v:
            break
    return path

```

Python • Tab Width: 8 •

```

def set_myopic_path_colors(G,p):
    c=[]
    for each in G.nodes():
        if each==p[0]:
            c.append('red')
        elif each==p[len(p)-1]:
            c.append('red')
        elif each in p:
            c.append('blue')
        else:
            c.append('yellow')
    return c

```

```

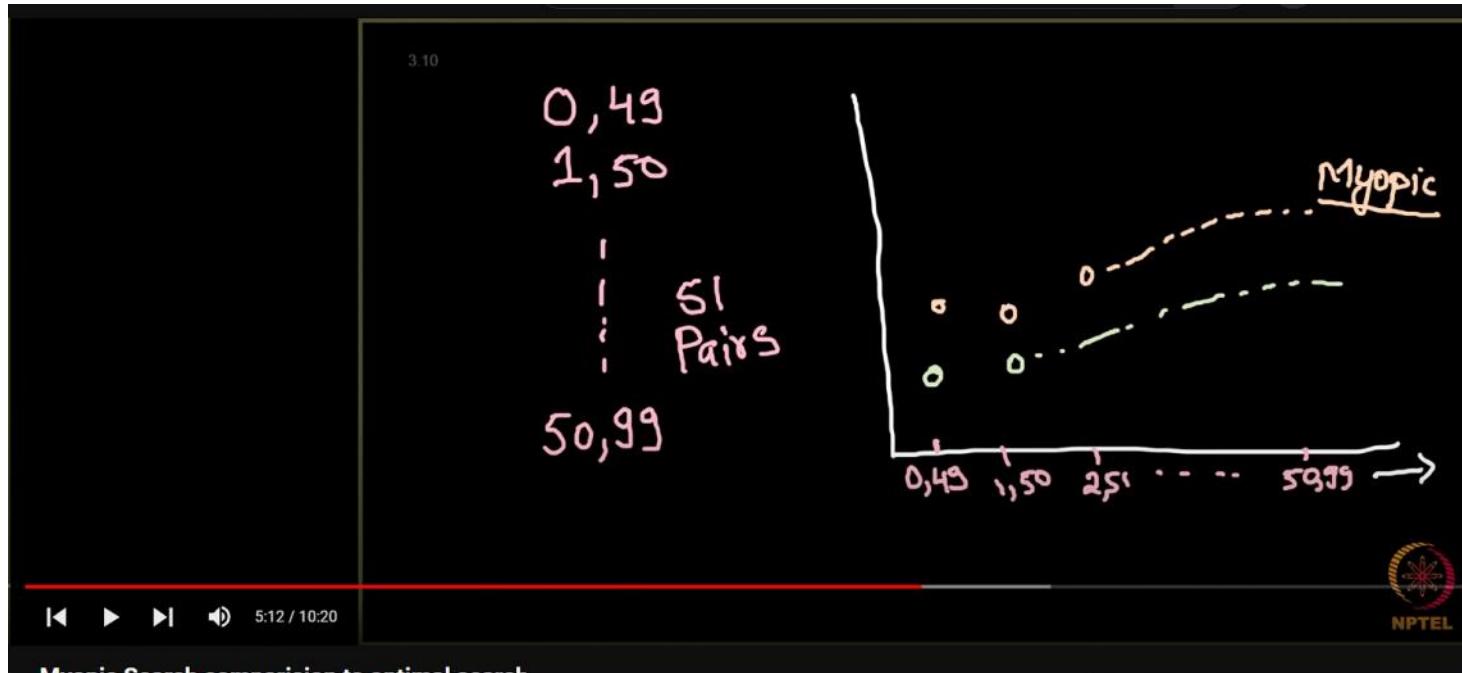
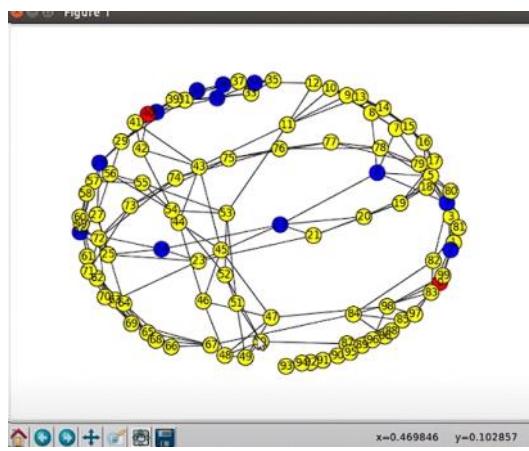
t=0
while(t<=100):
    add_long_link(G)
    t=t+1
    x.append(t)
    y.append(nx.diameter(G))
plt.xlabel('Number of weak ties added')
plt.ylabel('Diameter')
plt.plot(x,y)

p=myopic_search(G,0,40)
colors=set_myopic_path_colors(G,p)

nx.draw(G, node_color=colors)
plt.show()

```

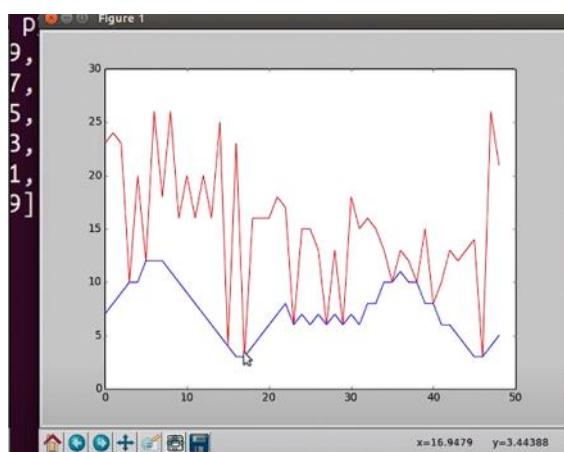
Python •



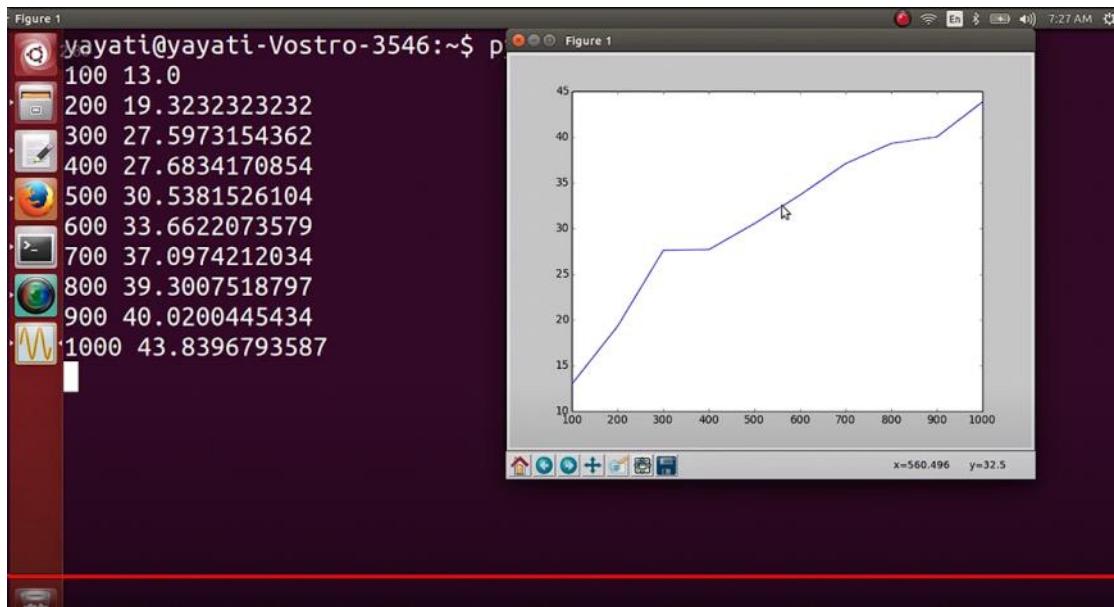
Myopic Search comparision to optimal search

These 2 distance comparison will be done for diametrically opposite ends !!

bcz unke beech me distance == max !!



Red graph is for myopic !!

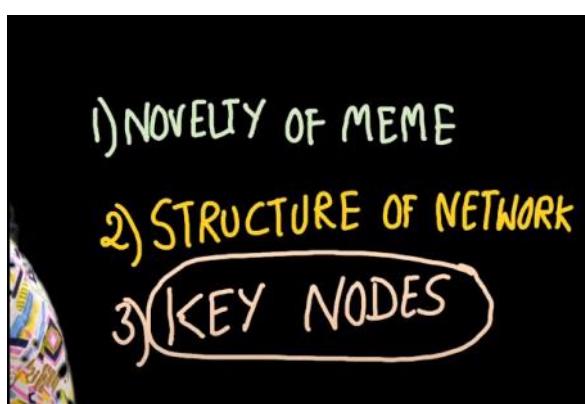


TIME TAKEN BY MYOPIC SEARCH INCREASES !! AS THE NODES INCREASES !!

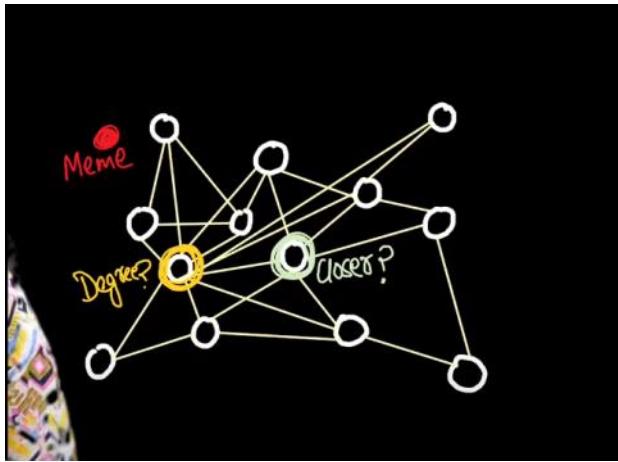
It is NOT A LINEAR CURVE IT IS A LOG PLOT !!



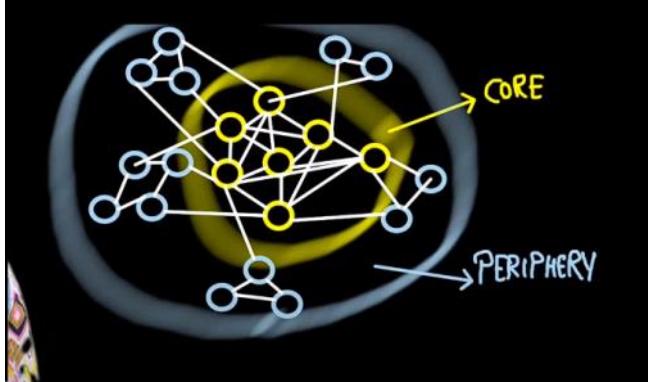
Structure is same / constant for a social media platform !!



Same idea as marketing and all !!



Finding the right key nodes (the core)



Certainly! Core-periphery networks are a significant concept in network theory, which is a field that studies the relationships between nodes (or entities) within complex systems. Let's delve into more detail about core-periphery networks:

1. **Definition:** A core-periphery network is characterized by a central core of highly interconnected nodes surrounded by a periphery of less densely connected nodes. The core nodes typically have many connections to other core nodes, while peripheral nodes have fewer connections and may primarily connect to the core nodes.
2. **Structure:** The structure of a core-periphery network can vary based on the density and strength of connections between nodes. In some cases, the core might consist of a small group of highly central nodes with many connections among them, while the periphery contains a larger number of nodes with fewer connections, primarily to the core.
3. **Examples:**
 - * **Social Networks:** In social networks, the core might represent individuals who are central to social interactions within a community or organization, such as influential leaders or key decision-makers. The periphery could include individuals who have fewer social connections or who are on the outskirts of the social group.

adoption of behaviors, and the spread of ideas or opinions.

2. Cascading Behavior:

- Cascading behavior refers to the spread of influence, behaviors, or actions through a network, often triggered by the actions of a small subset of nodes.
- In the context of social networks, cascading behavior can manifest in various forms, such as the adoption of new trends, the propagation of rumors or gossip, or the diffusion of innovations.
- Core-periphery networks can influence the dynamics of cascading behavior:
 - **Propagation through the Core:** Cascading behavior may spread more rapidly and extensively within the core of the network, where nodes are highly interconnected and influential.
 - **Propagation to the Periphery:** Once cascading behavior gains traction in the core, it may cascade outward to the periphery, reaching individuals with weaker ties to the core.
- **Resilience and Containment:** The core-periphery structure can affect the resilience of cascading behavior. If the core is resilient to external influences, cascades may be contained within the core without spreading to the periphery. However, if the core is receptive to external stimuli, cascades may propagate throughout the network more broadly.

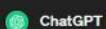
3. Examples:

A famous example is the 2007 Facebook Outbreak.



You

What is core of a network ? while spreading an idea ?

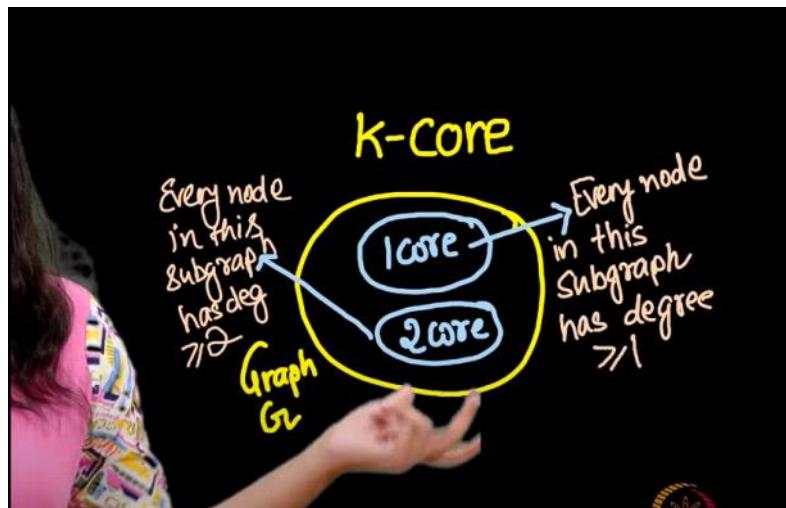


ChatGPT

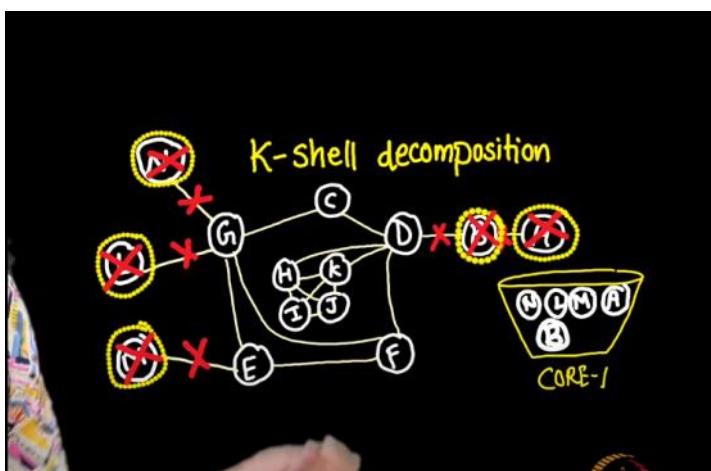
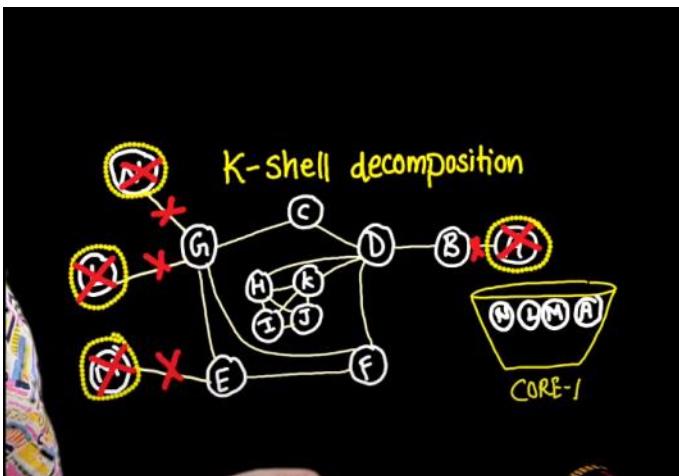
In the context of spreading an idea within a network, the "core" refers to the central or influential nodes within the network that play a crucial role in the dissemination and adoption of the idea. These core nodes typically have a higher degree of connectivity, influence, or centrality compared to other nodes in the network.

Identifying and targeting the core of a network can be an effective strategy for spreading ideas because these influential nodes have the potential to reach a large number of other individuals within the network. By engaging with or gaining the support of these core nodes, an idea can quickly gain momentum and spread more efficiently throughout the network.

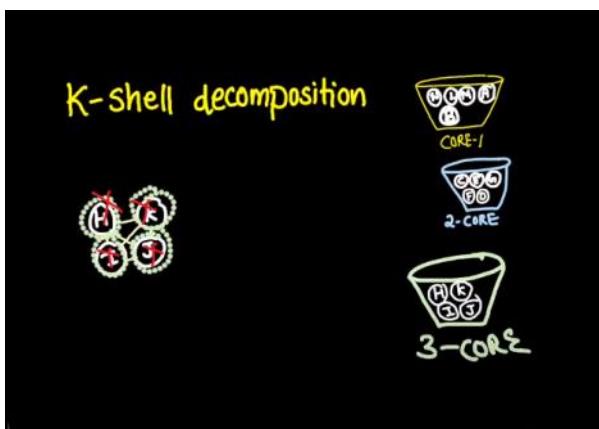
Network analysis techniques, such as social network analysis, can be used to identify the core nodes within a network based on factors such as degree centrality, betweenness centrality, or eigenvector centrality. Once identified, efforts can be focused on engaging with these core nodes to maximize the impact of the idea being spread.

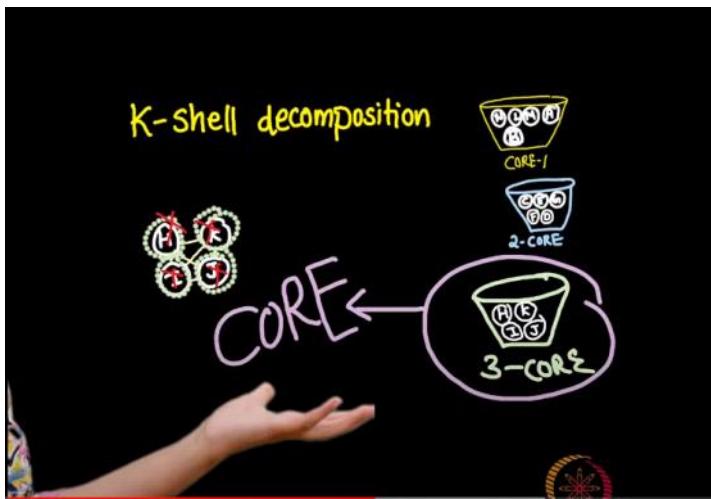


K-Shell Decomposition Algorithm :



SO DOING THIS K TIMES WILL LEAD YOU TO A K CORE !! { VVMUCH SIMILAR TO KAHN'S ALGO }





THESE NODES FORM THE CORE OF THE NETWORK !!

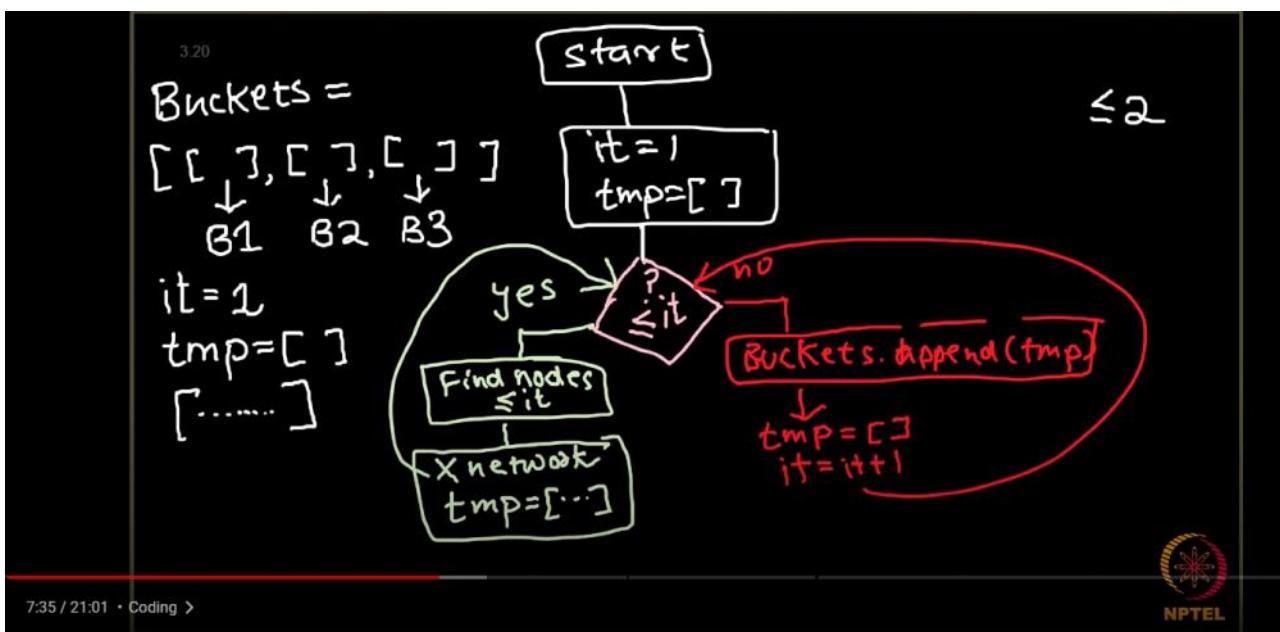
2.90

$$1\text{-core} = B_1 \cup B_2 \cup B_3$$

$$2\text{-core} = B_2 \cup B_3$$

$$3\text{-core} = B_3$$

B1 , B2 ... BK : ARE THE BUCKETS !! FOR CORE K IN THE K SHELL DECOMPOSITION !!!



K Shell Decomposition : Pseudo Code !!

EXPLANATION OF INTUTION BEHIND K SHELL DECOMPOSITION IN DEPTH :

A) Steps followed :

- Take all the nodes that have degree == 1 : unhe 1 core bucket me dalo
Repeat this step till you don't get a node with degree >=2
- Now take all nodes with degree <=2 and do it till you don't get a element with degreee>=3 and put it in : 2 core bucket !!

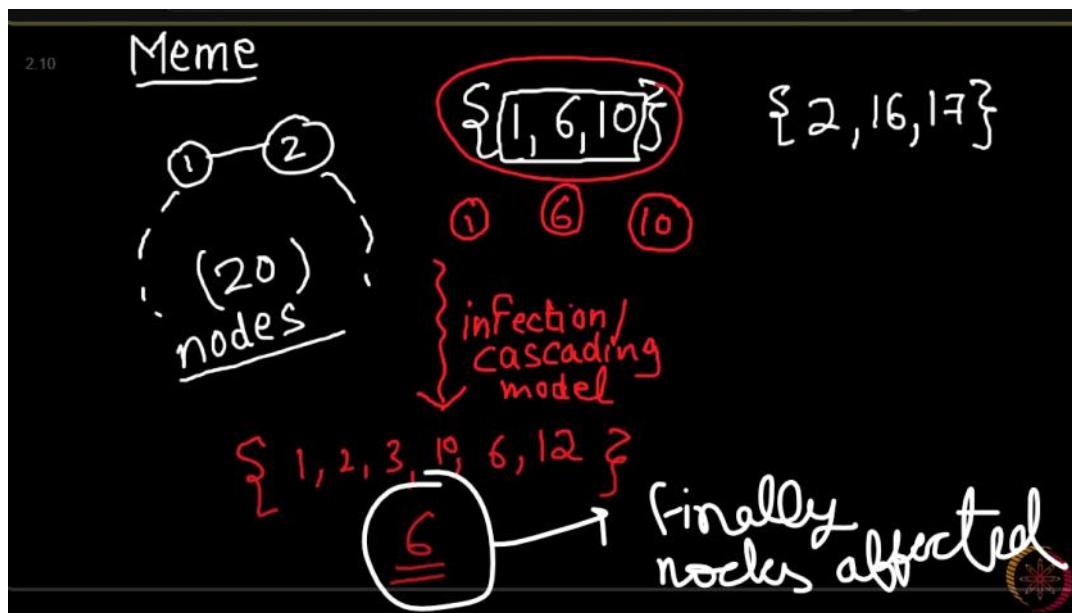
And do it tell network != empty

NOTE :

- K CORE = BUCKET K UNIONLAST BUCKET !!

INTUTION : SEE VIDEO TO GET THAT !! { BASICALLY YAAR HAM BUCKET K ME : <=K WALE SARE ISLIYE ALLOW KAR RAHE HAI BCZ : 1 DEGREE WALE BHI : K DEGREE WALE HI HAI BUT JUST DIKAT YE HAI KI UNKE K-1 NEIGHBOURS HAT GAYE AND BUCKET ME JAA CHUKE HAI !! }

DETERMINING INFLUENTIAL NODES IN A NETWORK :

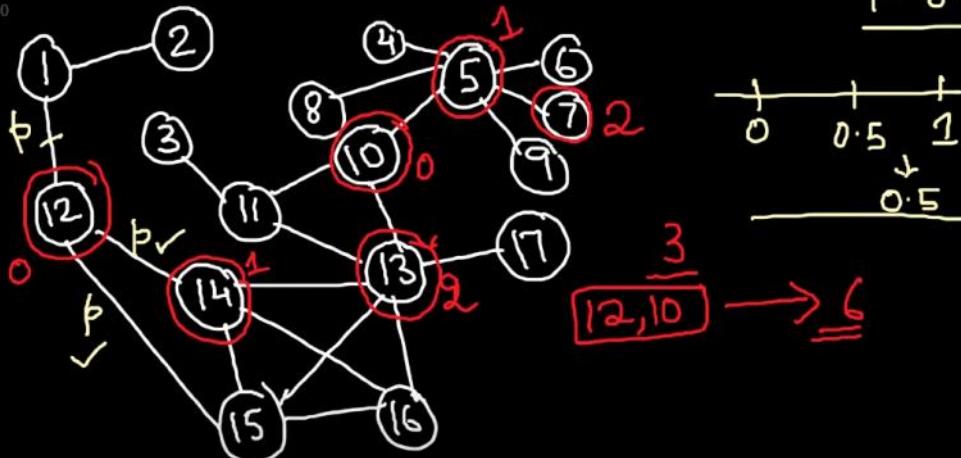


Infection/Cascade Model

Independent Cascade Model



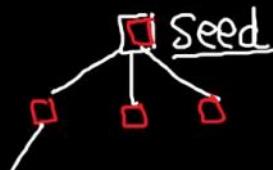
$$P=0.5$$



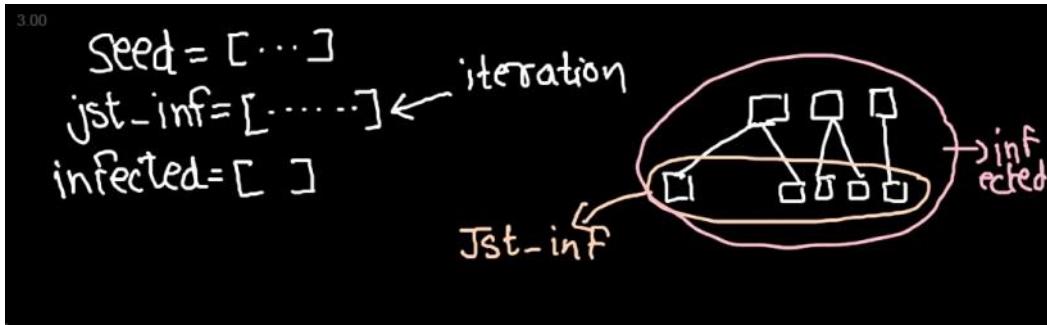
WHY IT IS CALLED INDEPENDENT CASCADING MODEL ?

BCZ : WHETHER : 12 WILL PASS THE IDEA TO : 14 IS INDEPENDENT OF ITS EDGE BTW 1 AND 12 !! I.E ITS OTHER NEIGHBOUR !!

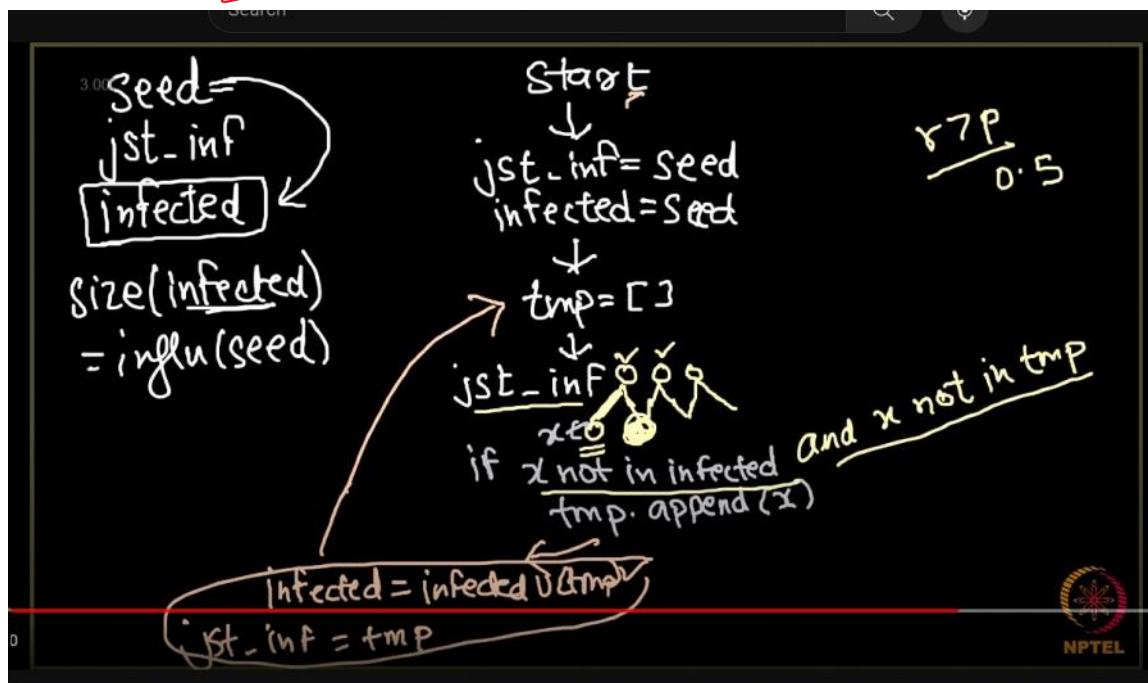
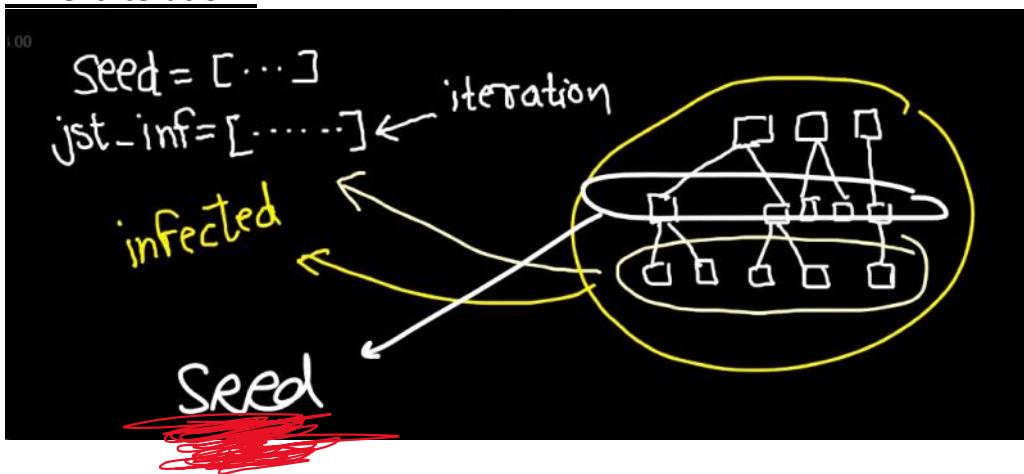
Independent Cascade Model



SIMILAR TO SIR : EK BAAR : JO AFFECT HO GAYE THE : UNHE LATER STAGE ME KOI AFFECT NAHI KAR SAKTA !!



In next iteration :



```

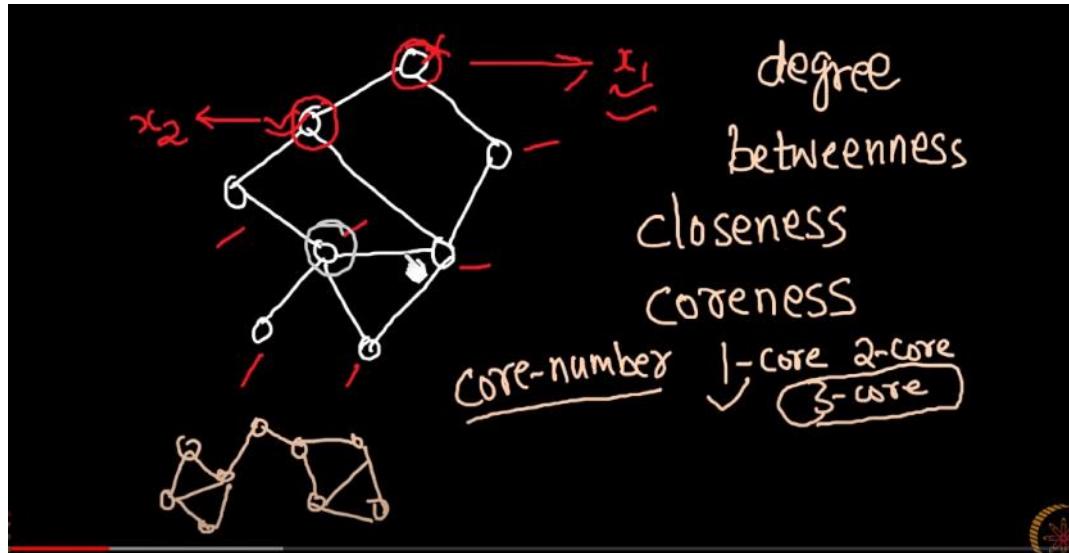
def ic(G,s):
    jst_inf=list(s)
    infected=list(s)
    while(1):
        print jst_inf
        if len(jst_inf)==0:
            return infected
        tmp=[]
        for each in jst_inf:
            for each1 in G.neighbors(jst_inf):
                r=random.uniform(0,1)
                if r<0.5 and each1 not in
infected and each1 not in tmp:
                tmp.append(each1)
        for each in tmp:
            infected.append(each)
        jst_inf=list(tmp)
    return infected
tmp=[]
for each in jst_inf:
    for each1 in G.neighbors(jst_inf):
        r=random.uniform(0,1)
        if r<0.5 and each1 not in
infected and each1 not in tmp:
            tmp.append(each1)
    for each in tmp:
        infected.append(each)
jst_inf=list(tmp)

```

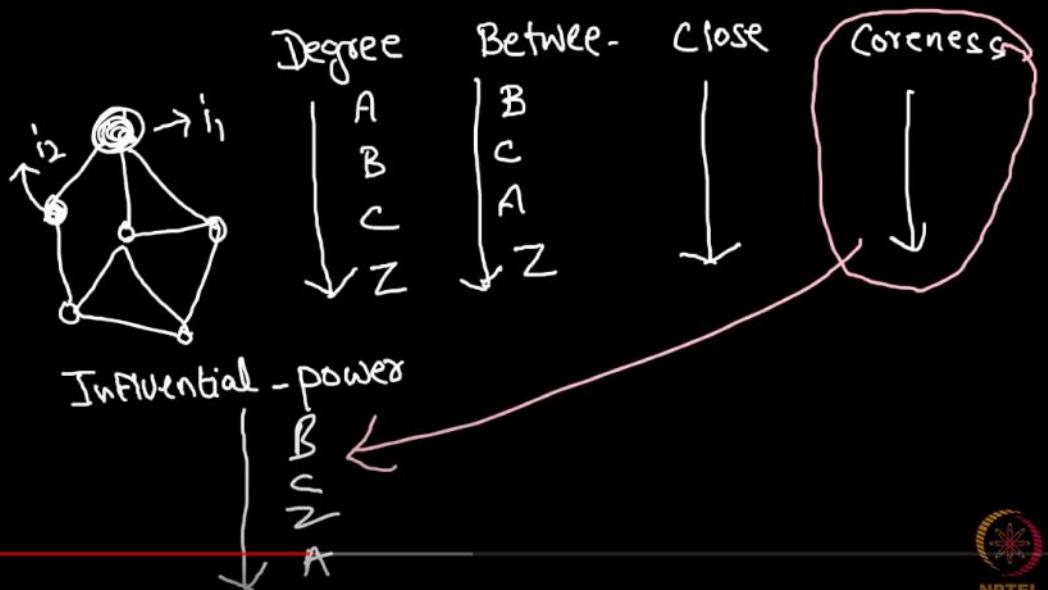
Copy, Download

THE FIRST NODE THAT GETS INFECTED IS CALLED : SEED NODE !!

RATHER A LESS TIME TAKING WAY TO FIND THE : MOST INFLUENTIAL NODES IN THE NETWORK : IS USING THESE CENTRALITY MEASURES !!!



1 NODE BELONGING TO 3-CORE : IS HAVING CORENESS CENTRALITY MEASURE == 3 !!



Coding the importance of core nodes in cascading

Social Networks



RESULT : CORENESS CENTRALITY IS THE BEST MEASURE FOR THE INFLUENTIAL POWER OF A NODE !!

```

dict_deg={}
dict_cl={}
dict_bw={}
dict_cr={}

for each in G.nodes():
    dict_deg[each]=G.degree(each)
    dict_cl[each]=nx.closeness_centrality(G,each)
    dict_bw[each]=nx.betweenness_centrality(G,each)
    dict_cr[each]=nx.core_number(G)[each]

```

Nx.core_number(G) is a dictionary !! { VVIMP !! }

```

dict_cascade={}

for each in G.nodes():
    c=[]
    for num in range(0,1000):
        seed=[each]
        i = ic(G,seed)
        c.append(len(i))
    dict_cascade[each]=np.average(c)

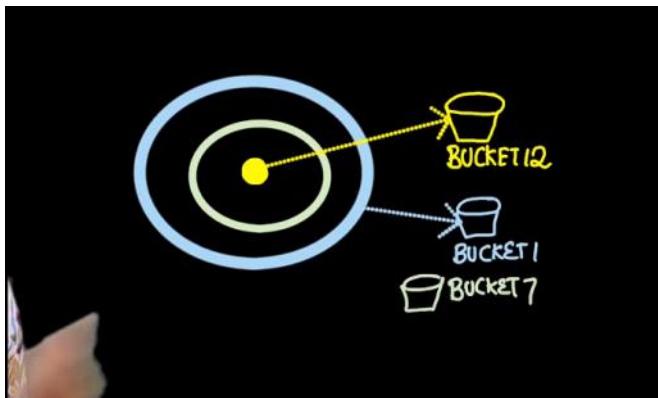
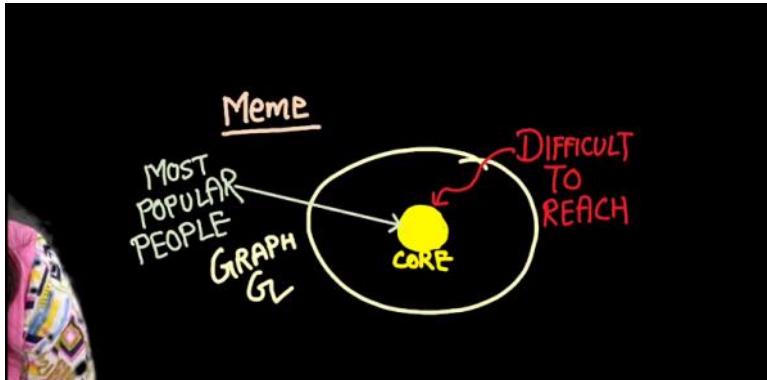
```

```

sorted_dict_cascade=sorted(dict_cascade,key=dict_cascade.get,
reverse=True)
sorted_dict_deg=sorted(dict_deg,key=dict_deg.get, reverse=True)
sorted_dict_cl=sorted(dict_cl,key=dict_cl.get, reverse=True)
sorted_dict_bw=sorted(dict_bw,key=dict_bw.get, reverse=True)
sorted_dict_cr=sorted(dict_cr,key=dict_cr.get, reverse=True)

```

```
yayati@yayati-Vostro-3546:~$ python tc.py
Nodes sorted according to degree
[5, 13, 14, 15, 10, 11, 16, 12, 1, 2, 3, 4, 6, 7, 8, 9, 17]
Nodes sorted according to closeness
[10, 13, 5, 11, 14, 15, 16, 17, 4, 6, 7, 8, 9, 3, 12, 1, 2]
Nodes sorted according to bbetweenness
[3, 4, 11, 14, 10, 16, 17, 13, 9, 12, 8, 15, 5, 7, 6, 2, 1]
Nodes sorted according to coreness
[13, 14, 15, 16, 10, 11, 12, 1, 2, 3, 4, 5, 6, 7, 8, 9, 17]
Nodes sorted according to influence
[13, 15, 14, 10, 16, 5, 11, 12, 17, 6, 4, 7, 8, 3, 9, 1, 2]
yayati@yayati-Vostro-3546:~$
```



Wrt k shell decomposition !!

INFLUENCE POWER : IMP TERM : SAME AS CASCADE CAPACITY OF A SHELL :
MEANS : AGAR KISI EK BUCKET KO INFECT KAROGE AS SEED : THEN HOW MANY
PEOPLE WILL GET INFECTED IN TOTAL IN THE NETWORK !!

RESULT :

IF WE PLOT SHELL NO VS CASCADE CAPACITY GRAPH : THE GRAPH WILL INC , INC AND AFTER SOME CORE : WILL COME TO A SATURATION POINT !!

THERE FORE IT IS NOT NECESSARY TO ALWAYS INFECT THE MOST CORE SHELLS !!
TO MAKE YOUR IDEA GO VIRAL !!

SO THESE NOT INNERMOST BUT RATHER : A MORE INNER : KIND OF NODES THAT HAVE APPROX SAME CASCADING CAPACITY AS THE INNERMOST ONES ARE CALLED AS : PSEUDO CORES !!

2) As per the definition, in a core-periphery structure (Choose the TRUE statement): **1 point**

- Low status people are linked in densely connected core while the high status people atomize around this core as periphery of the network.
- Core and the periphery occupy interchangeable positions in the network.
- The notion of a node being in a core or in a periphery does not depend on the social status or the wealth of a node.
- High status people are linked in densely connected core while the low status people atomize around this core as periphery of the network.

3) The i^{th} iteration of k-shell decomposition algorithm **1 point**

-
- Removes all the nodes of degree i from the graph.
-
- Recursively keeps removing the nodes of degree i from the graph, i.e., keeps removing the degree i nodes from the graph till there are no degree i nodes in the graph.
-
- Recursively keeps removing the nodes of degree $\leq i$ from the graph, i.e., keeps removing the nodes of degree $\leq i$ from the graph till there are no degree $\leq i$ nodes in the graph.
-
- Recursively keeps removing the nodes of degree $\geq i$ from the graph, i.e., keeps removing the node of degree $\geq i$ from the graph till there are no degree $\geq i$ nodes in the graph.

4) Identify the 1-core, 2-core and 3-core in the Figure 1 (Based on the definition of k-core) **1 point**

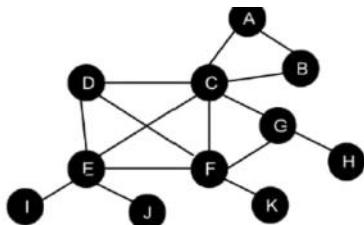


Figure 1: The network

- 1-core: A,B,C,D,E,F,G,H,I,J,K ; 2-core: A,B,C,D,E,F,G ; 3-core: C,D,E,F
- 1-core: H,I,J,K ; 2-core: A,B,G ; 3-core: C,D,E,F
- 1-core: C,D,E,F ; 2-core: A,B,C,D,E,F,G ; 3-core: A,B,C,D,E,F,G,H,I,J,K
- 1-core: I,J,K,H,A,B,G ; 2-core: Empty ; 3-core: C,D,E,F

SUPER DUPER IMP AND TRICKY QUESTION :

**1 CORE = BUCKET 1 UNION BUCKET 2 BUCKET N
K CORE = BUCKER K UNION BUCKET K+1BUCKET N**

✓ A, G and E

8) The core number of a node in a graph can not be

- Greater than its degree
- Lesser than its degree
- Equal to its degree
- Can't say

10) In a clique of size 5

- Every node has a coreness of 4
- Every node has a coreness of 5
- Every node has a coreness of 6
- Every node has a coreness of 7

i 9) Given that a node in a network is a part of exactly 3 cliques (complete subgraphs), **1 po**, of size 3, size 4 and size 6 respectively. Then the core number of this node can not be

- greater than 3
- less than 7
- less than 6
- less than 5

LOGICAL WHY ?

EVEN IF : BAAKI NODES : HAT BHI JAAYENGE LESSER ITERATIONS ME : YE NODE PAKE SE : HAT NAHI PAYEGA : TILL :5TH ITERATION NA AAJAYE BCZ TABTAK ISKI DEGREE>=5 HOGI !! AND THEREFORE KOI BHI ISEE NHI UDA PAYEGA !!