Part Of Recommend System

$\boxed{\text{Graph Matrix、PageRank、RandomWalks、Embeddings}}$

Google Algorithm: PageRank (Directed Graph)

Web → Graph

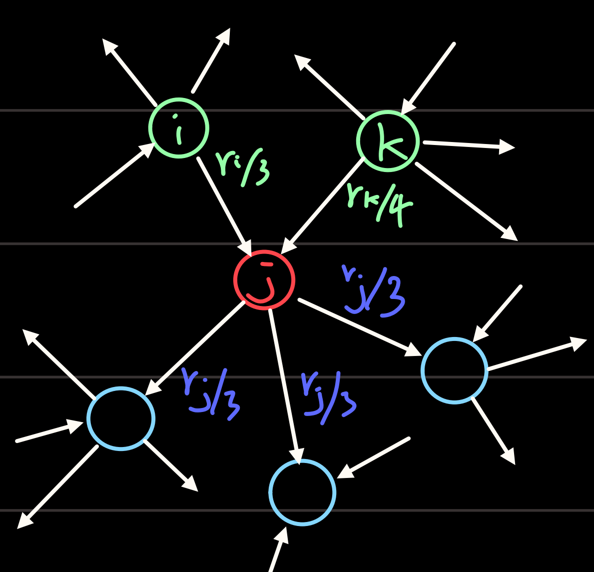
Pages → Nodes

Links(Edges) → Hyperlinks

Why Rank? → Web Pages are not equally important

Important pages have more links ⟺ A vote from important page is worth more

Link As Node: Flow Model



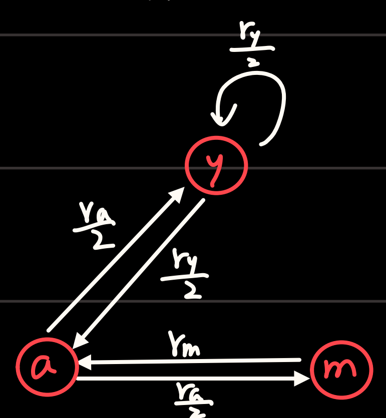
If page i with importance r_i has d_i ($=3$) out-links, each link get $\frac{r_i}{d_i}$ votes

Page j's own importance r_j is the sum of the votes on its in-links : $r_j = \frac{r_i}{3} + \frac{r_k}{4}$

Out-links → votes and In-links → importance

Def rank r_j for node j

$$r_j = \sum_{i \to j \text{ in}} \frac{r_i}{d_i}$$
, where d_i is out-degree of node i

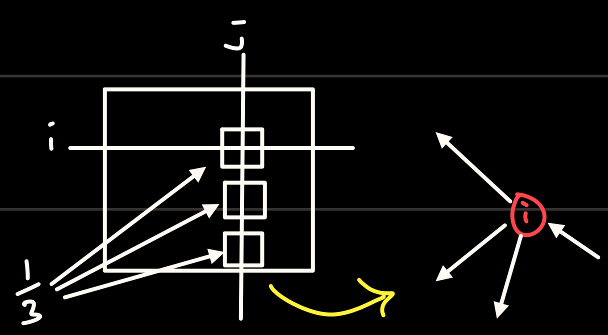


$$r_a = r_m + \frac{r_y}{2}$$
$$r_y = \frac{r_a}{2} + \frac{r_y}{2}$$
$$r_m = \frac{r_a}{2}$$

(Don't use Gaussian elimination to solve this system)

Stochastic Adjacency Matrix M (similar to transition matrix)

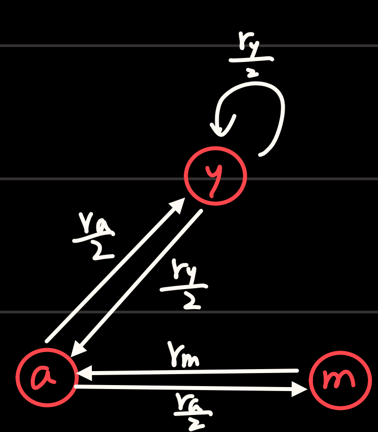
if $j \to i$ then $M_{ij} = \frac{1}{d_j}$, $\sum_i M_{ij} = 1$



→ only link-out index non-zero

Rank vector r : An entry per page, r_i is rank of page i where $\sum_i r_i = 1$

Flow Equation : $r = M \cdot r \longleftrightarrow r_j = \sum_{i \to j} \frac{r_i}{d_i}$



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

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

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

$$M = \begin{array}{c} \\ r_y \\ r_a \\ r_m \end{array} \begin{array}{c} \begin{array}{ccc} r_y & r_a & r_m \end{array} \\ \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & 1 \\ 0 & \frac{1}{2} & 0 \end{bmatrix} \end{array}$$

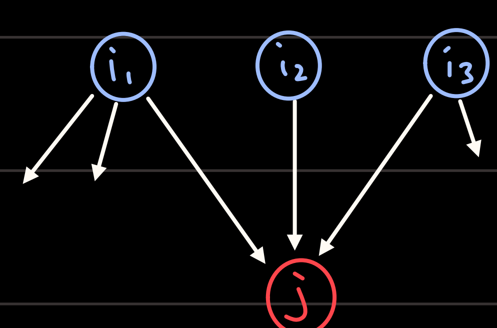
$$r = M \cdot r \to \begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & 1 \\ 0 & \frac{1}{2} & 0 \end{bmatrix} \begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix}$$

Def Connect to Random Walk : Random Web Surfer

{At any time t, surfer is on some page i

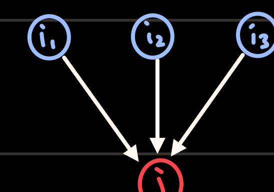
time $t+1$: the surfer follows an out-link from i uniformly at random

Ends up on some page j linked from i} Process repeats indefinitely



$p(t)$: a vector whose i^{th} coordinate is the prob that the surfer is at page i at time t

→ A probability distribution over pages



The Stationary distribution :

Follow a link uniformly at random : $p(t+1) = M\, p(t)$

Random Walk ⇒ If $p(t+1) = M \cdot p(t) = p(t) \rightarrow p(t)$ is a stationary distribution of a random walk

r satisfies $r = M \cdot r \rightarrow r$ is stationary distribution for the random walk

(Eigenvalue Centrality: $\lambda c = Ac$, where A is Adjacency matrix)

$$1 \cdot r = M \cdot r \rightarrow 1 \cdot \begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & 1 \\ 0 & \frac{1}{2} & 0 \end{bmatrix} \begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix}$$

Eigenvalue of M : 1

Eigenvector of M : $\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix}$

long-term distribution of surfer : $r = MM \ldots\ldots Mu \Rightarrow$ satisfies $1 \cdot r = M \cdot r$

$\rightarrow$ PageRank = Limiting distribution = principle eigenvector of M with eigenvalue (=1)

(asympotic)

$\parallel$

the stationary distribution of a random walk over the graph

Solve PageRank

Given $G(V, E)$, with n nodes

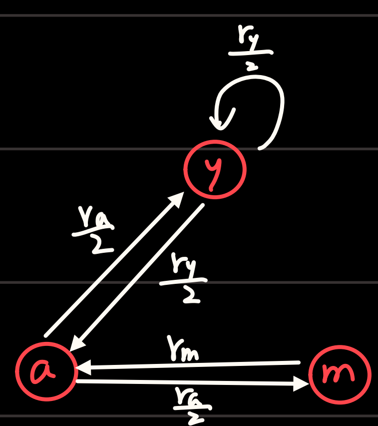
Init each node an initial pagerank

Repeat until achieve converge condition$(\sum_i |r_i^{t+1} - r_i^{t}| < \varepsilon)$ $\{ r_j^{(t+1)} = \sum_{i \to j} \frac{r_i^{(t)}}{d_i} \ \forall j \}$

Def Power Iteration (for finding Eigenvalues)

Init $r^{(0)} = \begin{pmatrix} \frac{1}{N} \\ \vdots \\ \frac{1}{N} \end{pmatrix}$

Repeat until $|r^{(t+1)} - r^{(t)}|_1 < \varepsilon$ $\{ r^{(t+1)} = M r^{(t)} \}$



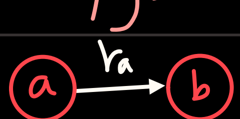
$$r_a = r_m + \frac{r_y}{2}$$
$$r_y = \frac{r_a}{2} + \frac{r_y}{2}$$
$$r_m = \frac{r_a}{2}$$

$$M = \begin{array}{c c} & \begin{array}{ccc} r_y & r_a & r_m \end{array} \\ \begin{array}{c} r_y \\ r_a \\ r_m \end{array} & \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & 1 \\ 0 & \frac{1}{2} & 0 \end{bmatrix} \end{array}$$

$$\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \overset{r^0}{\begin{bmatrix} \frac{1}{3} \\ \frac{1}{3} \\ \frac{1}{3} \end{bmatrix}}, \overset{r^1}{\begin{bmatrix} \frac{1}{3} \\ \frac{3}{6} \\ \frac{1}{6} \end{bmatrix}} \cdots\cdots \overset{r^T}{\begin{bmatrix} \frac{6}{15} \\ \frac{6}{15} \\ \frac{3}{15} \end{bmatrix}}$$

Problems:

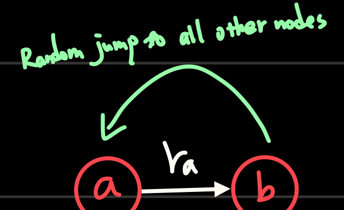
① Some pages are dead ends (no out-links)



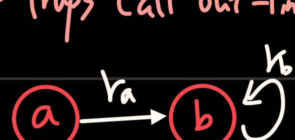
$r_a = 0$ $r_b = r_a$ $M = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}$

$\begin{bmatrix} r_a \\ r_b \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ (Random Walk Stop but $\begin{bmatrix} 0 \\ 0 \end{bmatrix}$)

$\xrightarrow{\text{solution}}$



② spider traps (all out-links are within the group)

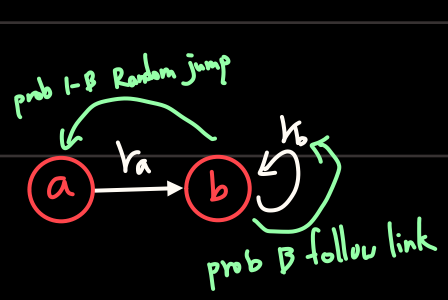


$r_a = 0$ $r_b = r_a + r_b$ $M = \begin{bmatrix} 0 & 0 \\ 1 & 1 \end{bmatrix}$

$\begin{bmatrix} r_a \\ r_b \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix} \cdots\cdots \begin{bmatrix} 0 \\ 1 \end{bmatrix} \cdots\cdots \infty$

(Random Walk keep running on the same node can't escape)

$\xrightarrow{\text{solution}}$



PageRank Equation:

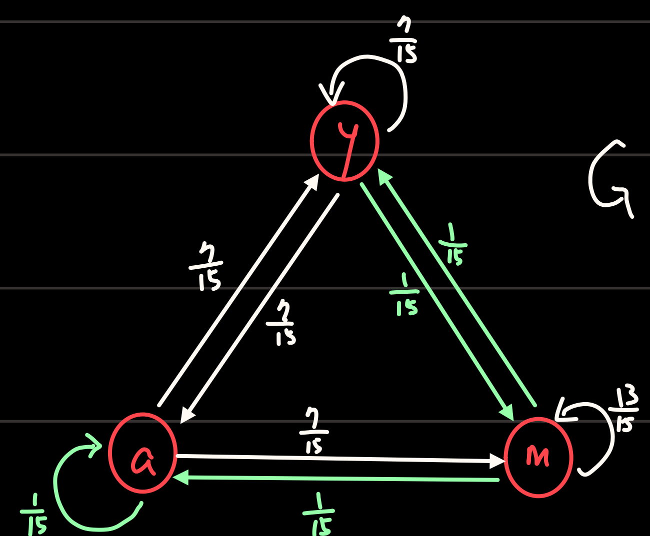
At each step, random surfer has 2 options:

① with prob β follow a link at random

② with prob $1-\beta$, jump to some random page

$r_j = \sum_{i \to j} \beta \frac{r_i}{d_i} + (1-\beta)\frac{1}{N} \quad \forall i,j$, where N is num of nodes in the graph (Expectation of link-in score)

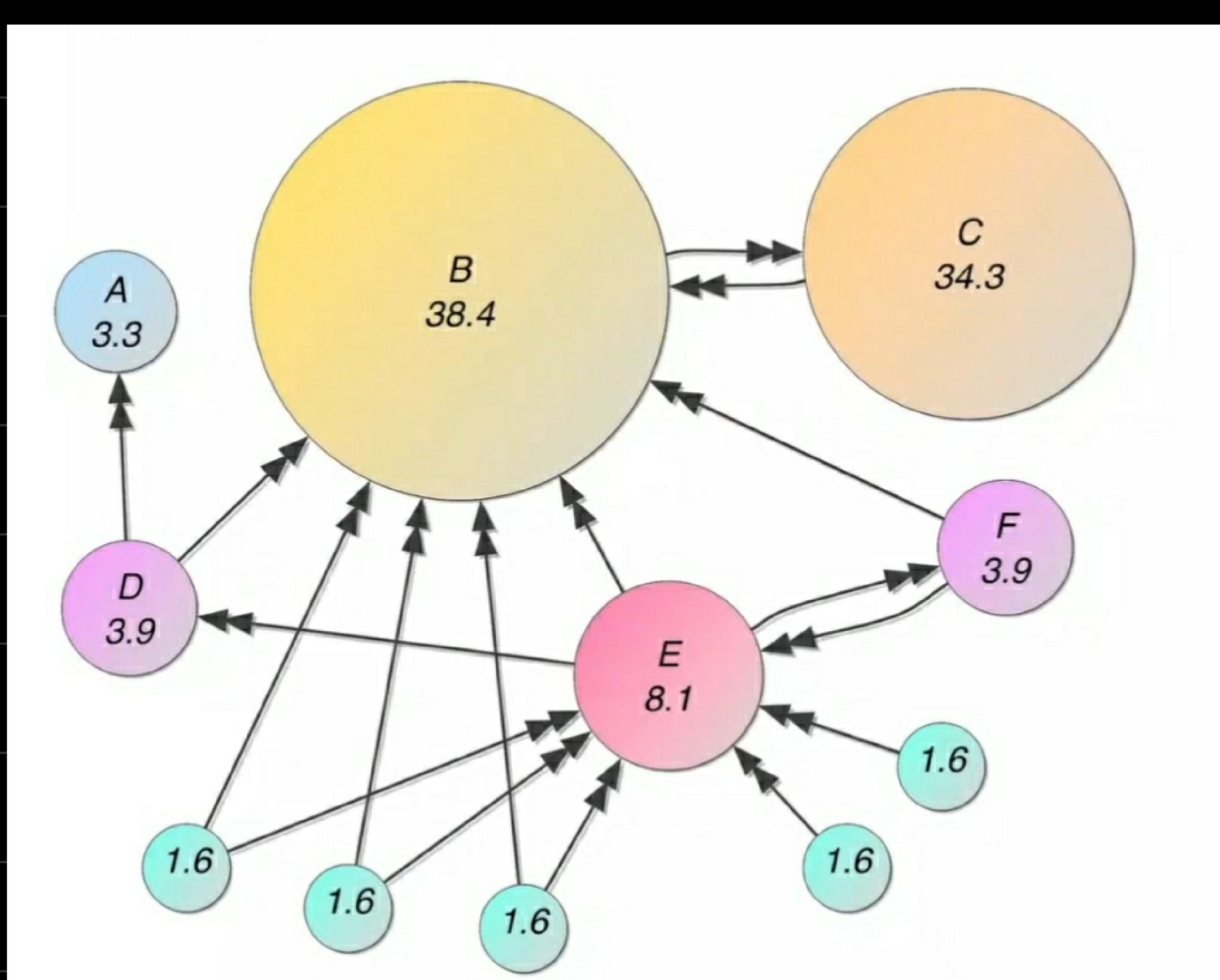
$\Rightarrow 1 \cdot r = G \cdot r$, where $G = \beta M + (1-\beta)\left[\frac{1}{N}\right]_{N \times N}$ (β usually: 0.8, 0.9)



$$G = 0.8\begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{2} & 1 \end{bmatrix} + 0.2\begin{bmatrix} \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \end{bmatrix} = \begin{bmatrix} \frac{7}{15} & \frac{7}{15} & \frac{1}{15} \\ \frac{7}{15} & \frac{1}{15} & \frac{1}{15} \\ \frac{1}{15} & \frac{1}{15} & \frac{13}{15} \end{bmatrix}$$

$$r = \begin{bmatrix} y \\ a \\ m \end{bmatrix} = \begin{bmatrix} \frac{1}{3} \\ \frac{1}{3} \\ \frac{1}{3} \end{bmatrix}, \begin{bmatrix} 0.33 \\ 0.2 \\ 0.46 \end{bmatrix} \cdots\cdots \begin{bmatrix} \frac{7}{33} \\ \frac{5}{33} \\ \frac{21}{33} \end{bmatrix}$$

$$\xrightarrow{\quad} \quad r = G \cdot r$$

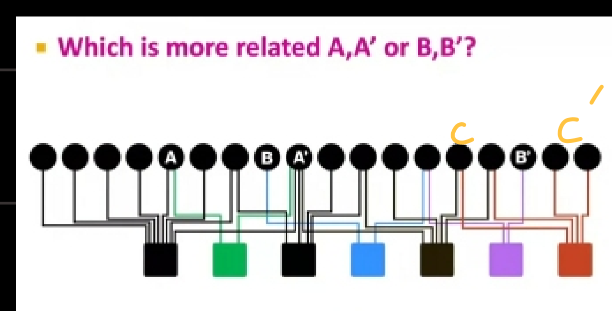


Random Walk with Restarts and Personalized PageRank

Intuition:

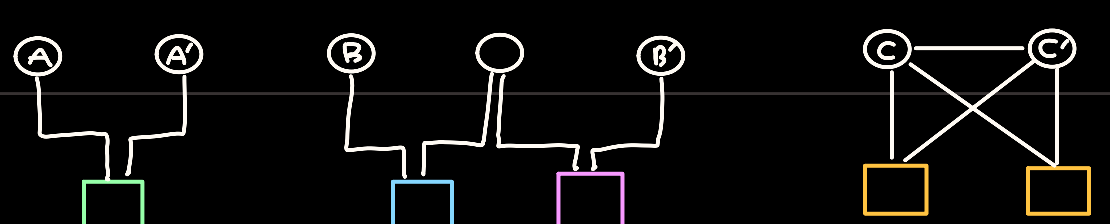
What items should we recommend to a user who interacts with item Q ?

Proximity → If items Q and P are interacted by similar users, recommend P when user interacts with Q



Measurement (Bipartite):

Similarity? How to recommend?



shortest path

Common Neighbors

Def Proximity on graphs

Page Rank: ① Rank nodes by importance
② jump to any node with same prob

↓

Personalized PageRank: Ranks Proximity of nodes to jump nodes S
(Topic specific PageRank)

↓ What is most related to item Q?

Random Walks with Restarts: with prob β jump back to the starting node S={Q}
(Topic specific PageRank)

Random Walk Simulation:

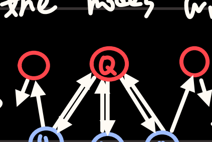
Every node has some importance → importance gets evenly split among all links and pushed to the neighbors

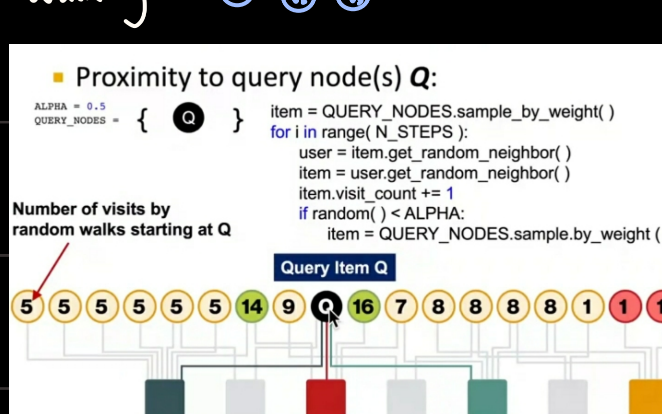
Given {query nodes} simulate a Random Walk:
$Q_1, Q_2 \ldots\ldots$ ① multi queries: personal
② single query: restart

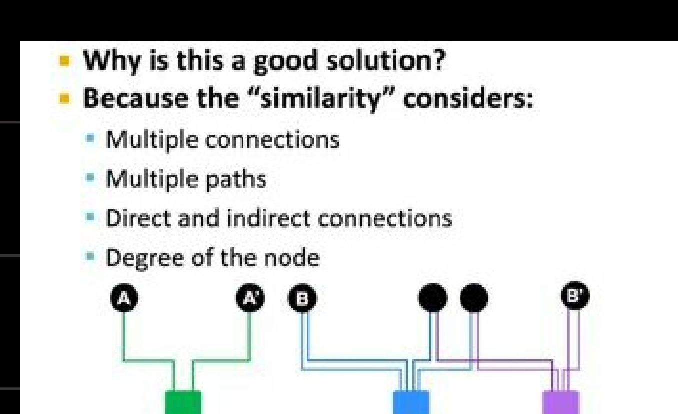
① Make a step to random neighbor and record the visit and visit count

② with prob α → restart the walk at one of the query nodes

③ the nodes with the highest visit count have highest proximity to the query nodes

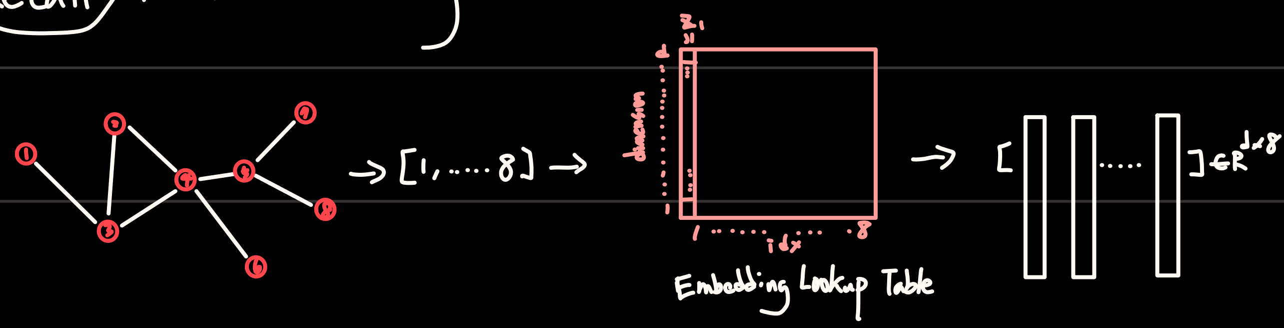
After Walking →  → Power Iteration





Matrix Factorization : Node Embedding

Recall Node embedding



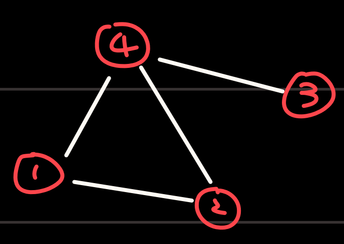
Objective : max $z_v^T z_u$ for node pair (u, v) that are similar

Previous similar : node pairs appear many times when doing Randomwalk ⇒ max $z_u^T z_v$

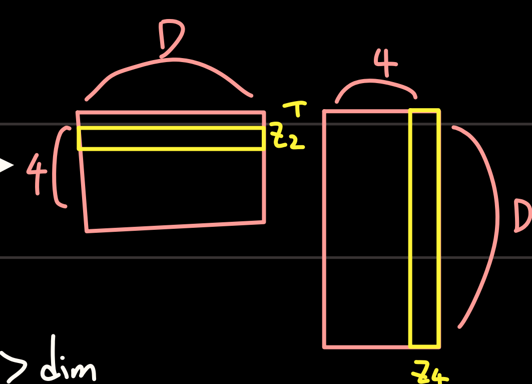
What if : u.v connected $z_v^T z_u = A_{uv} = 1 \longrightarrow z^T z = A$

Def Matrix Factorization

Key : edge connectivity ≈ Embedding dot-product similarity ⇒ if connected : $z_u^T z_v \approx 1$
ow : $z_u^T z_v \approx 0$



$\rightarrow A = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \rightarrow$



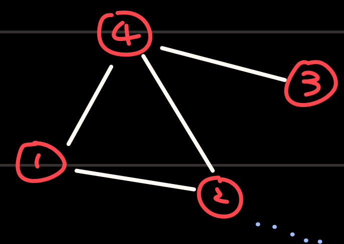
Exact factorize $z^T z = A$ is impossible since #nodes > dim

So we approach : $z^T z \approx A$ by optimization

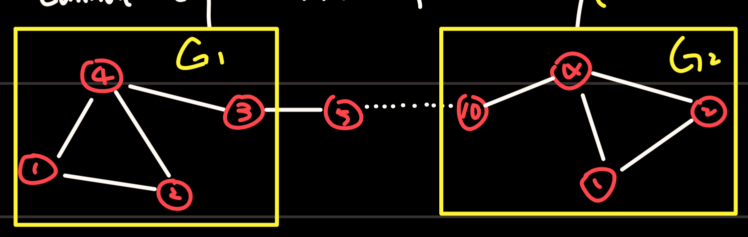
Objective : $\min_z \|A - z^T z\|_F^2$

limitations

① can't compute new node embedding ⇒ need to recompute whole embedding



② cannot capture structural similarity (nodes in G_1, G_2 should be close in embedding space)



③ Can't incorporate node/link/graph level feature

Solution : Deep Representation Learning and Graph Neural Network