CSE 3504: Probabilistic Analysis of Computer Systems
Spring 2016
Project – Part 2

Delivery deadline: Friday, April 22nd, 2016.
Goal: To implement Google's PageRank algorithm and understand the stochastic process in its core.
Prerequisites: some programming knowledge.
Group Size: 2. **Programming Language**: Any

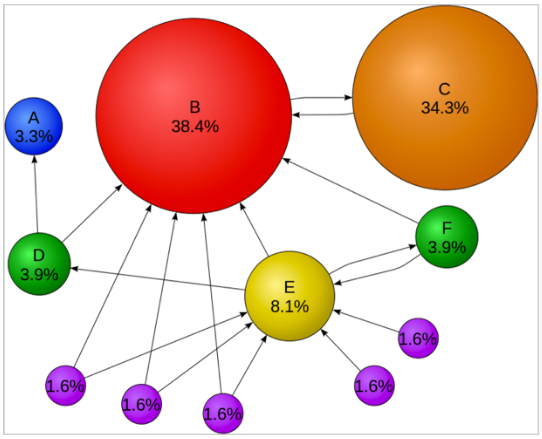
Background 1: Markov Chains

Certain systems change through a finite number of states over time and behave randomly according to certain probabilities. If the probability that the system will be in a particular state during a given time period depends only on its state during the previous time period, the system is called a Markov chain. Such system can be modeled using transition matrices. A transition matrix is an n x n square matrix P, where n is the number of states of the system, whose elements P(i, j) are positive and whose rows sum to 1. For each i and j, P(i, j) is the probability that given the system was in state i in one time period, it will be in state j during the next time period.

Background 2: PageRank Algorithm

PageRank is an algorithm used by Google Search to rank websites in their search engine results. PageRank was named after Larry Page, one of the founders of Google. PageRank is a way of measuring the importance of website pages. PageRank works by counting the number and quality of links to a page to determine a rough estimate of how important the website is. The underlying assumption is that more important websites are likely to receive more links from other websites.

The algorithm represents websites (also called documents) as nodes in a directed graph where an edge from website A to website B represents a hyperlink from website A to website B. Simply, the algorithm ranks website by counting the number of links incoming to this website from other websites. In addition, it considers the quality of links by inheriting the PageRank of the source website to increase/decrease the PageRank of the destination website. The picture shows a graphical representation of some web pages (A, B, C, etc.). Page C has a higher PageRank than page E, even though there are fewer links to C; the one link to C comes from an important page and hence is of high value.



Did you know?
Startup companies ask famous websites and blogs to write about their new idea and provide a link to the startup web page. For example, a newly published mobile app can get a high PageRank faster if TechCrunch or TheNextWeb websites wrote about it. One link from a famous website worth hundred links from other smaller one.

Please, read algorithm detailed description from the Wikipedia page:
https://en.wikipedia.org/wiki/PageRank#Algorithm

Algorithm Steps Summary:

Given a web of n pages, construct an n × n matrix P as:

$$\boldsymbol{p_{ij}} = \begin{cases} \boldsymbol{\frac{1}{n_j}} & \boldsymbol{if\ page\ j\ links\ to\ page\ i} \\ \boldsymbol{0} & \boldsymbol{otherwise} \end{cases}$$

where n_j is the number of outgoing links from node j.
note that the sum of *j*th column is n_j / n_j = 1, So P is a Markov matrix.

The initial state vector p(0) has equal probability over all entries. For example, if we have 5 pages, the initial state vector is [0.2 0.2 0.2 0.2 0.2]

Now, you have the initial state vector and the transition matrix, can you compute the state vector at p(1)? p(2)?

PageRank algorithm modifies the transition matrix P (before any computation) as follows:
P = 0.85 P + 0.15 where 0.85 is called a damping factor (read more about the damping factor).
Basically, the dumping factor is introduced to scale the transition matrix to all positive numbers to ensure that there is a steady state vector for the system.

Finally, the steady state vector is considered the ranking of the pages. Remember that the steady state vector is equal to $\boldsymbol{p(0)}.\boldsymbol{P^n}$ for sufficiently large n. (alternatively, a steady state vector is where the distribution does not change over two subsequent values of n)

In this project, you will be asked to produce the steady state vector (PageRank(s) for a set of pages). Then, sort the vector from highest PageRank to lowest.

Stochastic Interpretation of PageRank:
The authors of PageRank considered web surfing as a stochastic process. It can be thought as a model of user behavior. We assume that there is a "random surfer" who is given a web page at random and keeps clicking on links, never hitting "back" but eventually gets bored and starts on another random page. Surfer clicks on a link on the current page with probability 0.85; opens up a random page with probability 0.15. A page's rank is the probability the random user will end up on that page, or equivalently the fraction of time the random user spends on that page in the long run.

Practice 1 (70 points):

This programming assignment asks you to implement the simplest variation of PageRank on a given dataset. Download the dataset named *hollins.dat* from HuskyCT.

1. Load the text file into appropriate data structure.
    a. First line represents number of nodes (web pages) V and number of edges (hyperlinks) E
    b. Following V lines are website index and link. (index is not rank)
    c. Following E lines are source node and destination node.
2. Run PageRank algorithm and output the result to a text file ranking web pages.
    a. The first line of the file should contain the website index and link of the website ranked first by your algorithm.
    b. The second line of the file should contain the website index and link of the website ranked second by your algorithm.
    c. And so on.
    d. The file should contain V lines (ranking website from first to last).
3. Change the damping factor from 0.85 to 0.95 (that changes the whole transition probabilities of the matrix P). Repeat point (2) and answer the following questions:
    a. Does increasing the damping factor change your ranks?
    b. What is the new rank of the page that was ranked first when using a damping factor of 0.85?
    c. What is the new rank of the page that was ranked last when using a damping factor of 0.85?
4. Change the damping factor from 0.85 to 0.50 (that changes the whole transition probabilities of the matrix P). Repeat point (2) and answer the following questions:
    a. Does lowering the damping factor change your ranks?
    b. What is the new rank of the page that was ranked first when using a damping factor of 0.85?
    c. What is the new rank of the page that was ranked last when using a damping factor of 0.85?

Practice 2 (30 points):

You are the owner of a new startup and you want more people to visit your website. Unfortunately, you don't have enough budget to pay for Google Ads. So, you decided you are going to best utilize your website code to show in the first Google search results. For better understanding, you want to read more about other Google's algorithms used to order its results. Read more about how Google search works here:
https://www.google.com/insidesearch/howsearchworks/thestory/
Choose one algorithm that they run on Step 2 of the three steps and answer the following questions:

1. What is the algorithm name?
2. Write algorithm description and/or pseudo code.
3. What is the mathematical and/or statistical concepts underlying this algorithm?
4. How can you modify your startup website to show up in first search results provided that you now know that particular algorithm?
5. Provide references for your answer.

Maximum report length: 3 pages.
You cannot choose PageRank algorithm for your report.

Extra Points (10 points):

Post one Tip (1-3 sentences) on HuskyCT discussion board (Project Part 2) about how website owners can improve their websites to show first in Google search results.

This is not required in the assignment delivery. Points received in this question can make up for points lost in the midterm or project part 1. Tips collected from this discussion board will collectively appear on a blog post "50+ Tips to Increase Your Website Rank from UConn CS Students" with your names.

Important:

- On the online submission form on HuskyCT, submit the following:
    1. One CSV file containing the output of practice 1.
    2. One source code file.
    3. One PDF or Word document containing the answer to practice 2.
- You may compress the three files in one .zip or .rar file.
- Write the two students names on the PDF or Word document.
- Only one of the two students should make the submission. No need for redundancy.