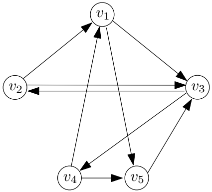
ENGG1410F Tutorial
Introduction to Page Ranks

Yufei Tao

Department of Computer Science and Engineering
Chinese University of Hong Kong

Google made its first debut with page ranks, which represent a technique for ranking the webpages on the Internet by importance. Today we will give a short introduction to this technique. Interestingly, at its core, the technique requires computing just an eigenvector.

Let us model the Internet as a graph. Each webpage is represented as a node. Given two nodes $v_1, v_2 \in V$, there is an edge from v_1 to v_2 if the webpage v_1 has a hyperlink to the webpage v_2.



Let us imagine the following process that mimics the behavior of a user surfing randomly:

1. Let u be a random webpage in the Internet.

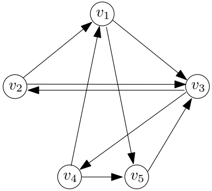
Let us imagine the following process that mimics the behavior of a user surfing randomly:

1. Let u be a random webpage in the Internet.
2. With probability α:
   2.1 If there is at least one out-going link on u
      2.2 Click on a random hyperlink in u
      2.3 Set u to the new webpage that opens up.
   2.4 Repeat from Step 2.

Let us imagine the following process that mimics the behavior of a user surfing randomly:

1. Let u be a random webpage in the Internet.
2. With probability α:
   2.1 If there is at least one out-going link on u
      2.2 Click on a random hyperlink in u
      2.3 Set u to the new webpage that opens up.
   2.4 Repeat from Step 2.
3. With probability $1 - \alpha$:
   3.1 Set u to a random webpage in the Internet.
   3.2 Repeat from Step 2.

The value of α is often set to 0.85 in practice.



For example, suppose we are at v_3. Conceptually this is what we do:

- Toss a coin that heads with probability α.
- If the coin comes up heads, jump to v_2 or v_4 with equal chance.
- If the coin comes up tails, jump to v_1, v_2, ..., v_5 with equal chance.

We'd like to ask this question:

If the user keeps surfing like this, where will s/he be at the 100000000000-th page visited?

We'd like to ask this question:

> If the user keeps surfing like this, where will s/he be at the 100000000000-th page visited?

But this is a wrong question, because the process is random, such that there won't be a deterministic answer. The correct question to ask is:

> If the user keeps surfing like this, what is the probability that s/he will land on v_1 as the 100000000000-th page?

We'd like to ask this question:

> If the user keeps surfing like this, where will s/he be at the 100000000000-th page visited?

But this is a wrong question, because the process is random, such that there won't be a deterministic answer. The correct question to ask is:

> If the user keeps surfing like this, what is the probability that s/he will land on v_1 as the 100000000000-th page?

The probability is the page rank of v_1. Of course, the same question can also be asked about any other page.

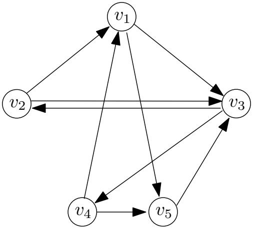
If the user keeps surfing like this, what is the probability that s/he will land on v_1 as the 100000000000-th page?

Why the number 100000000000? Interestingly, the theory of random walks (which we will not get into today) tells us that the probability remains the same as long as a sufficiently large number of steps have been performed! In other words, it won't matter if you replace 100000000000 with, say, 100000000001!

If the user keeps surfing like this, what is the probability that s/he will land on v_1 as the 100000000000-th page?

The rationale behind page ranks is this:

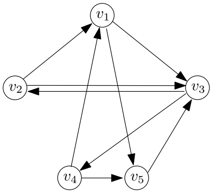
A page v is more "important", i.e., having a higher page rank, if a random surfer has a larger chance landing on v after a sufficiently large number of steps.



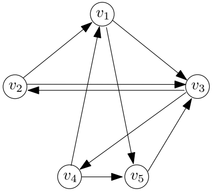
The page ranks of $v_1, \ldots, v_5$ are 0.1716, 0.1666, 0.3214, 0.1666, and 0.1737, respectively. Note that the sum of all the page ranks is 1.

Remaining question: How to calculate them?

Let n be the number of nodes.
Define $\boldsymbol{M} = [m_{ij}]$ as an $n \times n$ matrix
where m_{ij} is the probability of moving from node v_j to node v_i.



For example, when $\alpha = 0.85$, $m_{23} = 0.455$. Why? See next.



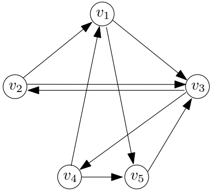
Recall: Suppose we are at v_3. Conceptually this is what we do:

- Toss a coin that heads with probability α.
- If the coin comes up heads, jump to v_2 or v_4 with equal chance.
- If the coin comes up tails, jump to v_1, v_2, ..., v_5 with equal chance.

So the probability to go from v_3 to v_2 is:

$$\alpha/2 + (1 - \alpha)/5$$

which is 0.455 for $\alpha = 0.85$.



You can verify:

$$\boldsymbol{M} = \begin{bmatrix} 0.03 & 0.455 & 0.03 & 0.455 & 0.03 \\ 0.03 & 0.03 & 0.455 & 0.03 & 0.03 \\ 0.455 & 0.455 & 0.03 & 0.03 & 0.88 \\ 0.03 & 0.03 & 0.455 & 0.03 & 0.03 \\ 0.455 & 0.03 & 0.03 & 0.455 & 0.03 \end{bmatrix}$$

Theory of random walks tells us some important facts:

- $\boldsymbol{M}$ must have an eigenvalue 1.
- The page ranks make an eigenvector under the eigenvalue 1!

Theory of random walks tells us some important facts:

- $\boldsymbol{M}$ must have an eigenvalue 1.
- The page ranks make an eigenvector under the eigenvalue 1!

In our example:

$$\boldsymbol{M} = \begin{bmatrix} 0.03 & 0.455 & 0.03 & 0.455 & 0.03 \\ 0.03 & 0.03 & 0.455 & 0.03 & 0.03 \\ 0.455 & 0.455 & 0.03 & 0.03 & 0.88 \\ 0.03 & 0.03 & 0.455 & 0.03 & 0.03 \\ 0.455 & 0.03 & 0.03 & 0.455 & 0.03 \end{bmatrix}$$

You can verify that $\begin{bmatrix} 0.1716 \\ 0.1666 \\ 0.3214 \\ 0.1666 \\ 0.1737 \end{bmatrix}$ is indeed an eigenvector of $\boldsymbol{M}$ under eigenvalue 1.

We now have an algorithm to compute the page ranks:

1. Obtain $\boldsymbol{M}$.
2. Obtain an arbitrary eigenvector $\boldsymbol{p}$ of $\boldsymbol{M}$ under the eigenvalue 1.
3. Scale $\boldsymbol{p}$ into $c\boldsymbol{p}$ with a proper real number c so that all components of $c\boldsymbol{p}$ add up to 1.
4. $c\boldsymbol{p}$ now stores the page ranks of all vertices.