More on Rankings

Query-independent LAR

- Have an a-priori ordering of the web pages
- Q: Set of pages that contain the keywords in the query q
- Present the pages in Q ordered according to order \( \pi \)
- What are the advantages of such an approach?

InDegree algorithm

- Rank pages according to in-degree
- \( w_i = |B(i)| \)

PageRank algorithm [BP98]

- Good authorities should be pointed by good authorities
- Random walk on the web graph
  - pick a page at random
  - with probability \( 1 - \alpha \) jump to a random page
  - with probability \( \alpha \) follow a random outgoing link
- Rank according to the stationary distribution
- \( PR(p) = \alpha \sum_{q \in F_p} \frac{PR(q)}{|F(q)|} + (1 - \alpha) \frac{1}{|G|} \)

Markov chains

- A Markov chain describes a discrete time stochastic process over a set of states
  \( S = \{ s_1, s_2, \ldots, s_n \} \)
- According to a transition probability matrix
  \( P = [P_{ij}] \)
  - \( P_{ij} = \text{probability of moving to state } j \text{ when at state } i \)
  - \( L_{ij} = 1 \) (stochastic matrix)
- Memorylessness property: The next state of the chain depends only on the current state and not on the past of the process (first order MC)
  - higher order MCs are also possible

Random walks

- Random walks on graphs correspond to Markov Chains
  - The set of states \( S \) is the set of nodes of the graph \( G \)
  - The transition probability matrix is the probability that we follow an edge from one node to another
An example

\[
\begin{bmatrix}
0 & 1 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 0 \\
0 & 1/2 & 1/2 & 0 & 0 \\
0 & 0 & 0 & 0 & 1/2
\end{bmatrix}
\]

\[
\begin{bmatrix}
0 & 1/2 & 1/2 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 \\
0 & 1/3 & 1/3 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 \\
1/3 & 1/3 & 1/3 & 0 & 0 \\
1/2 & 0 & 0 & 0 & 0/2
\end{bmatrix}
\]

State probability vector

- The vector \( q_i = (q_{1i}, q_{2i}, ..., q_{ni}) \) that stores the probability of being at state \( i \) at time \( t \)
- \( q_{0i} \) is the probability of starting from state \( i \)

\[
q_t = q_0 P^{t-1}
\]

Stationary distribution

- A stationary distribution for a MC with transition matrix \( P \) is a probability distribution \( \pi \), such that \( \pi = \pi P \)
- A MC has a unique stationary distribution if
  - it is irreducible
  - the underlying graph is strongly connected
  - it is aperiodic
  - for random walks, the underlying graph is not bipartite
- The probability \( \pi_i \) is the fraction of times that we visited state \( i \) as \( t \to \infty \)
- The stationary distribution is an eigenvector of matrix \( P \)

Computing the stationary distribution

- The Power Method
  - Initialize to some distribution \( q_0 \)
  - Iteratively compute \( q_t = q_{t-1} P \)
  - After enough iterations \( q_t \approx \pi \)
  - Power method because it computes \( q_t = q_0 P^t \)
- Rate of convergence
  - determined by \( \lambda_2 \)

The PageRank random walk

- Vanilla random walk
  - make the adjacency matrix stochastic and run a random walk

\[
\begin{bmatrix}
0 & 1/2 & 1/2 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 \\
0 & 1 & 0 & 0 & 0 \\
1/3 & 1/3 & 1/3 & 0 & 0 \\
1/2 & 0 & 0 & 0 & 0/2
\end{bmatrix}
\]
The PageRank random walk

• What about sink nodes?
  — what happens when the random walk moves to a node without any outgoing inks?

\[
P = \begin{bmatrix}
0 & 1/2 & 1/2 & 0 & 0 \\
0/5 & 0 & 0 & 0 & 0 \\
0/5 & 1/3 & 1/3 & 0 & 0 \\
1/2 & 0 & 0 & 1/2 & 0 \\
\end{bmatrix}
\]

The PageRank random walk

• Replace these row vectors with a vector \( v \)
  — typically, the uniform vector

\[
P' = \begin{bmatrix}
0 & 1/2 & 1/2 & 0 & 0 \\
1/5 & 1/5 & 1/5 & 1/5 & 1/5 \\
0 & 1 & 0 & 0 & 0 \\
1/3 & 1/3 & 1/3 & 0 & 0 \\
1/2 & 0 & 0 & 1/2 & 0 \\
\end{bmatrix}
\]

\[
P' = P + \alpha v^T \quad \text{d} = \begin{cases} 1 & \text{if its sink} \\ 0 & \text{otherwise} \end{cases}
\]

A PageRank algorithm

• Performing vanilla power method is now too expensive – the matrix is not sparse

\[
q^0 = v \\
t = 1 \\
\text{repeat} \quad q^t = (P')^t q^0 \\
\delta = \|q^t - q^{t-1}\| \\
t = t + 1 \\
\text{until} \quad \delta < \epsilon
\]

Efficient computation of \( y = (P'')^T x \)

\[
y = \alpha P^T x \\
\beta = \|P\| - \|v\| \\
y = y + \beta v
\]

Random walks on undirected graphs

• In the stationary distribution of a random walk on an undirected graph, the probability of being at node \( i \) is proportional to the (weighted) degree of the vertex

• Random walks on undirected graphs are not “interesting”

The PageRank random walk

• How do we guarantee irreducibility?
  — add a random jump to vector \( v \) with prob \( \alpha \)
  — typically, to a uniform vector

\[
P'' = \alpha P' + (1 - \alpha) uv^T, \text{ where } u \text{ is the vector of all 1s}
\]

Effects of random jump

• Guarantees irreducibility
• Motivated by the concept of random surfer
• Offers additional flexibility
  — personalization
  — anti-spam
• Controls the rate of convergence
  — the second eigenvalue of matrix \( P'' \) is \( \alpha \)
Research on PageRank

- Specialized PageRank
  - personalization [BP98]
  - Instead of picking a node uniformly at random favor specific nodes that are related to the user
  - topic sensitive PageRank [H02]
  - Compute many PageRank vectors, one for each topic
  - estimate relevance of query with each topic
  - produce final PageRank as a weighted combination
- Updating PageRank [Chien et al 2002]
- Fast computation of PageRank
  - numerical analysis tricks
  - node aggregation techniques
  - dealing with the "Web frontier"

Topic-sensitive pagerank

- HITS-based scores are very inefficient to compute
- PageRank scores are independent of the queries
- Can we bias PageRank rankings to take into account query keywords?

Topic-sensitive PageRank

- Conventional PageRank computation:
  - \( r^{(t+1)}(v) = \Sigma_{u \in N(v)} r^{(t)}(u)/d(v) \)
  - \( N(v) \): neighbors of \( v \)
  - \( d(v) \): degree of \( v \)
  - \( r = Mx \)
  - \( M' = (1-\alpha)P + \alpha \frac{[1/n]}{n \times n} \)
  - \( r = (1-\alpha)Pr + \alpha p \)
  - \( p = [1/n]_{n \times 1} \)

Conventional PageRank: \( p \) is a uniform vector with values \( 1/n \)

- Topic-sensitive PageRank uses a non-uniform personalization vector \( p \)
- Not simply a post-processing step of the PageRank computation
- Personalization vector \( p \) introduces bias in all iterations of the iterative computation of the PageRank vector

Topic-sensitive PageRank: Overall approach

- Preprocessing
  - Fix a set of \( k \) topics
  - For each topic \( c_j \) compute the PageRank scores of page \( u \) wrt the \( j \)-th topic: \( r(u,j) \)

- Query-time processing:
  - For query \( q \) compute the total score of page \( u \) wrt \( q \) as \( \text{score}(u,q) = \Sigma_{j=1,k} Pr(c_j\mid q) r(u,j) \)

Personalization vector

- In the random-walk model, the personalization vector represents the addition of a set of transition edges, where the probability of an artificial edge \((u,v)\) is \( \alpha p_v \)

- Given a graph the result of the PageRank computation only depends on \( \alpha \) and \( p \):
  \[ PR(\alpha,p) \]
Topic-sensitive PageRank: Preprocessing

- Create $k$ different biased PageRank vectors using some pre-defined set of $k$ categories $(c_1,...,c_k)$
- $T_j$: set of URLs in the $j$-th category
- Use non-uniform personalization vector $p = w_j$ such that:

$$w_j(v) = \begin{cases} \frac{1}{T_j}, v \in T_j \\ 0, \text{otherwise} \end{cases}$$

Topic-sensitive PageRank: Query-time processing

- $D_j$: class term vectors consisting of all the terms appearing in the $k$ pre-selected categories

$$Pr(c_j | q) = \frac{Pr(c_j)Pr(q | c_j)}{Pr(q)} = Pr(c_j) \prod_i Pr(q_i | c_j)$$

- How can we compute $P(c_j)$?
- How can we compute $Pr(q_i | c_j)$?

Comparing LAR vectors

- Comparing results of Link Analysis Ranking algorithms
- Comparing and aggregating rankings

Distance between LAR vectors

- Geometric distance: how close are the numerical weights of vectors $w_1, w_2$?

$$d_i(w_1, w_2) = \sum |w_1[i] - w_2[i]|$$

$$w_1 = [ 1.0 \ 0.8 \ 0.5 \ 0.3 \ 0.0 ]$$
$$w_2 = [ 0.9 \ 1.0 \ 0.7 \ 0.6 \ 0.8 ]$$
$$d_i(w_1, w_2) = 0.1+0.2+0.2+0.3+0.8 = 1.6$$

- Rank distance: how close are the ordinal rankings induced by the vectors $w_1, w_2$?
  - Kendall’s $\tau$ distance

$$d_i(w_1, w_2) = \frac{\text{pairs ranked in a different order}}{\text{total number of distinct pairs}}$$