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

References (both available online):

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

    $\text{To}(d)$: the set of documents that link to $d$.
    $\text{From}(d)$: 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 $\text{To}(W)$ consisting only of $X$, and $\text{To}(Y)$ containing $W$ and $X$. $\text{From}(X)$ is the two documents $W$ and $Y$, and $\text{From}(Z)$ 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 $\epsilon$ 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

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

VI. Example calculations
Here’s the first example from the other side of this handout, again:

\[ W \rightarrow Y \rightarrow Z \]

\( X \)

<table>
<thead>
<tr>
<th></th>
<th>W auth (hub)</th>
<th>X auth (hub)</th>
<th>Y auth (hub)</th>
<th>Z auth (hub)</th>
</tr>
</thead>
<tbody>
<tr>
<td>a. Init</td>
<td>1 (1)</td>
<td>1 (1)</td>
<td>1 (1)</td>
<td>1 (1)</td>
</tr>
<tr>
<td>b. Update-a</td>
<td>1 &quot;</td>
<td>0 &quot;</td>
<td>2 &quot;</td>
<td>1 &quot;</td>
</tr>
<tr>
<td>c. SNorm-a</td>
<td>1/4 &quot;</td>
<td>0 &quot;</td>
<td>1/2 &quot;</td>
<td>1/4 &quot;</td>
</tr>
<tr>
<td>d. Update-h</td>
<td>&quot; (1/2)</td>
<td>&quot; (3/4)</td>
<td>&quot; (1/4)</td>
<td>&quot; (0)</td>
</tr>
<tr>
<td>e. SNorm-h</td>
<td>&quot; (1/3)</td>
<td>&quot; (1/2)</td>
<td>&quot; (1/6)</td>
<td>&quot; (0)</td>
</tr>
<tr>
<td>f. Update-a</td>
<td>1/2 &quot;</td>
<td>0 &quot;</td>
<td>5/6 &quot;</td>
<td>1/6 &quot;</td>
</tr>
<tr>
<td>g. Snorm-a</td>
<td>1/3 &quot;</td>
<td>0 &quot;</td>
<td>5/9 &quot;</td>
<td>1/9 &quot;</td>
</tr>
</tbody>
</table>

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.