L23: PageRank

Jeff M. Phillips

April 9, 2025

Final Report

At most 4 pages/student. Don't cram in too much!

- Succinct title (and names)
- Problem definition and motivation.
- Explain your Data.
- **key idea**
- What did you do (which techniques, an implementation, a comparison, an extension)
- What did you learn? Artifacts (charts, plots, examples, math) and Intuition (in words, did it work?)

Markov Chain: (V, P, q_0) graph

V = set of states → nodes V in $G=(V,E)$

P = probabilistic transition matrix → normalized adjacency matrix

q_0 = initial state

if ergodic not needed arbitrary.

if $G=(V,E)$ is connected, undirected
$\lambda_1(A)$
$|V| > 2$ → ergodic

timeless state

q_∞ = for $i = 1$ to ∞

$q = P q$

q_∞ = answer in limit

unique if ergodic.

Algo for q_*

1. $q_* \leftarrow$ Eigen$(P) \Rightarrow v_1, \ldots$ $v_1 \leftarrow$ top eig
   $q_* = \frac{v_1}{\|v_1\|_1}$

2. $q_* = P^n q_0$ for large n
   $n=20$ precompute $P^n = P \cdot P \cdots P$

3. $q_* =$ for $i = 1$ to n
   $q = Pq$ (power method)

4. Random Walk Reed stats
   Start w/ Burn-in (1000 steps)
   at least 5000 steps

Metropolis Algorithm

Metropolis, Rosenbluth, Rosenbluth, Teller, and Teller in 1953

state ξ of physics

$E(\xi)$ energy

state space X $\xi \in X$

Random walk of $\xi \in X$

so $\Pr(\xi = x) \sim w(\xi) = e^{-E(\xi)}$

Metropolis Algorithm

Metropolis, Rosenbluth, Rosenbluth, Teller, and Teller in 1953

Metropolis on V and w

Initialize $v_0 = [0\ 0\ 0\ \ldots\ 1\ \ldots\ 0\ 0]^T$.
repeat
  Generate $u \sim K(v, \cdot)$ Neighborhood of v.
  if $(w(u) \geq w(v_i))$ **then** if energy is always risk.
    Set $v_{i+1} = u$
  else
    With probability $w(u)/w(v)$ set $v_{i+1} = u$
  else
    Set $v_{i+1} = v_i$ stay
until "converged"
return $V = \{v_1, v_2, \ldots, \}$

PageRank → Web similarity search

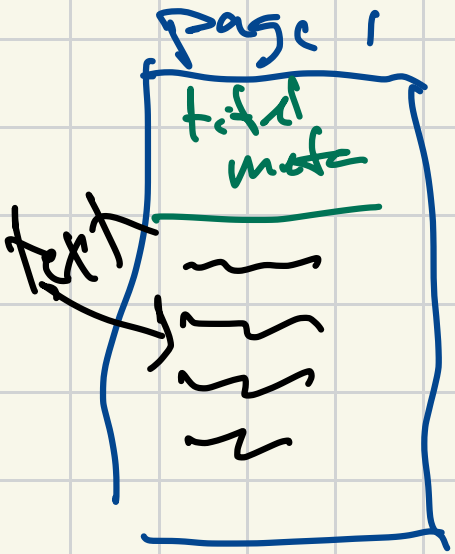
Inverted Index

how to precompute

"apple" → page1, page2, ... page10

"bank" → page1 page2

"car" → page2



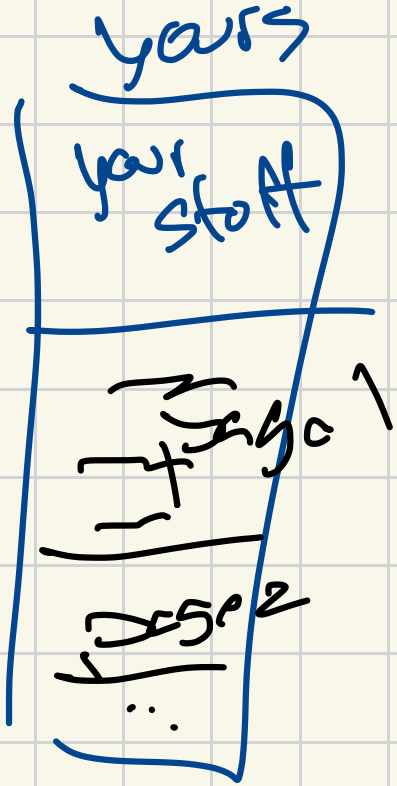
Jaccard on k-grams — MinHash

Cosine on bags-of-words — LSH

Alta Vista

Fake Webpages

Run AllaVista (app1) => page1, page2.



Crawler

program that surfs web.

1. visit page
2. Record info on page for index
3. Go to random linked web page.

with some probability $(1-p)$ or decide to go to random page.

Alternative to Search Engines

pre Google : AltaVista, Lycos
search engines

Built Index Rolodex

Webpage w/ list of links,
organized.

Looksmart , Yahoo!

Page Rank

Idea interesting webpages are linked to by other interesting web pages.

Insight correlated w/ random surfer would do.
what crawler were doing.

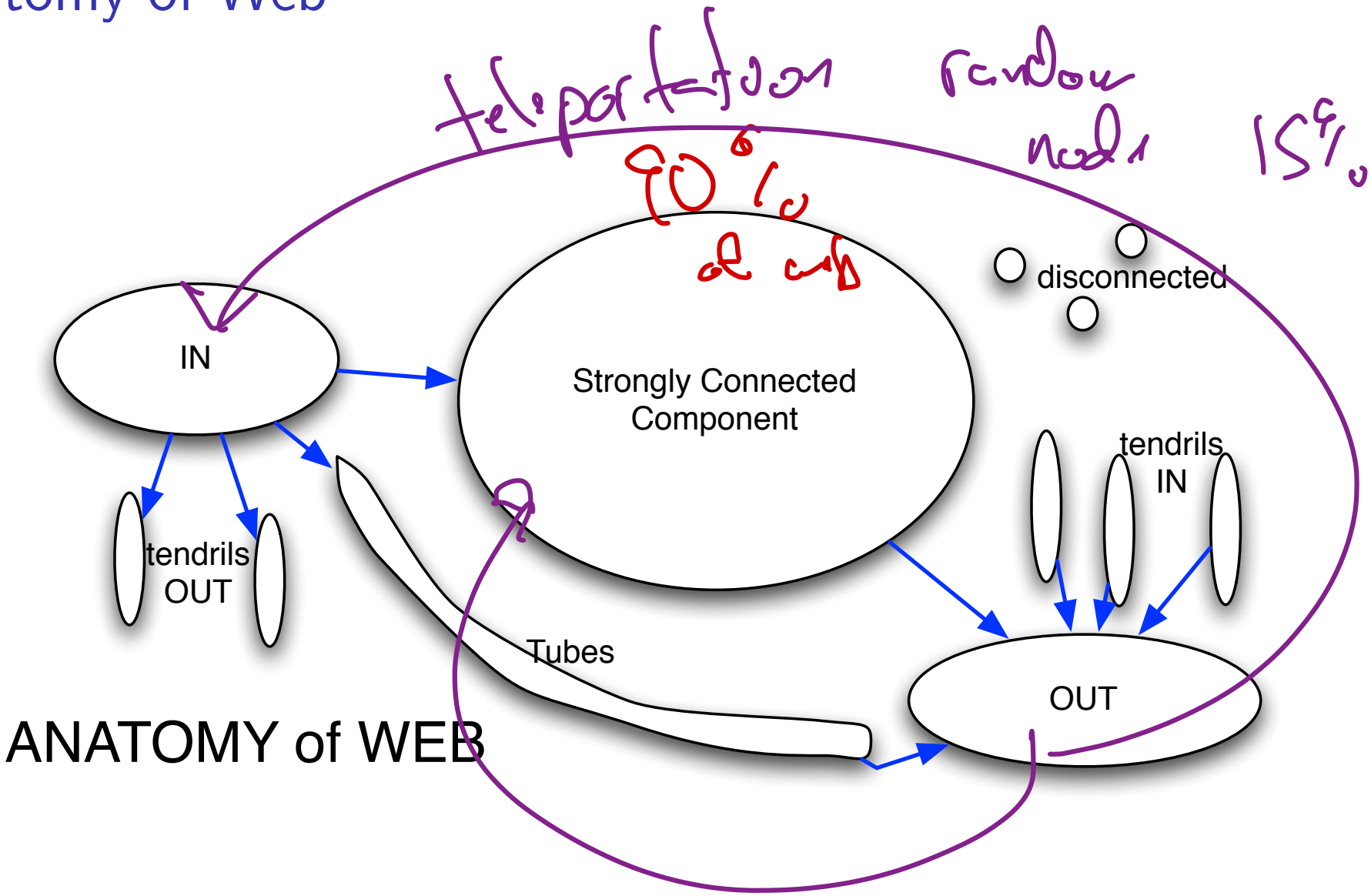
ranking(cosine("apple"), random surfer frequency $g^* \in$ limiting)

Compute g^* on Web Graph

- Crawlers learning web graph
  outgoing links
  V = pages, P = outgoing links
  updating at time
- Goal: rank page i higher if $g^*(i)$ is higher
- each night, run
  $g^* = P(g^*(old))$

updated

Anatomy of Web



Teleportation is important

↳ makes ergodic

$\beta = 0.15$

$$P = L(1-\beta) + \beta Q \qquad Q = \begin{bmatrix} \frac{1}{n} & \cdots \\ \vdots & \end{bmatrix}_{n}$$

web graph → L sparse

Q → Trust Rank

Q non uniform

Spam Farms