HW 1 - numbers

SCC = 60575

IN = 34525

OUT = 3364

Tendrils ~1000 ← num in and number of

Tubes ~ 50

No Gini

Review Centrality

↳ measure of "importance"

degree centrality: degree of a vertex
→ How far is our "reach" within a single hop ↳ influence

closeness centrality: average shortest paths length from a vertex

paths length from a vertex

→ small distance to the rest of the network means higher influence

betweenness centrality: ratio of shortest paths going through some vertex

→ How much information flows through a vertex

eigenvector centrality: How close a vertex is to other important vertices

→ PageRank

Back in the 90s

Good: cool new Internet

Bad: can't find much

Ugly: spam and scams

Search engines (Yahoo, Altavista)

↳ used basic keyword

Issue: spam or untrustworthy

sites could abuse the relatively simple search engines

→ Q: How can we define "trust" on a web graph?

It's impossible for Yahoo, etc. to manually evaluate trust?

However: we can consider

"wisdom of the crowds"

↳ sum or average behavior of a crowd can often carry substantial insight

* In this case, our "crowd" is the rest of the Internet.

* Then "vote" on trustworthiness via

* they "vote" on trustworthiness via hyperlinks to trusted sites

→ spam can still be an issue, but it's limited, as we expect not many links from the global internet graph to spam sites

Google:

Let's evaluate trustworthiness using the above ideas

→ PageRank algorithm to explicitly calcolate trust

PageRank

A spectral centrality measure

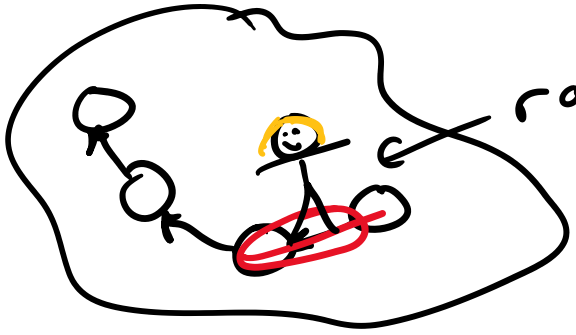
↳ eigenvalues/eigenvectors of the adjacency matrix or variants

Lots of applications
→ search, recommender systems, calculating rankings

=> You're important or trusted if other important or trusted sites link to you

---

1. Random surfer model



big ol' Internet

↓

Infinite random walk on the web graph

$d^-(v)=0$ $d^+(v)=0$

Issue: sources / sinks

↳ we randomly jump to a new page

PageRank of v: probability that if the walk is stopped at some

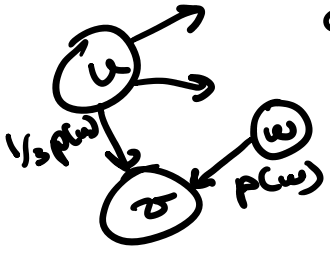
time t, the surfer is on page v

---

2. Vertex-centric graph algorithmic model

$p(v)$ ← pagerank of vertex v

all $p(v)$ initialized to $\frac{1}{|V(G)|}$



update $p(v) = \sum_{u \in N^-(v)} \frac{p(u)}{d^+(u)}$

For sinks: we can manually compute their contributions

OR

add edges from sinks to all other vertices

---

3. Linear Algebraic Model

We have adjacency matrix A

diagonal degree D

↳ $d_{ii} = \sum_j a_{ij}$

↳ $d_{ii} = \sum_j a_{ij}$

$d_{ij} = 0$ if $i \neq j$

Consider transition probability matrix $M^T = (D^{-1}A)$

To actually calculate PR, we'll want the transpose

$$M = (D^{-1}A)^T$$

We can use this matrix to compute PR similar as before

Initialize $P_0 = \begin{Bmatrix} \frac{1}{|V|} \\ \vdots \\ \frac{1}{|V|} \end{Bmatrix}$

Iterate $P_{i+1} = M P_i$

Until $\| P_{i+1} - P_i \| < \epsilon$

Note: $Ax = \lambda x$

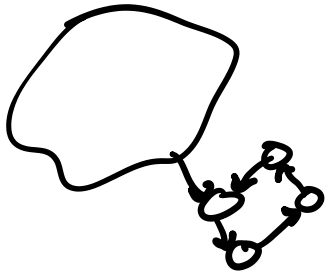
$Mx = \lambda x$

↑ eigenvector ↑ eigenvalue

$p_\infty = M p_\infty$ ← power iteration

↑

PageRank is the eigenvector of the transition prob. matrix with eigenvalue equal to 1



Issue: We might end up in an SCC with no external out edges

Damping Factor: every hop we also will jump with probability p

---

Personalized PageRank

To extend our surfer analogy to "personalize" PR to a vertex v

* Start our surfing at v
* Randomly jump back to v

* Randomly jump back to v

→ gives us a probability distribution for a random surfer from v

Use cases:

Personalized search, recommender systems, link prediction

PageRank for prediction and ranking in competition networks

→ vertices: competitors
edges: competitions

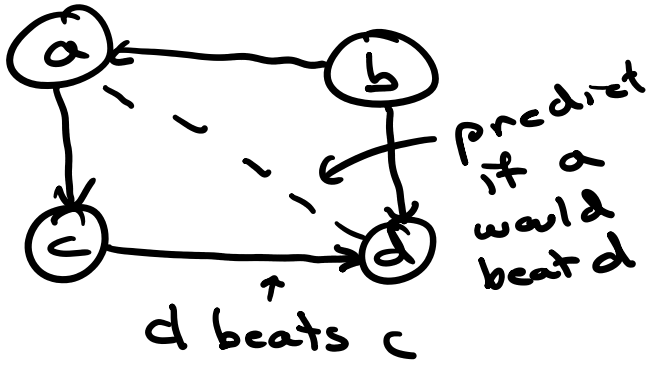
Consider players a, b, c, d

a plays b

b plays d

c plays d

c plays a



We can "orient" our competition

We can "orient" our competition network to have edges point to the victors of each match

We can use PageRank to rank competitors and predict outcomes of future competitions

---

Our experiment

- We have the 2023-2024 NFL season as a graph
- We can weight edges via margins of victory
- We can predict the outcome of SuperBowl 58