Lecture 3 Handout — Page Rank

36-350, Data Mining

29 August 2008

This is a *brief* over-view of the ideas behind page-rank that I talked about in class today. One could add a lot of refinements and wrinkles, and in applications they matter, but this gets at the principles.

Start with a random web-page, say i. Suppose this page has at least one out-going link, to various other pages, $j_1, j_2, \ldots j_{i_n}$. A simple random walk would choose each of those links with equal probability:

$$P_{ij} = \begin{cases} \frac{1}{i_n} & \text{if } j \in \{j_1, j_2, \ldots j_{i_n}\} \\ 0 & \text{otherwise} \end{cases}$$

If starting page i has no out-going links, then $P_{ij} = 1/n$, where $n =$ the total number of pages, for all j. That is, when the walk comes to a dead end, it re-starts to a random location.

Let X_t be the page the random walk is visiting at time t, and $N(i, n)$ be the number of $t \leq n$ where $X_t = i$, the number of times X_t visits i. The **page rank** of a page i is how often it is visited in the course of a very long random walk:

$$\rho(i) = \lim_{n \to \infty} \frac{N(i, n)}{n}$$

How do we know this is well-defined? Maybe the ratio doesn't converge at all, or it converges to something which depends on the page we started with.

Well, we know that the random walk is a Markov chain: the state of the chain is the page being visited. (Why?) We also see that there is some probability that the chain will go from any page to any other page eventually (if only by eventually hitting a dead-end page and then randomly re-starting). So the state-space of the Markov chain is **strongly connected**. The number of pages is finite. And remember from probability models that a finite Markov chain whose state-space is strongly connected obeys the **ergodic theorem**, which says, precisely, that the fraction of time the chain spends in any one state goes to a well-defined limit, which doesn't depend on the starting state.

So one way to calculate the page-rank is just to simulate, i.e., to do a random walk in the way I described. But this is slow, and there is another way.

Suppose that ν is a probability vector on the states, i.e., it's an n-dimensional vector whose entries are non-negative and sum to one. Then, again from probability models, if the distribution at time t is ν_t, the distribution one time-step

later is

$$\nu_{t+1} = \nu_t P = \nu_0 P^t$$

with P the transition matrix we defined earlier. It's another result from probability that the ν_t keep getting closer and closer to each other, so that

$$\lim_{t \to \infty} \nu_0 P^t = \rho$$

where ρ is a special probability distribution satisying the equation

$$\rho = \rho P$$

That is, ρ is an **eigenvector** of P with **eigenvalue** 1. In fact, this ρ is the same as the ρ we get from the ergodic theorem. So rather than doing the simulation, we could just calculate the eigenvectors of P, which is often faster and more accurate than the simulation.

Unpacking the last equation, it says

$$\rho(i) = \sum_j \rho(j) P_{ij}$$

which means that pages with high page-rank are ones which are reached, with high probability, from other pages of high page-rank. This sounds circular, but, as we've seen, it isn't. In fact, one way to compute it is to start with ν_0 being the uniform distribution, i.e., $\nu_0(i) = 1/n$ for all i, and then calculate $\nu_1, \nu_2, \ldots$ until the change from ν_t to ν_{t+1} is small enough to tolerate. That is, initially every page has equal page-rank, but then it shifts towards those reached by many strong links (ν_1), and then to those with many strong links from pages reached by many strong links (ν_2), and so forth.

There is a very simple way to use page-rank to do search:

- Calculate ρ once.
- Given a query Q, find all the pages containing all the terms in Q.
- Return the matching page i where $\rho(i)$ is highest (or the k pages with the highest page-rank, etc.)

However, this is *too* simple — it presumes that a highly-linked-to page is always good, no matter how tangential it might be to the topic at hand. From the beginning, Google has used a combination of page-rank, similarity scores, and many other things (most of them properietary) to determine its search results.

Computationally, all that matters is that there is a set of nodes with links between them; the same algorithm could be applied to any sort of graph or network. EigenFactor (`eigenfactor.org`) is a site which ranks academic journals by using the citations in the papers they publish. There are also other ways of using link structure to rank web-pages — Jon Kleinberg's "hubs and authorities" system distinguishes between the value of pages as *authorities* about a particular topic, and *hubs* that aggregate information about many topics (see `http://www.cs.cornell.edu/home/kleinber/auth.pdf`), and a version of this is, apparently, incorporated into Ask.com.