CS 485 Lecture Notes
3/26/06
Scribe: Liwei Chen

Random walk on directed graphs

Construction:

- At each node, we take an outgoing edge with uniform probability
- For nodes with no outgoing edges, we add a self loop
  - The resulting stationary probability ⇔ page rank of the node
- At each vertex, we also add an edge to every other node with weighted probability
  - 0.15 probability to use additional edges, 0.85 probability to use existing edges
  - Instead of adding n^2 edges, we add a restart node, and edges from every node to it.
  - We also add an edge to every node from the restart node with uniform probability
  - The result is a strongly connected graph with weighted edges

Definition: An aperiodic graph is one if the gcd of all cycle lengths is one (converges to stationary probability)

Theorem: If a graph is strongly connected and aperiodic, then the following is true:

- There is an unique stationary probability
- N(i,t) = number of times we hit node i in t steps, $\lim (t\to\infty)\ N(i,t) / t = \Pi_i$ (page rank)
- Expected time between visits to vertex i is $1/ c_i$

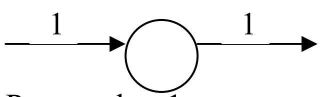
Adjacency matrix

- n by n matrix with entries being 1 if the corresponding column and row entries are adjacent.
- Normalize the rows so they sum to one (allow matrix to be multiplied by a probability vector)
- Transpose A (allow for solutions in the form of $A*\Pi = \Pi$)
  - Similar to solutions of the form $A*x = \lambda * x$
  - Which simplifies to $(A- \lambda*I)*x = 0$
  - Which only has a nontrivial solution if $\det (A- \lambda*I) = 0$
  - The resulting equations will be an n^{th} degree polynomial in λ with n solutions $\lambda_1, \lambda_2\ldots \lambda_n$ (eigenvalues)
  - Associated with each λ_i is solution x_i, (eigenvector)

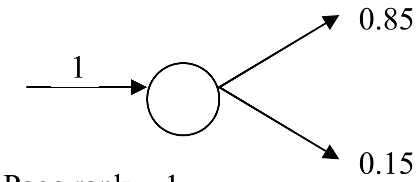
Markov Process: random process whose future behavior depends on current state but not how you got there

Terms:
Persistent state – strongly connected component with no out edges

Aperiodic
Irreducible – single strongly connected component
Ergodic state- persistent and aperiodic
Markov chain is ergodic if every state is ergodic = strongly connected component and aperiodc



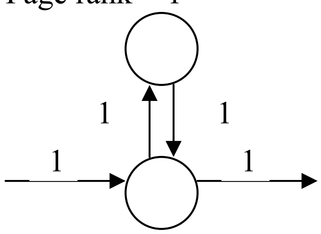
Page rank = 1



restart node

Page rank = 1

Page rank = 1



Page rank = 2

If we had a restart node to the graph on left, adding self loops will only increase page rank by 6.67

Discovery time(v): time it takes to first reach vertex v from uniform random start

We can modify internal and outgoing links, but we can’t change incoming links and thus it is hard to change your own discovery time.

Increasing Page Rank

- Capture random walk – short cycles
- Capture restart – buy 20,000 URLs, have them all point to your webpage

No known efficient algorithm to calculate discovery time (calculation for page rank is expensive but amortized over all nodes makes it manageable)

Altman and Tennehaltz – “ranking systems, the pagerank axioms”
The paper talks about 5 axioms which if your ranking system satisfies, will be the same as the page rank algorithm

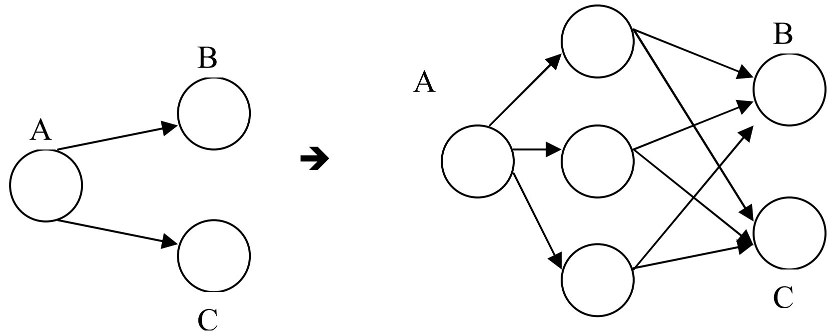
However, the paper assumes unweighted strongly connected graph, which doesn’t apply in our situation since we added weights to the graph when we added the restart node.

Axioms:

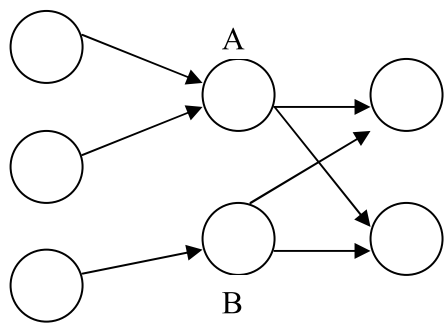
1. Isomorphic vertices have equal rank

2. Adding self loops to vertex v doesn’t change ranking of any pair of other vertices. Rank of v can only increase

3. Vote by committee



4. If several vertices have the same set of successors, they can be collapsed into a single vertex



(A and B can be merged into the same node)

5. Convert same rank nodes