Computation, Information, and Intelligence (COMS/ENGRI/INFO/COGST 172), Fall 2005
10/14/05: **Lecture 21 aid — link-based measures of webpage importance**

Agenda: Introduce Google's PageRank algorithm; perhaps get through the HITS (a.k.a. hubs-and-authorities) algorithm.

References (both available online):

Sergey Brin and Lawrence Page, "The Anatomy of a Large-Scale Hypertextual Web Search Engine", *WWW7*, 1998. Note that there is a typo in the PageRank equation given there.

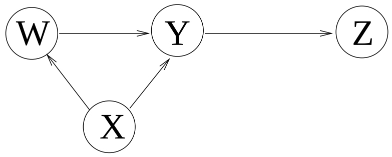
Jon Kleinberg, "Authoritative sources in a hyperlinked environment", *Journal of the ACM*, 1999. This is an extended version of a paper appearing in *Proceedings of the Symposium on Discrete Algorithms (SODA)*, 1998.

I. Definitions and conventions For a document d, we define:

To(d):	the set of documents that link to d.
From(d):	the set of documents that are linked to by d.

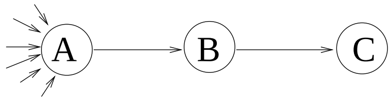
We can write $|\text{To}(d)|$ and $|\text{From}(d)|$ for the in-degree and out-degree of d, respectively.
For simplicity, we assume that documents have no self-links.

II. Example: in-degree vs. "prestige"



We have To(W) consisting only of X, and To(Y) containing W and X. From(X) is the two documents W and Y, and From(Z) doesn't contain any documents. Furthermore, note that the in-degree of W is the same as the in-degree of Z, and that the out-degrees of W and Y are equal.

III. Example: propagation of "prestige"



IV. PageRank We give an explicitly iterated version here. Let ϵ be some number between 0 and 1.

- For every d_j in the n-document corpus, set $\text{score}^{(0)}(d_j)$ to $1/n$.
- Repeat until the scores "converge" (the change in scores between one timestep and the next is sufficiently small): set

$$\text{score}^{(t+1)}(d_j) = \frac{\epsilon}{n} + (1-\epsilon) \sum_{d \in \text{To}(d_j)} \frac{\text{score}^{(t)}(d)}{|\text{From}(d)|}.$$

(OVER)

V. The hubs and authorities algorithm Here's how the algorithm processes queries.[1] For each document d_j, we want to compute its *authority score* a_j and its *hub score* h_j.

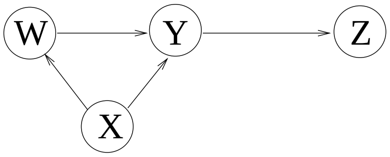
1. Initialization: For every document d_j, set both a_j and h_j to 1.

Repeat the following steps in order until no "significant" change occurs:

2. Update authority scores: For every document d_j, change a_j to $\sum_{d_k \text{ in } \mathrm{To}(d_j)} h_k$.
3. Sum-normalize authority scores: compute authnorm $= \sum_{k=1}^{n} a_k$;
   then, for every document d_j, change a_j to a_j/authnorm.
4. Update hub scores: For every document d_j, change h_j to $\sum_{d_k \text{ in } \mathrm{From}(d_j)} a_k$.
5. Sum-normalize hub scores: compute hubnorm $= \sum_{k=1}^{n} h_k$;
   then, for every document d_j, change h_j to h_j/hubnorm.

VI. Example calculations

Here's the first example from the other side of this handout, again:



		W		X		Y		Z	
		auth	(hub)	auth	(hub)	auth	(hub)	auth	(hub)
a.	Init	1	(1)	1	(1)	1	(1)	1	(1)
b.	Update-a	1	”	0	”	2	”	1	”
c.	SNorm-a	1/4	”	0	”	1/2	”	1/4	”
d.	Update-h	”	(1/2)	”	(3/4)	”	(1/4)	”	(0)
e.	SNorm-h	”	(1/3)	”	(1/2)	”	(1/6)	”	(0)
f.	Update-a	1/2	”	0	”	5/6	”	1/6	”
g.	Snorm-a	1/3	”	0	”	5/9	”	1/9	”

Note that convergence has not yet occurred (which can be seen by going through another round of the algorithm), but since these computations are just for demonstration purposes we'll stop here anyway.

[1] We're using sum-normalization rather than (the analog of) length-normalization to make the calculations a little easier. Also, while it would have been more consistent with our presentation of PageRank to add time-step superscripts to the hub and authority scores, doing so seems to complicate the notation too much. Finally, rather than work with all the documents in the corpus, Kleinberg proposed that the algorithm be applied only to those documents in a *root set* of (hopefully) relevant documents determined via content-based IR given a specific query. (One may expand this root set by adding in the documents that link to or are linked from some document in the root set.) But we'll ignore this difference here, since a similar thing could, in principle, be performed for PageRank as well.