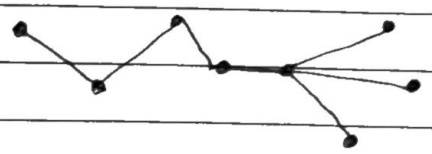
SOCIAL NETWORK ANALYSIS

A network is a collection of objects where some pairs of objects are connected by links.

(N) objects: nodes, vertices

(E) interactions: links, edges

G(N, E) System: network, graph



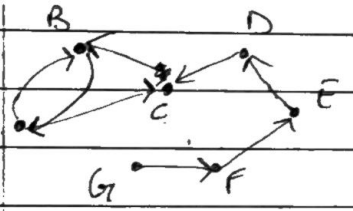
Network often refers to real systems eg. web, social network, And graph is a mathematical representation of a network. eg. web graph, social graph, knowledge graph.

~~eg: If you connect all papers with the same word~~

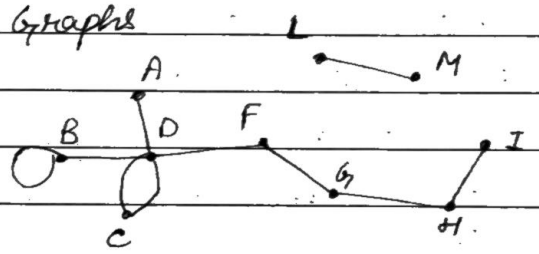
eg: If you connect scientific papers that cite each other, you will work and studying the citation network.

Choice of Network Representation :-

① Directed vs. Undirected Graphs



eg: phone calls, following on twitter



eg: Collaborations, friendship on facebook

② Node degrees (K_i): the no. of edges adjacent to node 'i'

avg. degree ($\bar{K}$): $\frac{1}{N}\sum_{i=1}^{N} K_i = \frac{2E}{N}$ sum of degrees = 2E

where as in directed graph we have in-degree and out-degree and total degree = in-degree + out degree

$K_C^{in} = 2$; $K_C^{out} = 1$ $K_C = 3$

DATE : / /
PAGE NO. :

at source node $K^{in} = 0$ and
at sink node $K^{out} = 0$

$$\overline{K} = \frac{E}{N} \qquad \overline{K^{in}} = \overline{K^{out}}$$

③ Complete Graph – The max number of edges in an undirected graph on N nodes is

$$E_{max} = \binom{N}{2} = {}^{N}C_2 = \frac{N(N-1)}{2}$$

$$\overline{K} = N-1$$

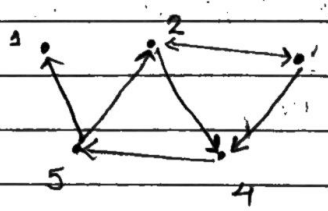
Representation of graphs :-

Adjacency matrix :

→ For an undirected graph matrix is symmetric.
→ For a directed graph matrix is not symmetric.
→ Adjacency matrix is easier to work with if network is large or sparse.

Edge list :

→ Allows us to quickly retrieve all neighbours of a given node



1 :
2 : 3, 4
3 : 2, 4
4 : 5
5 : 1, 2

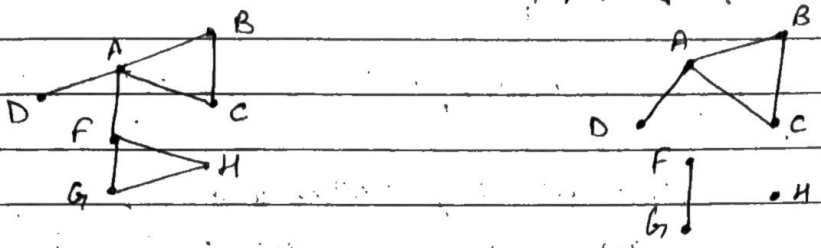
dense sparse

DATE: / /
PAGE NO.:

NOTE: Most real-world networks are sparse.
$(E << E_{max})$ or $(K << N-1)$
Consequence: adjacency matrix is filled with zeros.
density of the matrix = E/N^2

Connectivity of undirected graphs.

In a connected-undirected graph, any two vertices can be joined by a path. And a disconnected graph is made up by two or more connected components.



Bridge edge - If we erase the edge, the graph becomes disconnected
like AF edge
Articulation node - If we erase the node, the graph becomes disconnected
like A and F

Network Properties :-

Degree distribution: $P(K)$
Path length: h
Clustering coefficient: C
Connected components: s

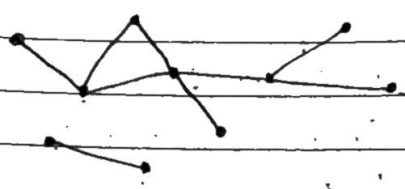
① Degree distribution $P(K)$: Probability that a randomly chosen node has degree k.

N_K = no. of nodes with degree K.

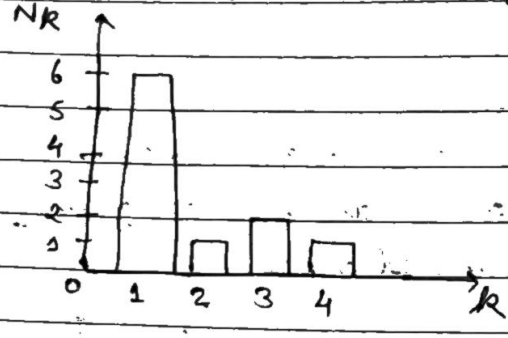
Normalized histogram: $P(K) = \frac{N_K}{N}$ → total no of nodes

DATE: / /
PAGE NO.:

undirected graph



$N_1 = 6$
$N_2 = 1$
$N_3 = 2$
$N_4 = 1$



For directed graphs we have separate in and out degree distributions

NOTE:- In social media there are many networks with millions of nodes and billions of edges. — They are complex and it is difficult to analyze them. Therefore we design models that generate graphs. The generated graphs should be similar to real-world networks. And if we can guarantee that generated graphs are similar to real-world networks:

1. We can analyze simulated graphs instead of real-networks (cost-efficient).
2. We can better understand real-world networks by providing concrete mathematical explanations.
3. We can perform controlled experiments on synthetic networks when real world networks are unavailable.

The pareto principle :- In case of global distribution of wealth most individuals have average capitals, and only few are considered wealthy i.e exponentially more individuals with average capital than the wealthier ones. Similary, if we consider city population only a few metropolitan areas are densely populated and most cities

DATE : / /
PAGE NO. :

have an average population size. This is the same phenomenon observed regularly in social media when measuring popularity or interestingness for entities. This phenomenon
This is called Pareto Principle (80-20 rule): 80% of the effects come from 20% of the causes.

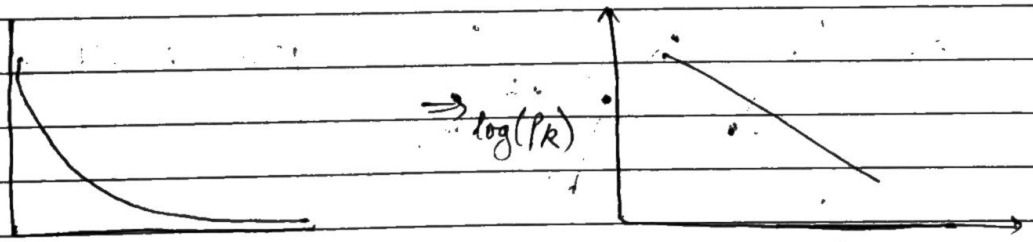
Based on the above concept we have ~~degree distribution~~ power law degree distribution - when the frequency of an event changes as a power of an attribute. The frequency follows a power-law.

$$P_d = a\, d^{-b}$$

- P_d: fraction of users with degree 'd'.
- a: power-law intercept
- d: node degree
- b: the power-law exponent & its value is typically in the range of [2, 3].

$$\therefore \ln P_d = -b \ln d + \ln a$$

many real world networks exhibit a power-law distribution as seen above. Power law seems to dominate when the quantity being measured can be viewed as a type of popularity i.e quantity is ~~measured~~ computed based on the ~~no~~ popularity. In power law distribution small occurences is common and large instances is extremely rare.



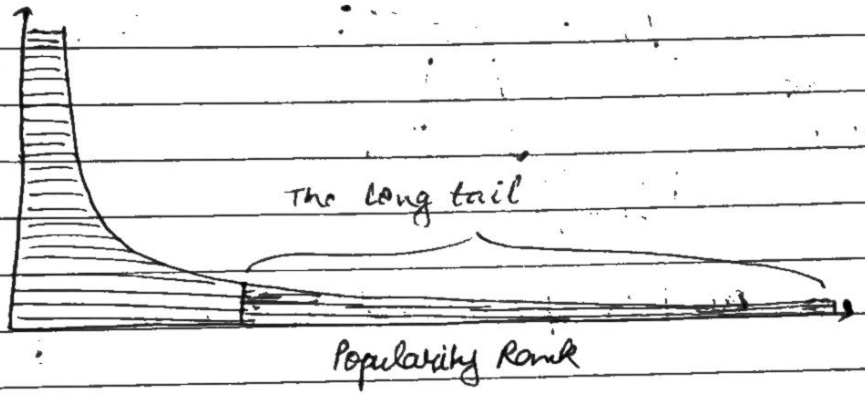
Power law distribution

log-log plot of power law distribution.

DATE: / /
PAGE NO.:

Networks with a power-law degree distribution are called scale-free networks. The tail of the graph of power law distribution is long! telling that ~~most sales being generated by a small set of items that are enormously popular and~~ most sales are being generated by a large set of items that are each individually less popular (or in less demand) when considering a business sales problem yet producing more sales

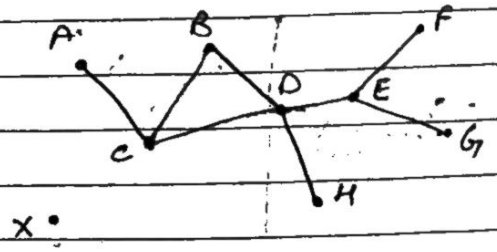
eg: 57% of Amazon's sales is from the long tail.



(2) Paths in a graph (h)

A path is a sequence of nodes in which each node is linked to the next one. A path can intersect itself and pass through the same edge multiple times.

eg:

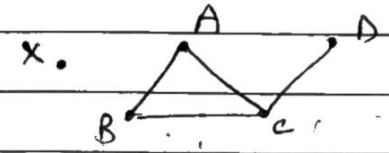


path - ACBDCBEG

DATE : / /
PAGE NO. :

Distance in a graph

- distance between a pair of nodes is defined as the number of edges along the shortest path connecting the nodes And if the two nodes are not connected, the distance is usually defined as infinite (or zero)

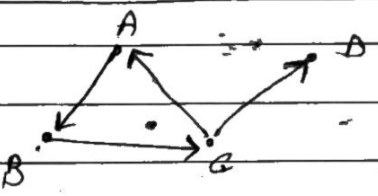


$h_{B,D} = 2 \quad h_{A,X} = \infty$

- In directed graphs, paths need to follow the direction of the arrows. Telling that distance is not symmetric i.e

$h_{B,C} \neq h_{C,B}$

$1 \neq 2$



Diameter :- The maximum (shortest path) distance between any pair of nodes in a graph.

NOTE: Average path length ($\bar{h}$) :- for a connected graph or a strongly connected directed graph

$$\bar{h} = \frac{1}{2 E_{max}} \sum_{i,j \neq i} h_{ij}$$

h_{ij} is the distance from node i to node j

E_{max} is the max no of edges (total no of node pairs) $= \frac{n(n-1)}{2}$

max edges possible (E_{max}) $= {}^{n}C_2$

note: we compute the average path length only over connected pairs of nodes and ignore ∞ length paths.

DATE / /
PAGE NO

(3) Clustering Coefficient (C_i)

How connected are i's neighbors to each other

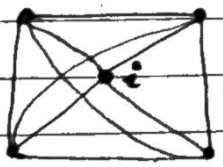
Node i with degree k_i

$C_i \in [0, 1]$

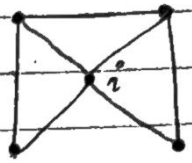
$$C_i = \frac{2e_i}{k_i(k_i - 1)}$$

where e_i is the no of edge. between the neighbors of node i

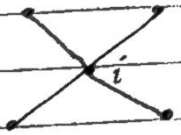
where, $\frac{k_i(k_i-1)}{2}$ is the max no of edges between the k_i neighbors



$C_i = \frac{2 \times 6}{4(3)} = 1$



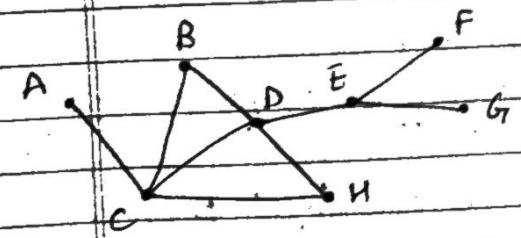
$C_i = \frac{2 \times 3}{4 \times 3} = \frac{1}{2}$



$C_i = \frac{2 \times 0}{4 \times 3} = 0$

Clustering coefficient is undefined (or defined to be 0) for nodes with degree 0 or 1.

Average clustering coefficient: $C = \frac{1}{N} \sum_{i=1}^{N} C_i$



$C_B = \frac{2 \times 1}{2 \times 1} = 1$

$C_D = \frac{2 \times 2}{4 \times 3} = \frac{1}{3}$

A giant component is a subgraph of a graph in which a significant fraction of the nodes are connected to each other through a path of edges. The presence of giant component in a graph is important for GNNs bcz it allows information to flow more easily between nodes through the message passing mechanism. In dense graph with a giant component, even nodes that are not directly connected (4) to each other can share information through the common connections they have with other nodes in the giant component.

Connectivity: Size of the largest connected component i.e largest set in the graph where any two vertices can be joined by a path.

largest component = giant component.

~~How to find~~

Erdös - Rényi Random graph model

Two variants:

→ G_{np} : undirected graph on n nodes where each edge (u, v) appears with probability p independent from every other edge.

→ G_{nm} : undirected graph with n nodes, and m edges picked uniformly at random.

G_{np}

- Degree Distribution (P_k) of G_{np} is binomial.

let $P(k)$ denote the fraction of nodes with degree k

$$P(k) = \binom{n-1}{k} p^k (1-p)^{n-1-k}$$

$\binom{n-1}{k}$: select k nodes out of $n-1$

p^k: probability of having k edges

$(1-p)^{n-1-k}$: probability of missing the rest of the $n-1-k$ edges

(average degree) mean $\bar{k} = p(n-1)$

variance $\sigma^2 = p(1-p)(n-1)$

$$\frac{\sigma}{\bar{k}} = \left[\frac{1-p}{p}\frac{1}{(n-1)}\right]^{1/2} \approx \frac{1}{(n-1)^{1/2}}$$

By the law of large no., as the network size increases, the distribution becomes increasingly narrow

DATE : / /
PAGE NO :

- Clustering coefficient of G_{np}

$$C_i = \frac{2e_i}{k_i(k_i - 1)}$$

edges in G_{np} appear with prob. p

$$e_i = p \times \frac{k_i(k_i-1)}{2}$$

p: each pair is connected with prob. p

$\frac{k_i(k_i-1)}{2}$: no. of distinct pairs of neighbors of node i of degree k_i

$$C_i = \frac{p \cdot k_i(k_i-1)}{k_i(k_i-1)} = p = \frac{\bar{k}}{n-1} \approx \frac{\bar{k}}{n}$$

Clustering coefficient of a random graph is small. If we generate bigger and bigger graphs with fixed avg. degree $\bar{k}$ (that is we set $p = \frac{\bar{k}}{n}$) then C decreases with the graph size n.

- Average path length = $O(\log n)$
- giant component
- "Evolution" of a random graph.

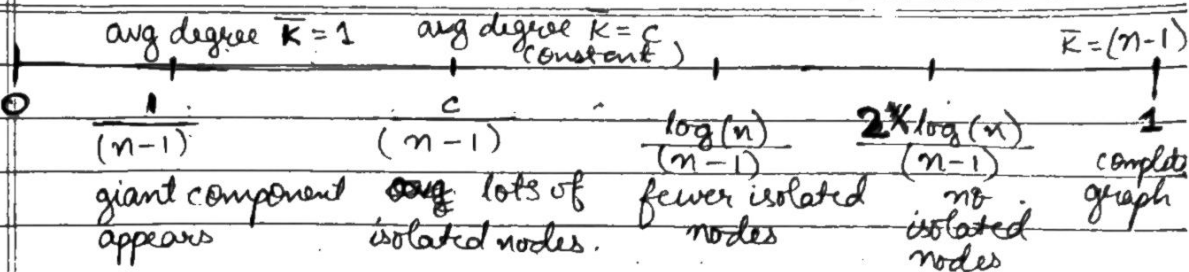
when average degree ($\bar{k} = 1$), giant component appears. avg degree $\bar{k} = p(n-1)$

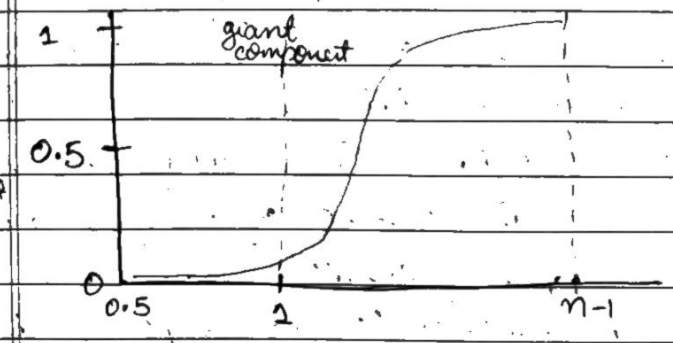
if $\bar{k} = 1$ we have $p = \frac{1}{n-1}$

$k = 1 - \varepsilon$ = all components are of size $\Omega(\log n)$

$k = 1 + \varepsilon$ = only 1 component is of size $\Omega(n)$, others have size $\Omega(\log n)$ and each node has at least one edge in expectation.

DATE : / /
PAGE NO :





$\bar{k} = p(n-1)$

Real Networks vs. Gnp :-

When compared Gnp random graph model with the real world network

Degree distribution — ✗

Avg path length — ✓

avg Clustering coefficient — ✗

('giant component') Connected components — ✓

Problems with Gnp (random network model):

- Degree distribution differs from that of real networks
- Giant component in most real networks does NOT emerge through a phase transition
- No local structure – clustering coefficient is too low

Except for the small world property, the properties observed for the real world networks are not matching

DATE: / /
PAGE NO.:

with that observed for random networks.
Although G(n,p) ~~can be used~~ is useful for following purposes:-

(1) If a certain property is observed for real world networks, we can refer to the random graph theory and analyze whether the property is observed by chance (like the small world property).
(2) If the property observed does not coincide with that of the random networks (like the local clustering coefficient), we need to further analyze the real world network for the existence of the property bcoz it did not just happen by chance.
(3) Establish useful benchmarks (eg for component structure, diameter, degree distribution, clustering etc).
(4) It is the reference model

Small-World Model

note:- small world model doesn't care about degree distribution

goal: High Clustering coefficient
Low average path length

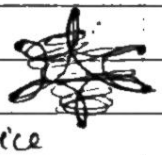
A small world network is a type of mathematical graph in which most nodes are not neighbors of one another, but the neighbors of any given node are likely to be neighbors of each other and most nodes can be reached from every other node by a small number of hops or steps.

A small world network is defined to be a network where the typical distance L between two randomly chosen nodes grows proportionally to the ~~log~~ logarithm of the number of nodes N in the network

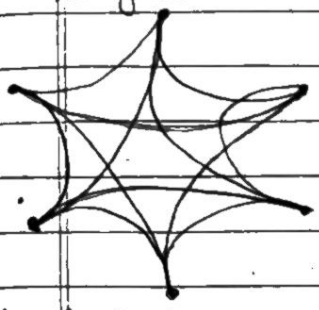
$L \propto \log N$

DATE : / /
PAGE NO. :

could a network with high clustering also be small world (have $\log n$ diameter)? Or How can we have high clustering and small diameter?



Ring lattice



High clustering
High diameter

Sol^n to this is clustering implies edge "locality", Randomness enables "shortcuts"

Two components to the model:

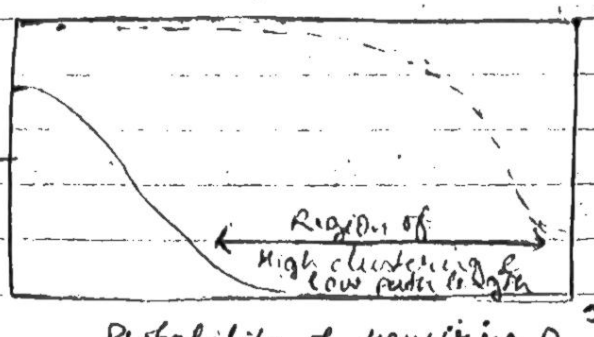
1) Start with a low dimensional regular lattice
   ↳ High clustering coefficient
2) Rewire: Introduce randomness ("shortcuts")
   ↳ Add/remove edges to create shortcuts to join remote parts of the lattice.
   ↳ For each edge, with probability 'p' move the other endpoint to, a random node.

Regular network	Small world network	Random network
$p = 0$	'p' increases randomness	$p = 1$
High clustering	High clustering	Low clustering
High diameter	Low diameter	Low diameter
$h = \frac{N}{2\bar{K}}$ $C = \frac{1}{2}$		$h = \frac{\log N}{\log \bar{K}}$ $C = \frac{\bar{K}}{N}$

Rewiring allows us to interpolate between a regular lattice and a random graph.

DATE / /
PAGE NO

Excess of randomness can ruin the clustering.



Could a network with high clustering be at the same time a small world?

=> Yes! Only few random links can be used to achieve it.

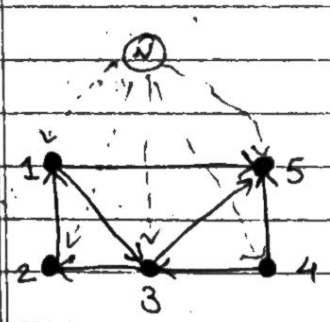
Preferential Attachment Model (Watts - Strogatz Model)

main assumption :- when a new user joins the network, the probability of connecting to existing nodes is proportional to existing nodes degrees.

— For the new node v, connect v to a random node v_i with probability

$$P(v_i) = \frac{d_i}{\sum_j d_j}$$

Eg:



$P(v_1) = 1/7$

$P(v_2) = 1/7$

$P(v_3) = 2/7$

$P(v_4) = 0$

$P(v_5) = 3/7$

DATE : / /
PAGE NO. :

Limitations of Gnp :

① Its degree distribution differs from that of real world networks, which usually follow power law.

② The clustering coefficients of the Erdos Renyi networks are too small.

③ Most importantly, real world networks are not generated randomly.

Limitations of small world models :

① It still has unrealistic degree distribution which does not follow power-law.

② Fixed number of nodes, unlike real network number of nodes continually grows.

③ Random models generate only undirected networks, while many real-world networks are directed.

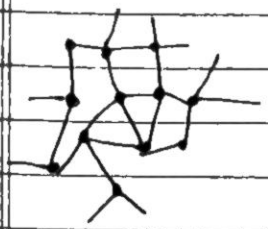
Small-world networks are designed to generate networks that are similar to real world networks to provide a mathematical explanation of real world networks.

The Erdos-Renyi Model and the Watts-Strogatz model are generated randomly.

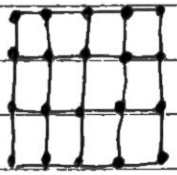
DATE : / /
PAGE NO.

Why is graph deep learning hard?

- Arbitrary size and complex topological structure i.e no spatial locality like grids.



networks



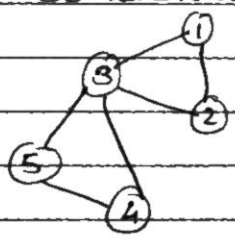
Images

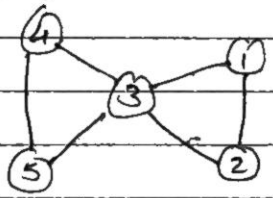
text

- No fixed node ordering or reference point.
- Often dynamic and have multimodal features.

Why is a graph difficult to analyze?

- A graph does not exist in a euclidean space, i.e it cannot be represented by any coordinate systems. unlike other types of data such as waves, images, time-series signals ("text" is also treated as time-series) which can be mapped to a 2D or 3D euclidean space.
- A graph does not have a fixed form. Both graphs are visually diff & has diff structures but has same adjacency matrix. So whether these graphs are considered same or diff?





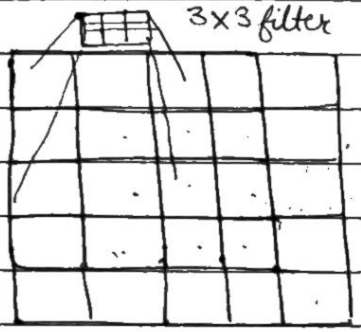
- Giant graphs with very high dimensions and densely grouped nodes are even difficult for a human to understand the graph. Therefore, it is challenging to train a machine for this task. eg:- graph of internet.

DATE : / /
PAGE NO. :

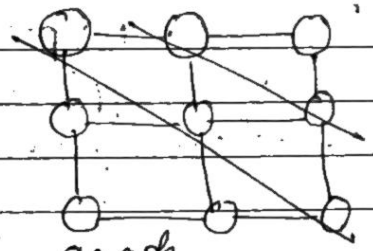
· GNN (Graph neural networks)

GNNs are a class of deep learning methods designed to perform inference on data described by graphs.
GNN's are neural networks that can be directly applied to graphs and provide an easy way to do node-level, edge-level, and graph-level prediction tasks. GNNs can do what CNNs failed to do.

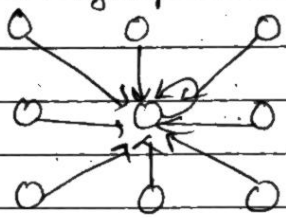
Why Convolution neural networks (CNN.s) fail on graphs?
-> CNNs can be used to make machines visualize things, and perform tasks like image classification or recognition, or object detection.



Image



graph



Transform information at the neighbors and combine it:
transform messages h_i from neighbors $w_i h_i$
: add them up $\sum_i w_i h_i$.
In convolution, center node of the center pixel aggregates information from its neighbors, as well as from itself, to produce a new value. Its very difficult to perform CNN on graphs because of the arbitrary size of the graph, and the complex topology, which means there is no spatial locality.

DATE : / /
PAGE NO. :

Secondly, there's also unfixed node ordering. If we first labeled the nodes A, B, C, D, E and the second time we ~~table~~ labeled them B, D, A, E, C then the inputs of the matrix will change in the network will change. Graphs are invariant to node ordering, so we want to get the same result regardless of how we order the nodes.

GNN comes in many forms, but should always be either invariant (permutation of the nodes of the input graph does not affect the output) or equivariant (permutation of the input permutes the output).

(a)
→ O(N) parameters.
→ Not applicable to graph of diff sizes → for every new graph size, new model needs to be created.
→ Not invariant to node orderings.
↓
for a graph with n nodes, no of ~~possible~~ graphs possible by changing the graph labels are n!. And the model gives n! diff output for the same graph. Therefore we do not use the naive approach.

The intuition of GNN is that nodes are naturally defined by their neighbors and connections. Having this in mind, we then give every node a state (x) to represent its concept. We can use the node state (x) to produce an output (o) i.e decision about the concept. The final state (x_n) of the node is normally called "node embedding". The task of all GNN is to determine the "node embedding" of each node, by

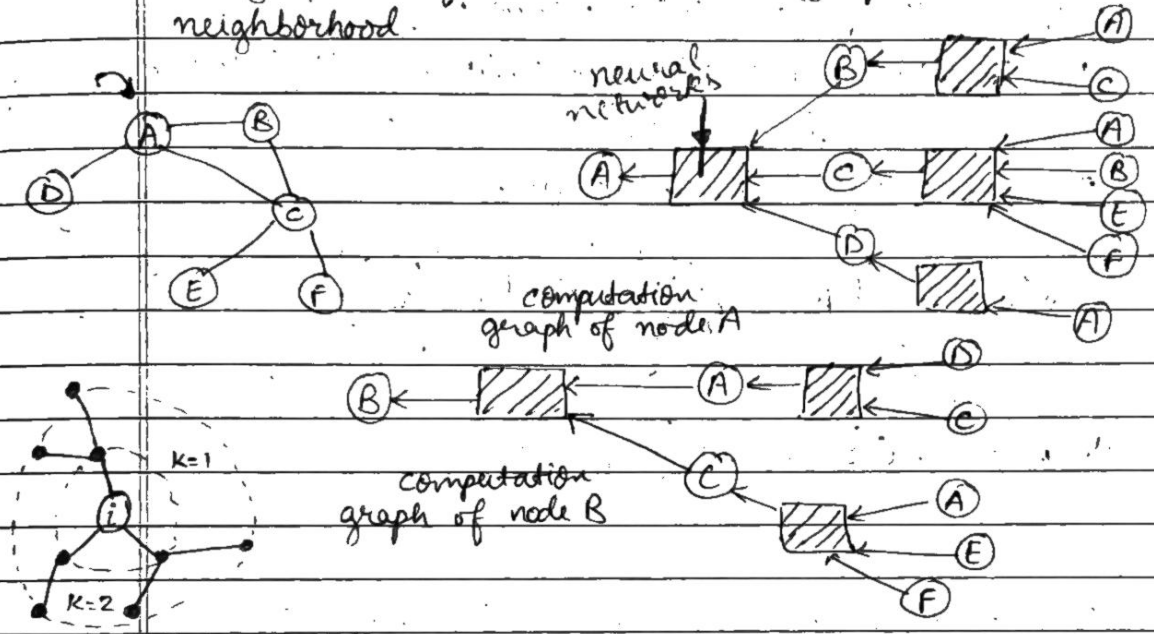
① Locality (local network neighborhoods)
② Aggregate information
③ Stacking multiple layers (computation)

DATE : / /
PAGE NO. :

looking at the information on its neighbouring nodes.

Ref In graph theory, we implement the concept of Node Embedding. It means mapping nodes to a d-dimensional embedding space (low dimensional space rather than the actual dimension of the graph), so that similar nodes in the graph are embedded close to each other.

- Generate node embeddings based on local network neighborhoods.
- Nodes aggregate information from their neighbors using neural networks.
- Every node defines a computation graph based on its neighborhood.



Locality information can be achieved by using a computational graph. In the graph 'i' is the node where we see how this node is connected to its neighbors and those neighbors neighbors. After seeing all the connections we form a computation graph for every node and those will be used for embedding. Note: By doing this we are capturing

DATE: / /
PAGE NO.

the structure, and also borrowing feature information at the same time

Once the locality information preserves the computation graph, we start aggregating. This is done using neural networks. Note: The neural network boxes in the diagram requires aggregation to be order invariant, like sum, avg, max, min because they are permutation-invariant functions.

In GCN we use avg as aggregate func^n.

Let's see the forward propagation rule in GNNs. It determines how the info from input will go to the output side of the neural network.

1. Initialize the activation units // Initial 0-th layer embeddings are equal to node features.

$$h_v^0 = x_v \text{ (feature vector)}$$

2. Every layer in the network

$$h_v^k = \sigma\left(W_k \sum_{u \in N(v)} \frac{h_u^{k-1}}{|N(v)|} + B_k h_v^{k-1}\right)$$

σ: non-linearity (e.g. ReLU)

$\sum_{u \in N(v)} \frac{h_u^{k-1}}{|N(v)|}$: avg of neighbors previous layer embeddings

$B_k h_v^{k-1}$: self loop.

h_v^{k-1}: prev layer embedding of v.

$\forall$ where $k \in \{1, \ldots, K\}$

3. The last equation (at the final layer):

$$z_v = h_v^K$$

Its the embedding after 'K' layers of neighborhood aggregation.

DATE / /
PAGE NO

We can feed these embeddings into any loss function and using stochastic gradient descent to train the weight parameters.

Rewritten in vector form

$$H^{(l+1)} = \sigma\left(H^{l} W_0^{l} + \bar{A} H^{l} W_1^{l} \right)$$

learnable parameters — weights
sharable parameters — K

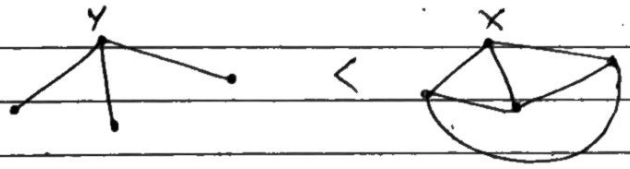
(?) Why we are not exceeding from K=2
① increases computational power
② diameter (due to small world problem).

One of the imp characteristic of embedding is dimension reduction of output.

↳ Network-X
create graph

↳ stellar graph (STC)
↳ automatically graph is created.

#.



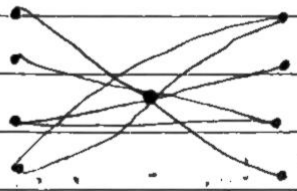
node with more links are more imp.

If we compare X is relatively more important node.
Improvements:- How to find these important nodes in a graph:
① clustering coefficient can be used to select nodes rather than randomly choosing nodes.

DATE : / /
PAGE NO

② Closeness centrality - how far the node is from other nodes.

③ Between the centrality



GCN application :- • Traffic used by (OLA, google MAPS)
→ Creating new antibiotics in field of medical science.

Node Centrality (Prioritizing nodes / finding imp nodes)

① Degree Centrality
High degree, high priority to the node

② Closeness Centrality

$$C_{close}(x) = \frac{1}{\sum_y d(y,x)}$$

(d → distance)

① ② ③ ④ ⑤

$$C_{close}(1) = \frac{1}{d(1,1)+d(2,1)+d(3,1)+d(4,1)+d(5,1)}$$

$$= \frac{1}{0+1+2+3+4} = \frac{1}{10} = 0.1$$

$$C_{close}(3) = \frac{1}{d(1,3)+d(2,3)+d(4,3)+d(5,3)} = \frac{1}{2+1+1+2} = \frac{1}{6}$$

$$= 0.16$$

High the Cclose, high priority to the node.

DATE : / /
PAGE NO.

(3) Betweeness Centrality

$$C_{bet}(x) = \sum_{y, z \neq x,\ \sigma_{yz} \neq 0} \frac{\sigma_{yz}(x)}{\sigma_{yz}}$$

x should not be src or dst

there is at least one path b/w yz

σ_{yz} = total path from $y \rightarrow z$

$\sigma_{yz}(x)$ = total path passing through x

note: src node and dst node have total zero paths passing through them.

High C_{bet}, High priority to node.

(4) Rank Centrality

application: Google uses Rank Centrality for web surfing i.e ranking the websites.

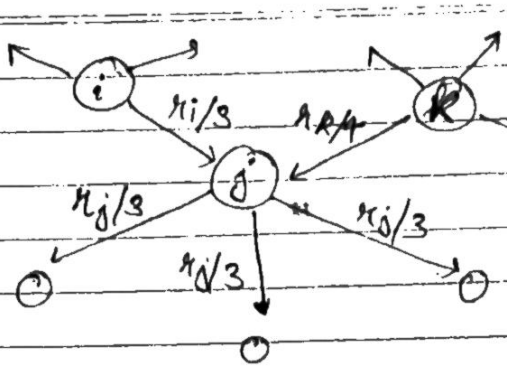
→ PageRank (aka the google algorithm)

Ranking nodes on the graph.

PageRank: The "Flow" model

- Each link's vote (consider link as vote) is proportional to the importance of its source page.
- if page i with importance r_i has d_i out links, each link gets r_i/d_i votes.
- Page j's own importance r_j is the sum of the votes on its in-links.

DATE : / /
PAGE NO.



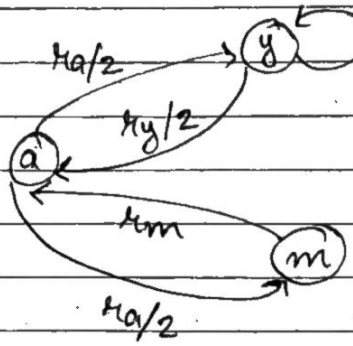
$$r_j = \frac{r_i}{3} + \frac{r_k}{4}$$

$$r_j = \sum_{i \to j} \frac{r_i}{d_i}$$

d_i → out degree of i

A page is important if it is pointed by other important pages.

eg:



$$r_y = \frac{r_y}{2} + \frac{r_a}{2} \; ; \; r_a = \frac{r_y}{2} + r_m \; ; \; r_m = \frac{r_a}{2}$$

$$2r_m = \frac{r_y}{2} + r_m \Rightarrow \frac{r_y}{2} = r_m \Rightarrow r_y = 2r_m$$

$$2r_m = \frac{2r_m}{2} + \frac{2r_m}{2}$$

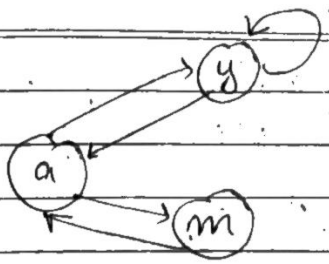
$r_a = 2r_m$; $r_y = 2r_m$; r_m

Matrix ~~representation~~ formulation

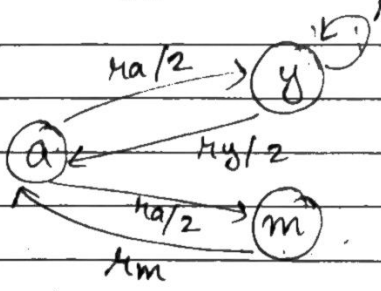
$$\boxed{R = M \cdot R}$$

DATE : / /
PAGE NO. :

eg:



=>



$r_y = \frac{r_y}{2} + \frac{r_a}{2}$

$r_a = \frac{r_y}{2} + r_m$

$r_m = \frac{r_a}{2}$

	r_y	r_a	r_m
r_y	1/2	1/2	0
r_a	1/2	0	1
r_m	0	1/2	0

$$r = M \cdot r$$

$$\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 1 \\ 0 & 1/2 & 0 \end{bmatrix} \begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix}$$

Random walk interpretation :-

Given a web graph with N nodes, where the nodes are pages and edges are hyperlinks. Imagine a web surfer:

- At any time 't', surfer is on some page 'i'.
- At time 't+1', the surfer follows an out-link from 'i' uniformly at random
- Ends up on some page 'j' linked from 'i'
- Process repeats indefinitely.

DATE / /
PAGE NO

let $P(t)$ vector whose i^{th} coordinate is the probability that the surfer is at page 'i' at time 't'. So $p(t)$ is probability distribution over pages.
and at time $t+1$ i.e follow a link at random

$$p(t+1) = M \cdot p(t)$$

we stop when

$$p(t+1) = p(t)$$

random walk reaches a stationary state ie $p(t)$ is stationary distribution of a random walk.

Power Iteration Method.

(1) Initialize: $r^0 = [1/N, \ldots 1/N]^T$
(2) Iterate: $r^{(t+1)} = M \cdot r^{(t)}$
(3) Stop when $|r^{(t+1)} - r^{(t)}| < \varepsilon$

Compare $Ax = \lambda x$ with $r = M \cdot r$

x is an eigen vector with the corresponding eigenvalue λ

$M \cdot r = 1 \cdot r$

eigen value = 1

& r is the corresponding eigen vector for eigenvalue 1 in matrix M.

eg: Step1: $\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}$ Initialization

Step2: $r^{(t+1)} = M \cdot r^{(t)}$

$$= \underset{3\times3}{\begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 1 \\ 0 & 1/2 & 0 \end{bmatrix}} \underset{3\times1}{\begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}} = \begin{bmatrix} 1/6+1/6 \\ 1/6+1/3 \\ 1/6 \end{bmatrix} = \begin{bmatrix} 1/3 \\ 1/2 \\ 1/6 \end{bmatrix}$$

DATE : / /
PAGE NO. :

$$r^{(t+1)} = M \cdot r^{(t)}$$

$$= \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 1 \\ 0 & 1/2 & 0 \end{bmatrix} \begin{bmatrix} 1/3 \\ 1/2 \\ 1/6 \end{bmatrix} = \begin{bmatrix} 1/6+1/4 \\ 1/6+1/6 \\ 1/4 \end{bmatrix} = \begin{bmatrix} 5/12 \\ 1/3 \\ 1/4 \end{bmatrix}$$

$$r^{(t+1)} = M \cdot r^{(t)}$$

$$= \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 1 \\ 0 & 1/2 & 0 \end{bmatrix} \begin{bmatrix} 5/12 \\ 1/3 \\ 1/4 \end{bmatrix} = \begin{bmatrix} 5/24 + 1/6 \\ 5/24 + 1/4 \\ 1/6 \end{bmatrix} = \begin{bmatrix} 9/24 \\ 11/24 \\ 1/6 \end{bmatrix}$$

$$r^{(t+1)} = M \cdot r^{(t)}$$

$$= \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 1 \\ 0 & 1/2 & 0 \end{bmatrix} \begin{bmatrix} 9/24 \\ 11/24 \\ 1/6 \end{bmatrix} = \begin{bmatrix} 9/48 + 11/48 \\ 9/48 + 11/24 \\ 11/48 \end{bmatrix} = \begin{bmatrix} 5/12 \\ 31/48 \\ 11/48 \end{bmatrix}$$

⋮

Step3: Stop when $r^{(t+1)} = r^{(t)}$ ans. $\begin{bmatrix} 2/5 \\ 2/5 \\ 1/5 \end{bmatrix}$

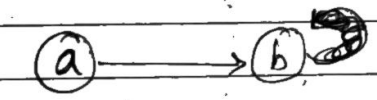
$$r^{(t+1)} = M \cdot r^{(t)}$$

$$= \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 1 \\ 0 & 1/2 & 0 \end{bmatrix} \begin{bmatrix} 2/5 \\ 2/5 \\ 1/5 \end{bmatrix} = \begin{bmatrix} 2/5 \\ 2/5 \\ 1/5 \end{bmatrix}$$

$r^{(t+1)} \approx r^{(t)}$

Issues/Problems in PageRank :-

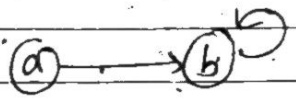
(i) Some pages are dead ends (have no out links). Some pages cause importance to "leak out"



all importance will be given to 'b' now i.e. 'b' become the imp node.

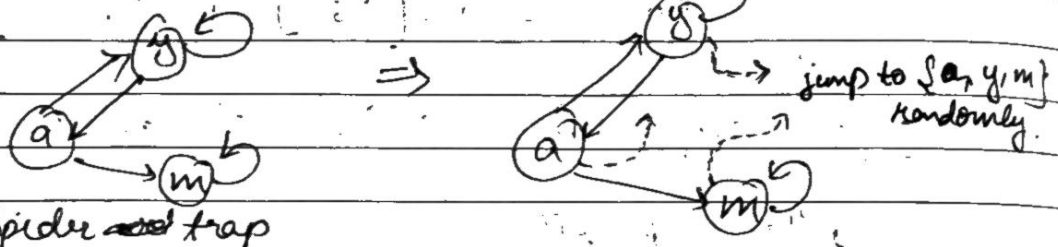
DATE : / /
PAGE NO. :

(2) Spider traps - (all out links are within the group)
eventually spider traps absorb all importance.

eg: 

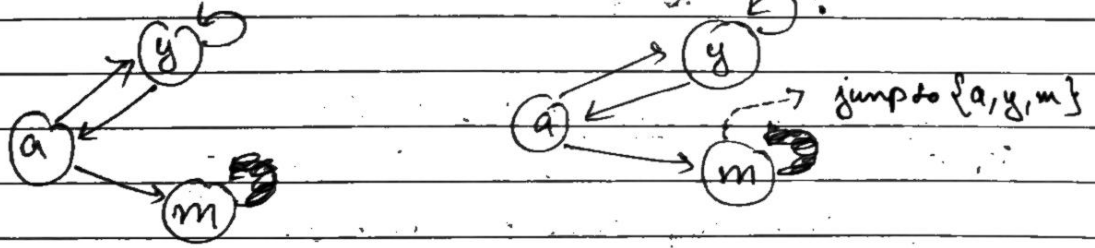
sol^n of spider traps : At each time step, the random web surfer has two options:
- with probability β, follow a link at random
- with probability $1-\beta$, jump to a random page
- Common values of β are in the range 0.8 to 0.9

Surfer will teleport out of spider trap within few time steps.



Spider trap

sol^n to dead ends is teleport i.e follow random teleport links with total probability 1 from dead ends.



	y	a	m
y	1/2	1/2	0
a	1/2	0	0
m	0	1/2	0

	y	a	m
y	1/2	1/2	1/3
a	1/2	0	1/3
m	0	1/2	1/3

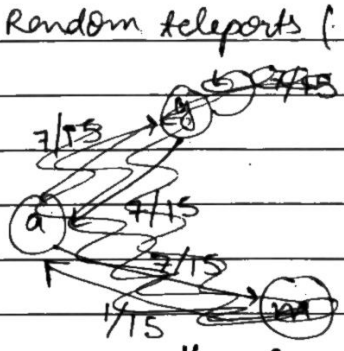
DATE : / /
PAGE NO. :

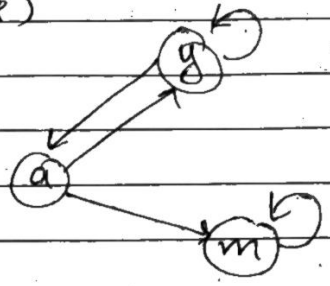
soln: Random Teleports. (google's solution)
~~with probability~~

Page Rank equation

$$r_j = \sum_{i \to j} \beta \frac{r_i}{d_i} + (1-\beta)\frac{1}{N}$$

eg: Random teleports ($\beta = 0.8$)





Imp:
Receiving/input ends. / links

$$\Rightarrow M = \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \\ 0 & 1/2 & 1 \end{bmatrix} \begin{matrix} y \\ a \\ m \end{matrix}$$

$$A = \beta \cdot M + (1-\beta)\left[1/N\right]_{N \times N}$$

$r = A \cdot r$
apply power method.

$$A = 0.8 \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \\ 0 & 1/2 & 1 \end{bmatrix} + \begin{bmatrix} 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{bmatrix} (0.2)$$

$$= \begin{bmatrix} 7/15 & 7/15 & 1/15 \\ 7/15 & 1/15 & 1/15 \\ 1/15 & 7/15 & 13/15 \end{bmatrix}$$

Node Proximity / Personalized PageRank / Random Walk with Restarts.

DATE / /

PAGE NO

PageRank

① "Normal" PageRank

- Teleports uniformly at random to any node
- All node have the same probability of surfer landing there.

$S = [0.1, 0.1, 0.1, 0.1, 0.1, 0.1]$

② Topic-Specific PageRank also known as personalized PageRank.

- Teleports to a topic specific set of pages.
- Nodes can have different probabilities of surfer landing there.

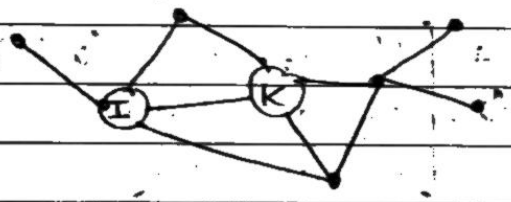
$S = [0.1, 0, 0, 0.2, 0, 0, 0.5]$

③ Random Walk with Restarts.

- Topic-Specific PageRank where teleport is always to the same node.

$S = [0, 0, 0, 0, 1, 0, 0, 0]$

Q. Which conference are closest to KDD & ICDM?



=> Personalized PageRank with teleport set

$S = \{KDD, ICDM\}$

Pinterest Pin Recomendations

uses ~~Page Rank~~ Random walk with Restarts

DATE: / /

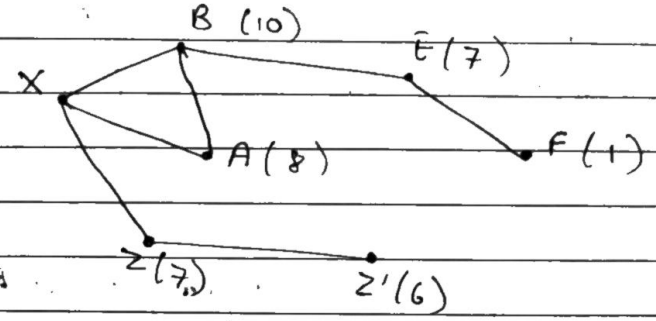
PAGE NO.:

Random walk eg.

eg: $\alpha/P = 0.75$ Query Nodes $\{X\}$

H → move forward

T — Back to the src & restart



If you are at website X and it links to other 3 website B, A, Z then what are the chances you will visit website B. For this recomendation we apply randomwalks idea and this count will tell you that which is the closest node to node 'X' or which is the closest website to website X.

Pixie Random Walk algorithm :-

1. Make a step to a random neighbor and record the visit (visit count)
2. with probability ALPHA, restart the walk at one of Query-nodes.
3. The nodes with the highest visit count have highest proximity to the Query-nodes.

Pixie is a flexible, graph based system for making personalized recomendations in real time

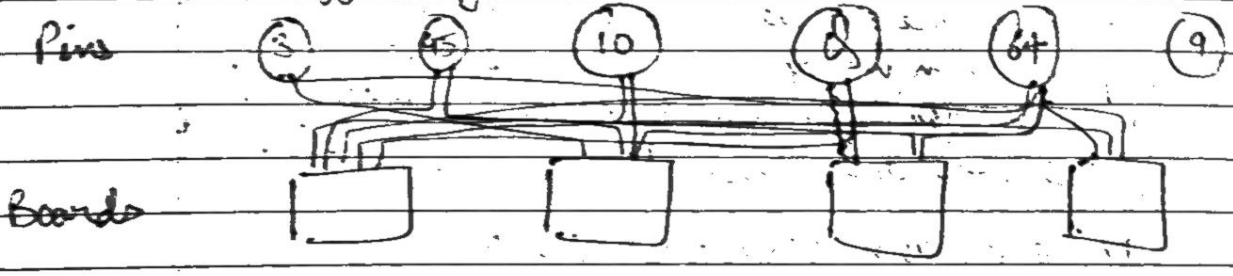
So, Basically in Random walk we start with the Query-nodes, the query-nodes either consist of a single node or multiple nodes. If multiple then we assign weights to all the nodes. Then we start the random walk. Starting from the node X (query node) we either move forward with probability 'α' or go back to the querynode to restart with probability (1-α). And for each time we encounter a

DATE :
PAGE NO.

node we increment visit_count by 1 which was initialized to 0.

In case of Pinterest recommendation it becomes the two way process. of moving from a pin → Board and Board → Pin. And increating visit count for every pin.

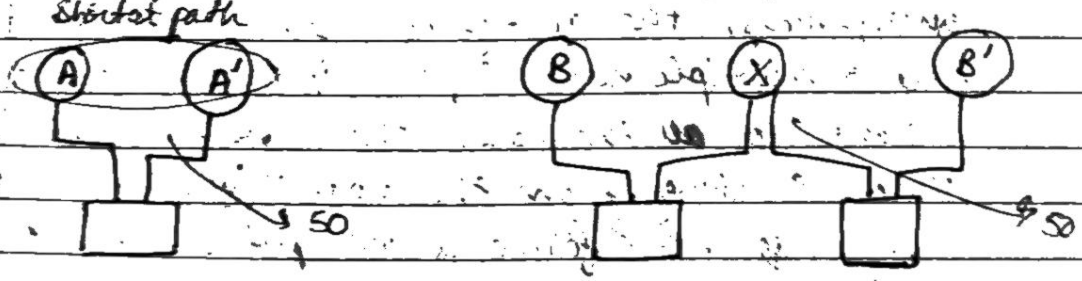
This process of random walk is done more than 100 00 times to increas the efficiency.



We can say that the most related pin to pin 8 is the one with the highest visit count i.e 64 in this example.

Random walk gives the most closest node to node 8 in the graph.

which is more similar or more related pins



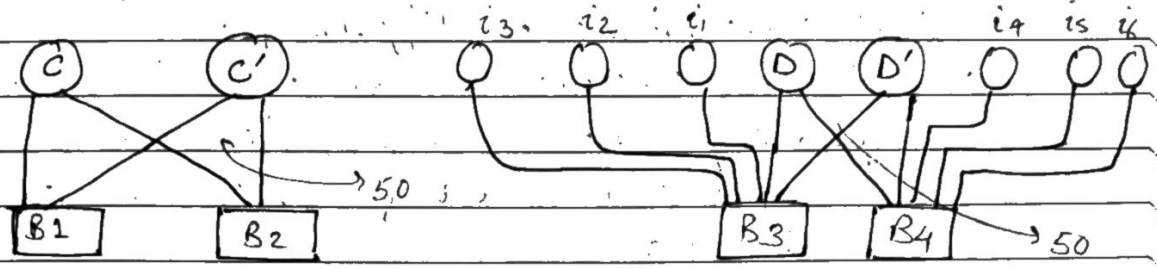
⇒ AεA' is more similar than BB' because its

DATE : / /
PAGE NO. :

Say both the patterns are repeating for 50 times in the graph. So, if observed when a board is saving both A & A' and not any other pins where as one board is saving pins B & X and another board X & B' and we can say that pins B and B' are in a way related to pin X. But A & A' are directly related.

∴ AA' is more similar than B & B'.

which is more similar CC' or DD'?



⇒ CC' is more similar than DD' because consider that both the patterns are repeating 50 times in the network. We see that B_1 and B_2 people are saving pins C & C' together whereas B_3 and B_4 people are saving pins D & D' with other extra pins. It can bee seen as B_3 & B_4 people are purchasing other items with items D & D' and B_1 & B_2 people are purchasing only C & C' items with more clarity.

∴ CC' is more similar than DD'.

Ref:

The Pixie random walk algorithm is easy but there are many other features hidden in it.

(1) Degree - A particular board has 1000 pins then this algorithm make sure that its importance is divided along the 1000 edges and will finally select only one of them. Variation in degree distribution is given importance and does not hampere the recomendation.

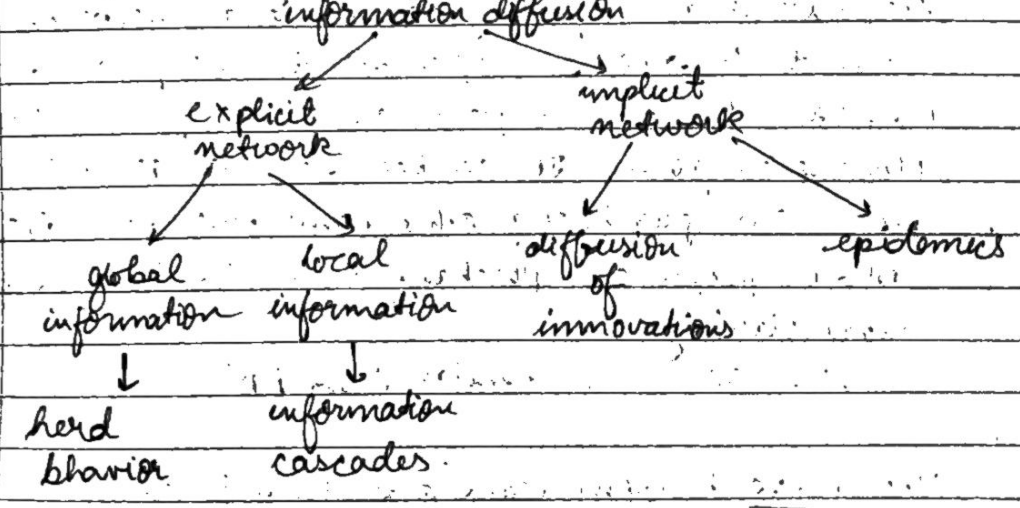
- ~~Reduce~~ Stop the random walk after 20 counts is achi...
- pre processing.

DATE: / /
PAGE NO

2nd/(2) Shortest path pens are more likely to be similar.

Information Diffusion :- process by which a piece of information (knowledge) is spread and reaches individuals through interactions.

- Sender (s) - A sender or a small set of senders that initiate the information diffusion process.
- receiver(s) - A receiver or a set of receivers that receive diffused information. Commonly, the set of receivers is much larger than the set of senders and can overlap with the set of senders
- medium - this is the medium through which the diffusion takes place

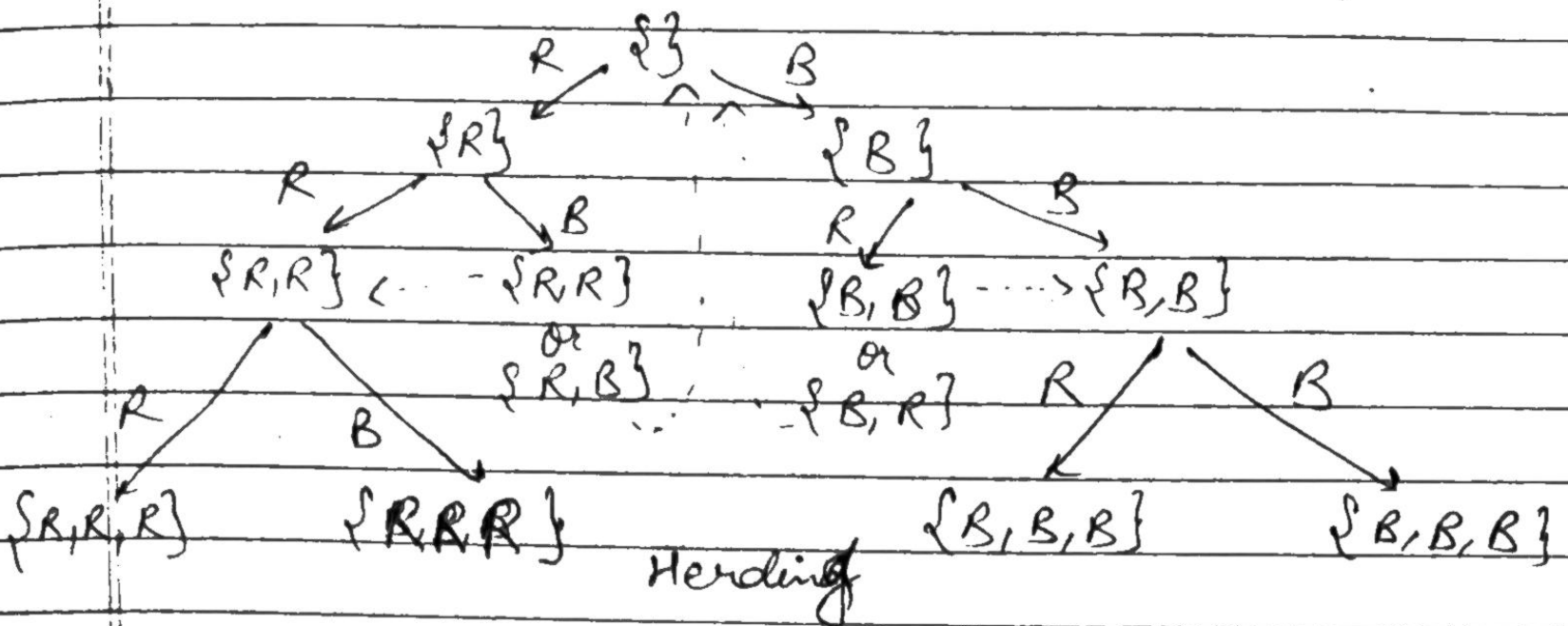


We define the process of interfering with information diffusion by expediting, delaying, or even stopping diffusion as intervention.

DATE : / /

PAGE NO. :

In Herd Behavior, individuals make decisions by observing all other individuals decisions. In general, herd behavior's network is close to a complete graph where nodes can observe at least most other nodes and they can observe public information.



GAT. RNN