Combating Web Spam with TrustRank

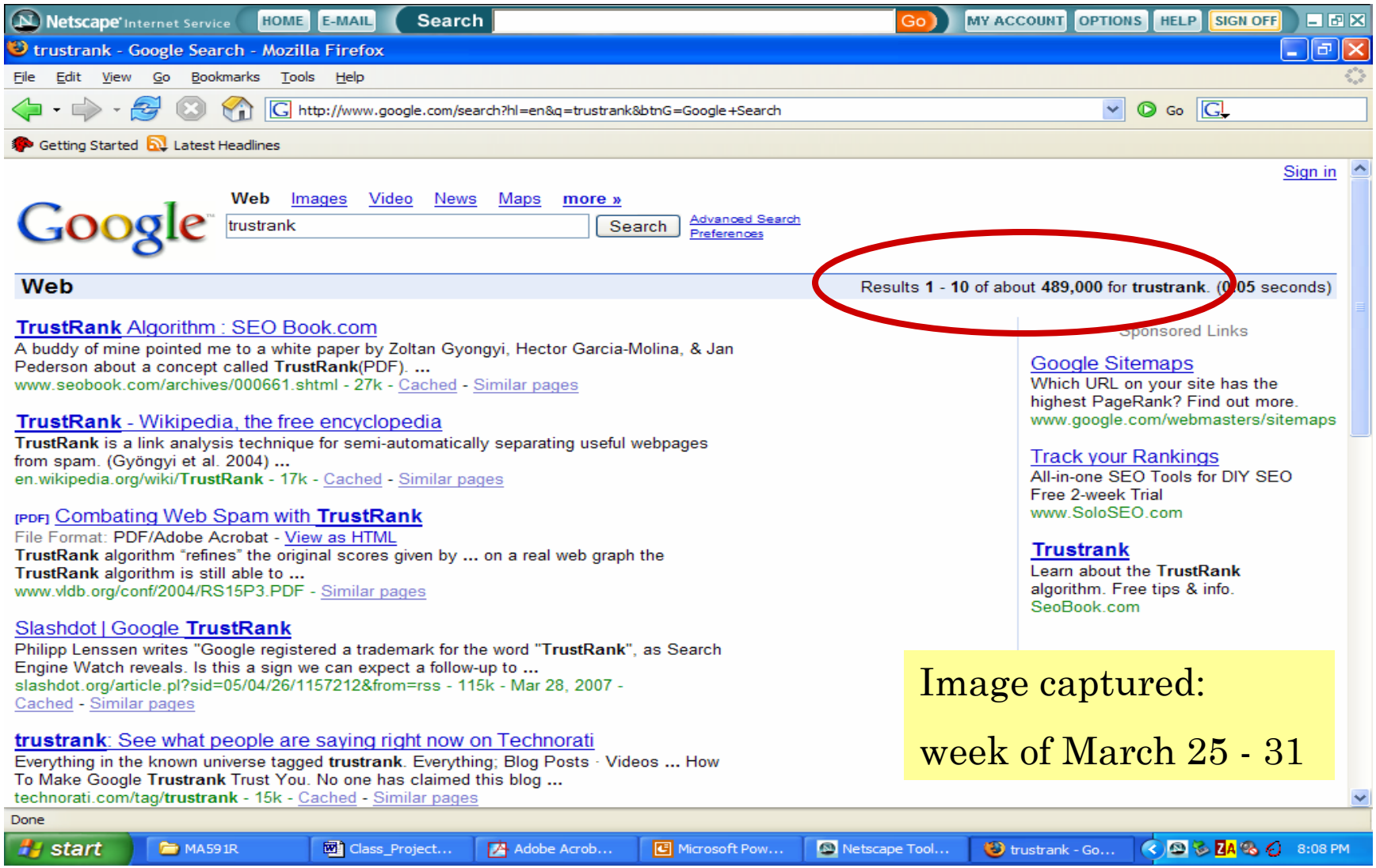
Authors:
Gyöngyi, Garcia-Molina, and Pederson
Published in:
Proceedings of the 30th VLDB Conference
Year:
2004

Presentation by: Rebecca Wills
Date: April 4, 2007

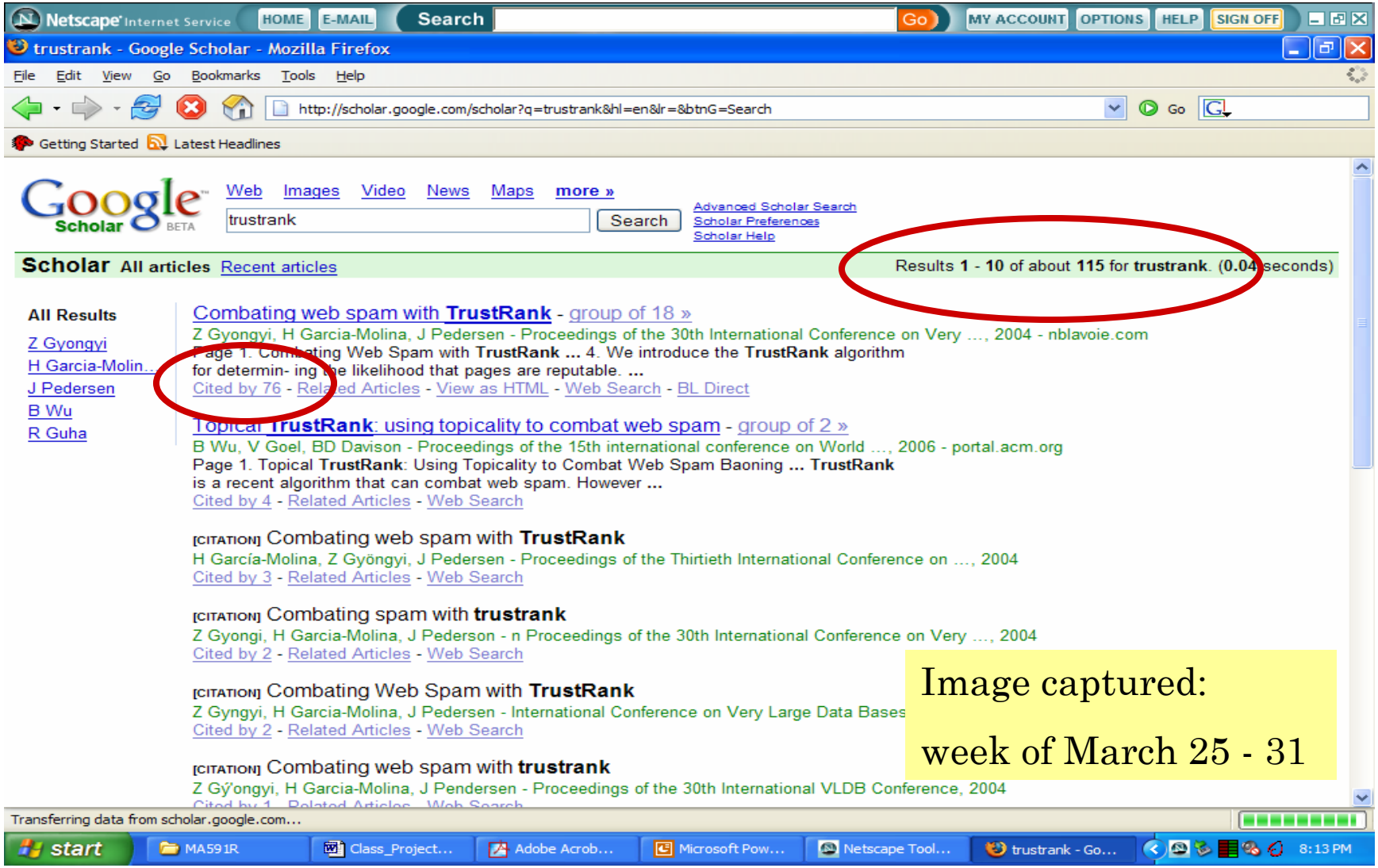
Questions we will answer today include…

- Why should we care about this paper?
- What is TrustRank?
- Is TrustRank mathematically sound?

Why should we care about this paper?



Why should we care about this paper?



Why should we care about this paper?

Google registered the trademark for “TrustRank” on March 16, 2005.

The algorithm receives human assistance.

The authors state, “We believe that our work is a first attempt at formalizing the problem and at introducing a comprehensive solution to assist in the detection of Web spam.”

What is Web spam?

The term refers to hyperlinked webpages that are created to mislead search engines.

Example 1: webpages containing numerous words having nothing to do with the webpage using text invisible to humans but observed by search engines

Example 2: webpages receiving links from numerous other real or phony webpages for the sole purpose of increasing PageRank

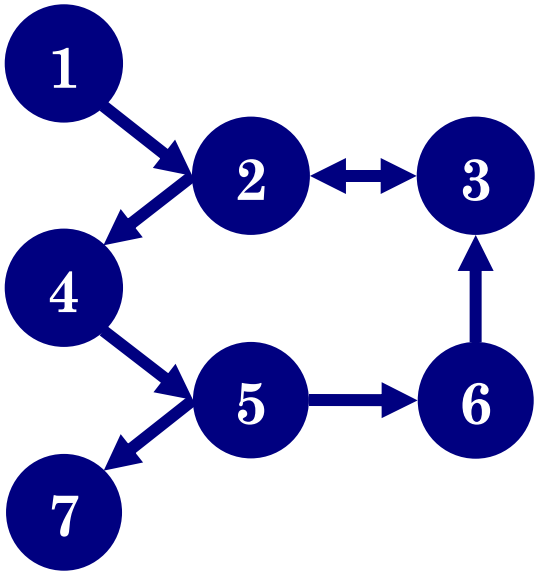
Research Goal of the Authors

"Our research goal is to assist the human experts who detect web spam. In particular, we want to identify pages and sites that are likely to be spam or that are likely to be reputable."

What is TrustRank?

An algorithm that creates a personalization vector to be used in the PageRank computation for the purpose of combating Web spamming

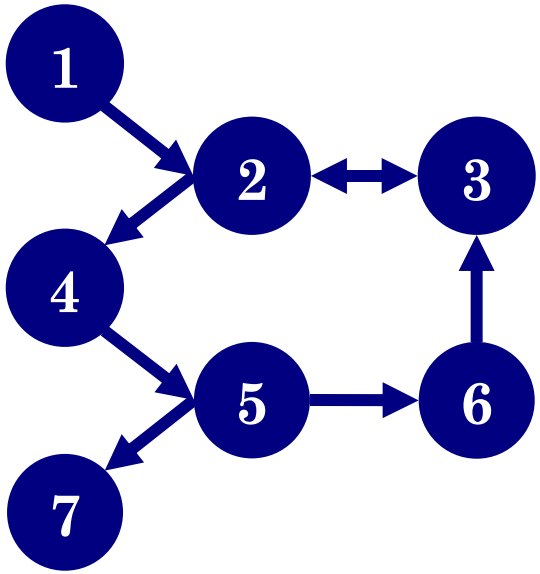
Example Web Graph



$V = \{1, 2, 3, 4, 5, 6, 7\}$

$E = \{(1,2), (2,3), (2,4), (3,2), (4,5), (5,6), (5,7), (6,3)\}$

Example Web Graph



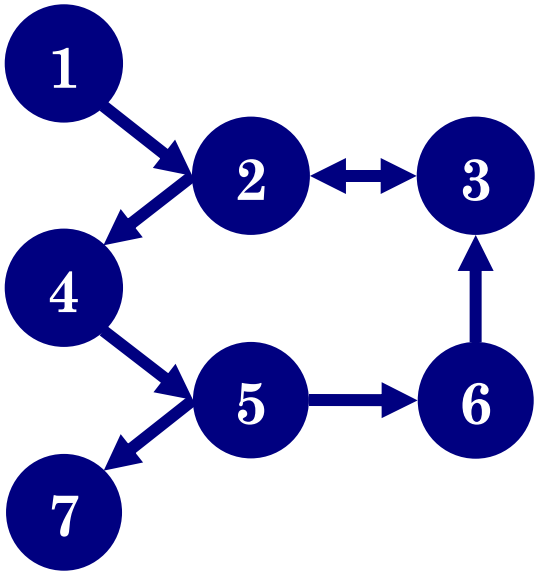
$$H = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Basic PageRank

$$r^T = \alpha\left(r^T H\right) + \frac{(1-\alpha)}{n} e^T$$

$0 \le \alpha < 1$, $n = |V|$, and $e^T = \begin{pmatrix} 1 & 1 & \dots & 1 \end{pmatrix}$

Example Web Graph



$$H = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Basic PageRank

$$r^T = \alpha\left(r^T H\right) + \frac{(1-\alpha)}{n} e^T$$

Note: The authors do not directly address the dangling node issue. They do mention the "biased PageRank" definition, where v is any probability vector, but they compare their results to the "regular PageRank" definition.

Why is the definition okay?

Biased PageRank

$$r^T = \alpha\left(r^T H\right) + \left(1-\alpha\right)v^T$$

Class Definition of PageRank (with dangling node fix v^T)

$$\pi^T = \pi^T\left(\alpha\left[H + av^T\right] + \left(1-\alpha\right)ev^T\right)$$

$$= \alpha\left(\pi^T H\right) + \underbrace{\left[\alpha\left(\pi^T a\right) + \left(1-\alpha\right)\right]}_{\text{Scalar}}v^T$$

So, $\pi^T\left(I - \alpha H\right) = \left[\alpha\left(\pi^T a\right) + \left(1-\alpha\right)\right]v^T$

$$\Rightarrow \pi^T = \left[\alpha\left(\pi^T a\right) + \left(1-\alpha\right)\right]v^T\left(I - \alpha H\right)^{-1}$$

Why is the definition okay?

Biased PageRank

$$r^T = \alpha\left(r^T H\right) + \left(1-\alpha\right)v^T$$

Class Definition of PageRank (with dangling node fix v^T)

So, $\pi^T\left(I-\alpha H\right) = \left[\alpha\left(\pi^T a\right) + \left(1-\alpha\right)\right]v^T$

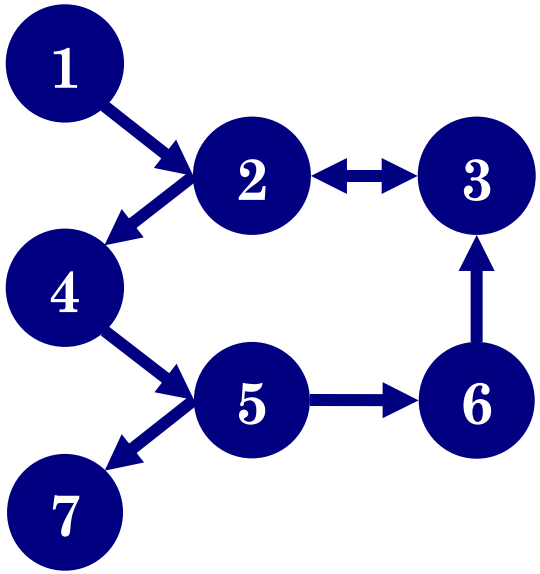
$$\Rightarrow \pi^T = \left[\alpha\left(\pi^T a\right) + \left(1-\alpha\right)\right]v^T\left(I-\alpha H\right)^{-1}$$

$x^T\left(I-\alpha H\right) = \beta v^T, \quad \pi^T = {x^T}/{x^T e}$, $\beta > 0$

produces the PageRank vector.

[Langville & Meyer, 2006, Thm. 7.3.1, pages 73 - 74]

Example Web Graph



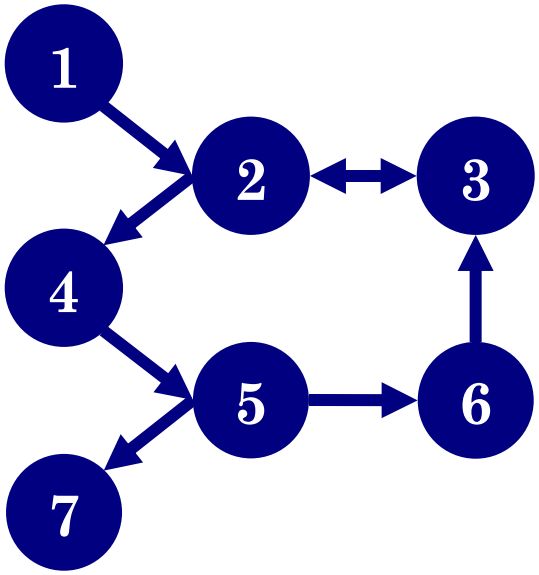
$$H = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

TrustRank

$$r^T = \alpha\left(r^T H\right) + \left(1 - \alpha\right) v^T$$

where v^T is formed to combat Web spam

Overall Idea



TrustRank

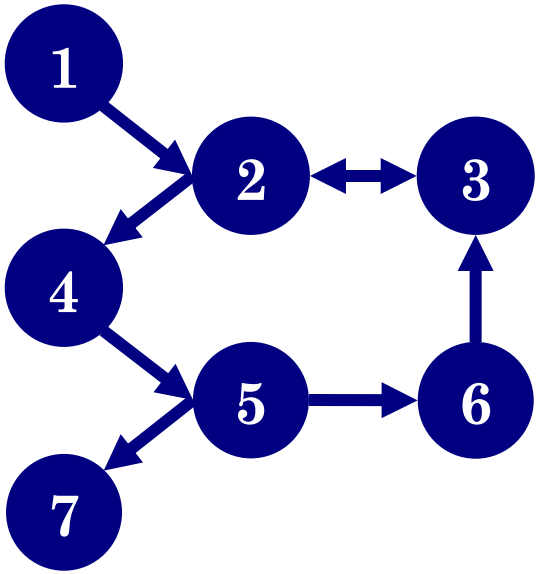
$$r^T = \alpha\left(r^T H\right) + (1-\alpha)v^T$$

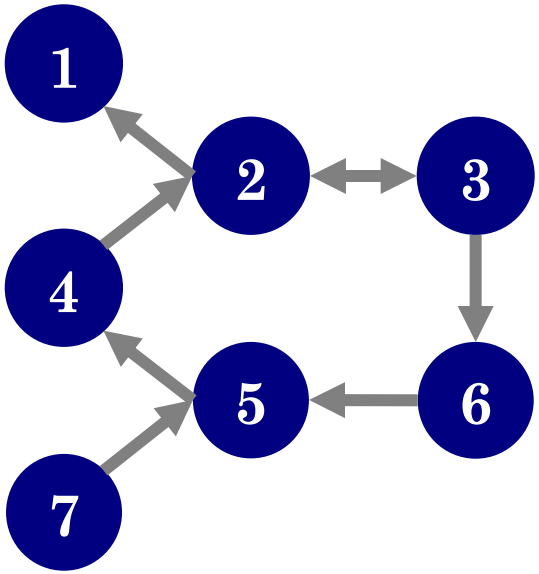
Step 1: Select a small "seed set" of webpages.

Step 2: Identify good webpages from the "seed set".

Step 3: Create personalization vector based on identification of good webpages.

Step 1: Select a small "seed set" of webpages.



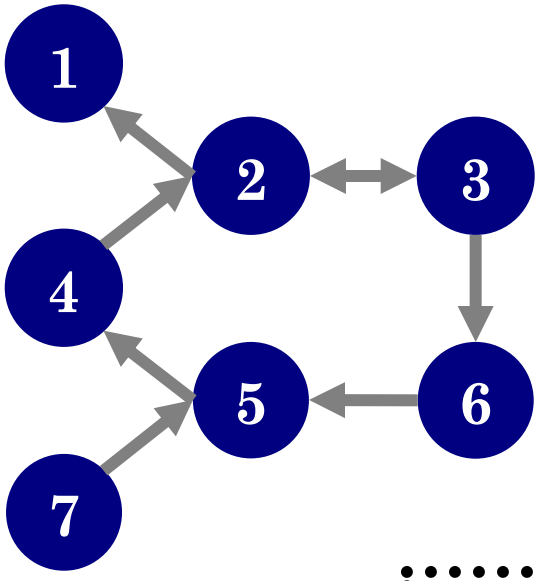


$V = \{1, 2, 3, 4, 5, 6, 7\}$

$E = \{(1,2), (2,3), (2,4), (3,2), (4,5), (5,6), (5,7), (6,3)\}$

$E^{-1} = \{(2,1), (3,2), (4,2), (2,3), (5,4), (6,5), (7,5), (3,6)\}$

Step 1: Select a small "seed set" of webpages.



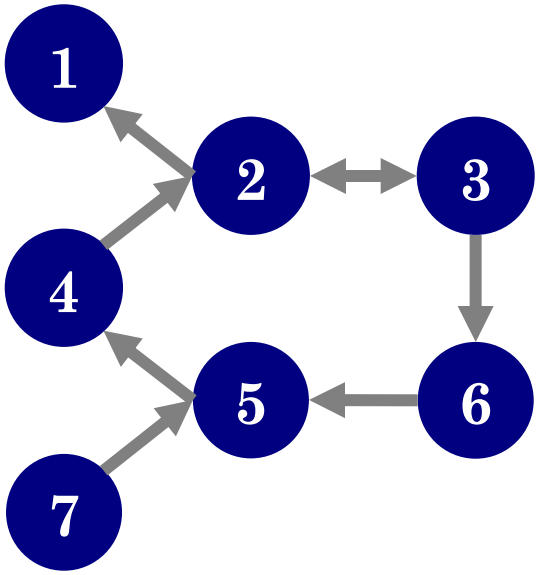
$$U = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

Inverse PageRank

$$s^T = \alpha\left(s^T U\right) + \frac{(1-\alpha)}{n} e^T$$

Note: The authors emphasize that Inverse PageRank (named based on E^{-1}) works well in practice.

Step 1: Select a small "seed set" of webpages.



Function: SelectSeed

Initial iterate: $s_0^T = e^T$

While: $k \le M$

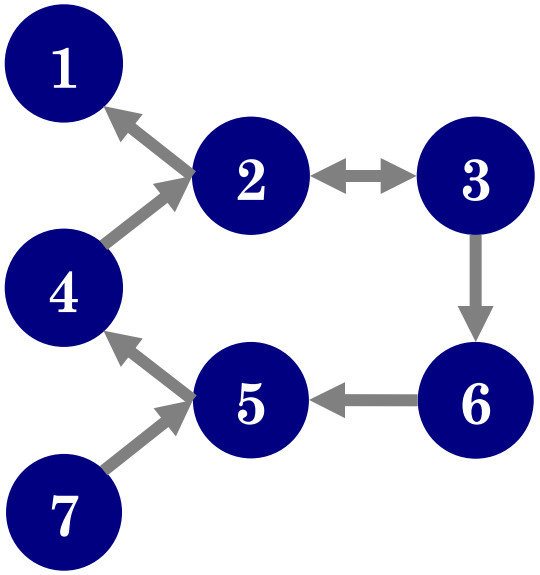
Do: $s_k^T = \alpha\left(s_{k-1}^T U\right) + \dfrac{(1-\alpha)}{n} e^T,\ k \ge 1$

This is based on the belief that trust flows out of good seed webpages.

It gives preference to webpages from which many other webpages can be reached.

See Maple file for implementation of SelectSeed for this example.

Step 1: Select a small "seed set" of webpages.



Function: SelectSeed

Initial iterate: $s_0^T = e^T$

While: $k \le M$

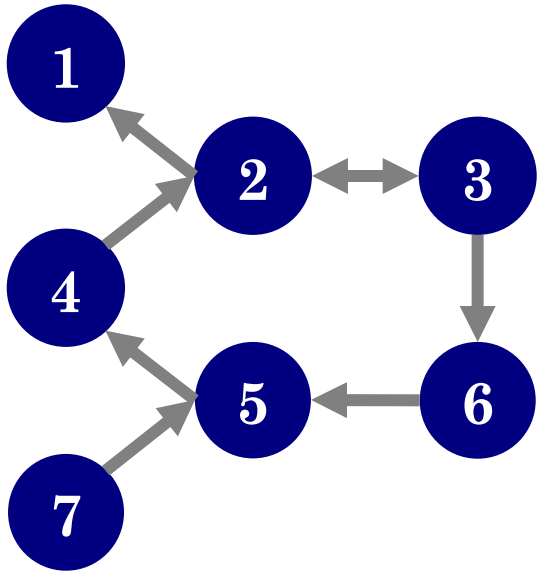
Do: $s_k^T = \alpha\left(s_{k-1}^T U\right) + \dfrac{(1-\alpha)}{n} e^T,\ k \ge 1$

Note: This algorithm is the Jacobi Method applied to $r^T(I - \alpha U) = {}^{(1-\alpha)}/_{n} e^T$.

The diagonal part of $(I - \alpha U)$ is I, and the sum of the upper and lower triangular parts is αU.

See Maple file for implementation of SelectSeed for this example.

Step 1: Select a small "seed set" of webpages.



Function: SelectSeed

Initial iterate: $s_0^T = e^T$

While: $k \le M$

Do: $s_k^T = \alpha\left(s_{k-1}^T U\right) + \dfrac{(1-\alpha)}{n} e^T,\ k \ge 1$

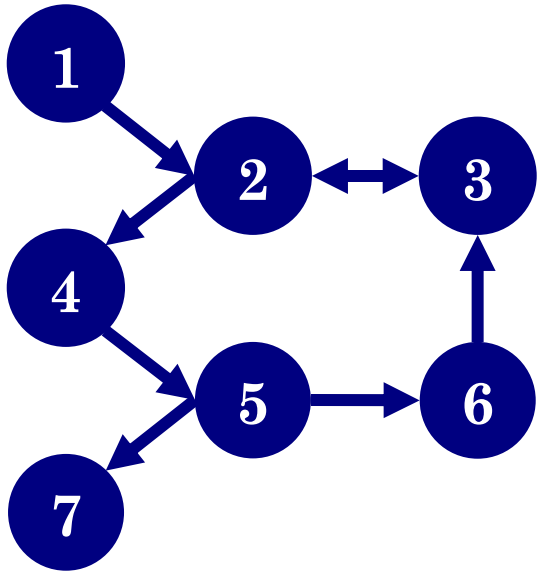
For $\alpha = 0.85$ and $M = 20$, we obtain

$s_{20}^T \approx (0.08 \quad 0.14 \quad 0.08 \quad 0.10 \quad 0.09 \quad 0.06 \quad 0.02)$.

Suppose we want to check the top 3 webpages (in blue above).

Then, our seed set is $S = \{2,4,5\}$.

Step 2: Identify good webpages from "seed set".



Back to the original graph

$$V = \{1,2,3,4,5,6,7\}.$$

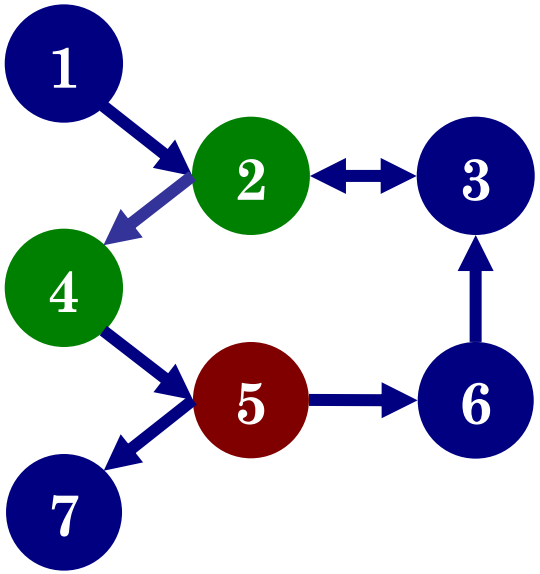
$$S = \{2,4,5\}.$$

Oracle function:

$$O(i) = \begin{cases} 1, & \text{if webpage } i \text{ is good} \\ 0, & \text{if webpage } i \text{ is bad.} \end{cases}$$

This is the step requiring human involvement, and it is, to some extent, subjective.

Step 2: Identify good webpages from "seed set".



$V = \{1,2,3,4,5,6,7\}$.

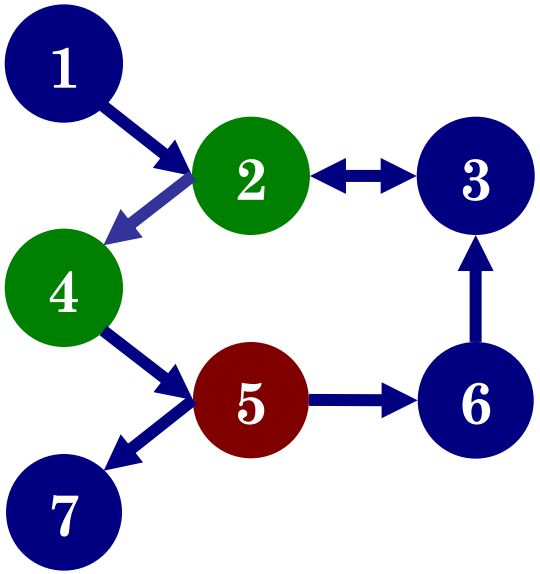
$S = \{2,4,5\}$.

Oracle function:

$$O(i) = \begin{cases} 1, & \text{if webpage } i \text{ is good} \\ 0, & \text{if webpage } i \text{ is bad.} \end{cases}$$

$S^+ = \{2,4\}$ and $S^- = \{5\}$.

Step 3: Create personalization vector based on identification of good webpages.

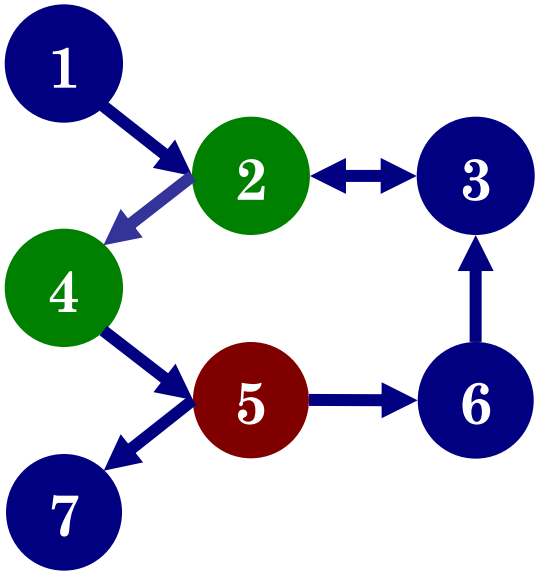


$$S^+ = \{2,4\} \text{ and } S^- = \{5\}.$$

$$v^T = \begin{pmatrix} 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 & 0 \end{pmatrix}$$

Comment: I think it's interesting that the authors go through the trouble of making the personalization vector a probability vector even though the PageRank vector will not be a probability vector. Also, they do not use a probability vector to initialize the Inverse PageRank algorithm.

Compute TrustRank



Function: TrustRank

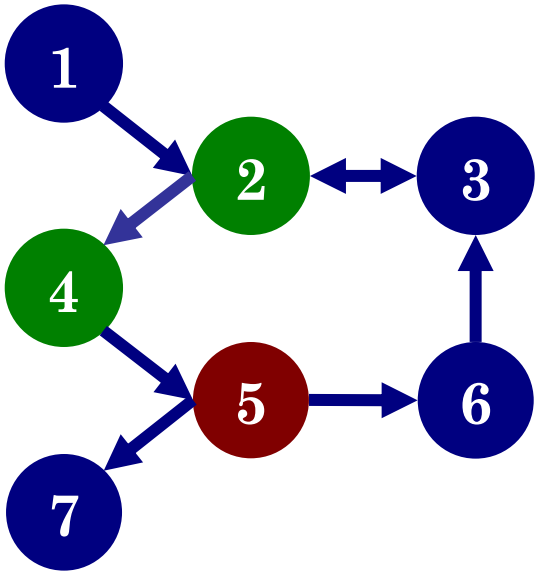
Initial iterate: $\boldsymbol{r}_0^{\boldsymbol{T}} = \boldsymbol{v}^{\boldsymbol{T}}$

While: $\boldsymbol{k} \leq \boldsymbol{M}$

Do: $\boldsymbol{r}_k^{\boldsymbol{T}} = \alpha\left(\boldsymbol{r}_{k-1}^{\boldsymbol{T}}\boldsymbol{H}\right) + (1-\alpha)\boldsymbol{v}^{\boldsymbol{T}},\ \boldsymbol{k} \geq 1$

Note: This $\boldsymbol{M}$ and α can be different from the ones used for Inverse PageRank.

Compute TrustRank



Function: TrustRank

Initial iterate: $r_0^T = v^T$

While: $k \le M$

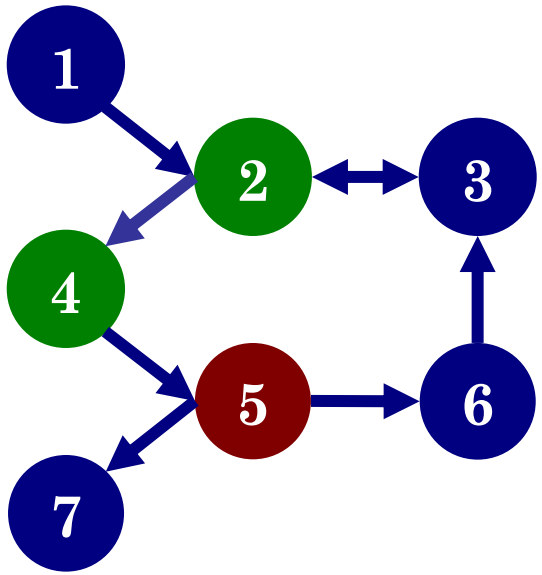
Do: $r_k^T = \alpha\left(r_{k-1}^T H\right) + (1-\alpha)v^T,\ k \ge 1$

Note: This algorithm is the Jacobi Method applied to $r^T(I - \alpha H) = (1 - \alpha)v^T$.

The diagonal part of $(I - \alpha H)$ is I, and the sum of the upper and lower triangular parts is αH.

See Maple file for implementation of TrustRank for this example.

Compute TrustRank



Function: TrustRank

Initial iterate: $r_0^T = v^T$

While: $k \le M$

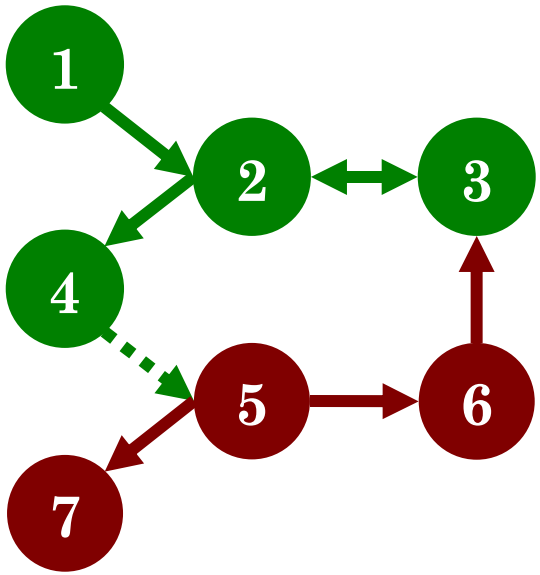
Do: $r_k^T = \alpha\left(r_{k-1}^T H\right) + (1-\alpha)v^T,\ k \ge 1$

For $\alpha = 0.85$ and $M = 20$, we obtain

$r_{20}^T \approx (0 \quad 0.18 \quad 0.12 \quad 0.15 \quad 0.13 \quad 0.05 \quad 0.05)$.

See Maple file for implementation of TrustRank for this example.

Compute TrustRank



Interestingly, for the whole Web graph, the authors identified:

Good webpages, $V^+ = \{1, 2, 3, 4\}$, and
Bad webpages, $V^- = \{5, 6, 7\}$.

The "Basic PageRank" algorithm ranked **good webpage 3** higher than **bad webpage 5**, but the TrustRank algorithm did not. *(Perhaps, they should have identified good and bad webpages differently for the example.)*

For $\alpha = 0.85$ and $\boldsymbol{M} = 20$, we obtain

TrustRank: $\boldsymbol{r}_{20}^{\,T} \approx (0 \quad 0.18 \quad 0.12 \quad 0.15 \quad 0.13 \quad 0.05 \quad 0.05)$.

Basic PageRank: $\boldsymbol{x}_{20}^{\,T} \approx (0.02 \quad 0.26 \quad 0.21 \quad 0.17 \quad 0.16 \quad 0.09 \quad 0.09)$.

See Maple file for implementation of PageRank for this example.

Experiments

- Web data
  - Entire AltaVista index (August 2003)
  - Site-level Web graph
    - 31 million vertices
    - 13 million without inlinks
- Seed set
  - 25,000 candidates → reduced to 7,900 → then 1,250
  - 178 selected high-quality sites
- Evaluation sample
  - 1000 manually tagged sites
- Oracle: Gyöngyi (the first author of the paper)

Experiments

- Manual evaluation took weeks
- Compared Inverse PageRank to other options for selecting the "seed set" (such as using webpages with high Basic PageRank scores)
- Observed that websites with highest inverse PageRank scores showed a heavy bias toward spam
- Removed all websites not listed in any major web directories
- Final filter – only selected websites with a clearly identifiable authority

Experiments

Section 6.2: Seed Set

The authors indicate that they compare inverse PageRank to high PageRank (seed set made up of pages with highest basic PageRank scores). They state:

"We describe these experiments in [4]. Due to space limitations, here we just note that inverse PageRank turned out to be slightly better at identifying useful seed sets. Thus, for the rest of our experiments, we relied on the inverse PageRank method."

I searched the Web and never found [4].

The above statement reminds me of Fermat's comment about his last theorem.

Experiments

See slide 19 from below webpage.

Each bucket represents 5% of the total PageRank score. There are 20 buckets.

TrustRank buckets have the same number of websites as PageRank buckets.

Picture of graph available at:

http://infolab.stanford.edu/~zoltan/presentations/trustrank-vldb-2004-09-02.pdf

Experiments

See slide 20 from below webpage.

Picture of graph available at:

http://infolab.stanford.edu/~zoltan/presentations/trustrank-vldb-2004-09-02.pdf

Also discussed: Assessing Trust

Oracle Function:

$$O(i) = \begin{cases} 1, & \text{if webpage } i \text{ is good} \\ 0, & \text{if webpage } i \text{ is bad.} \end{cases}$$

Trust function:
For webpage i, $T(i) = Pr[O(i) = 1]$.

Threshold trust property:
For webpage i, $T(i) > \delta \Leftrightarrow O(i) = 1$.

Signal function:

$$I(T, \quad O, \quad i, \quad j) = \begin{cases} 1, & T(i) \geq T(j) \text{ and } O(i) < O(j) \\ 1, & T(i) \leq T(j) \text{ and } O(i) > O(j) \\ 0, & \text{otherwise.} \end{cases}$$

Assessing Trust

Suppose $X \subseteq V$ with m randomly selected elements.
Let $P = \{(i, j) \in X \times X: i \neq j\}$. Then, $|P| = m(m-1)$.

Pairwise Orderedness:

$$\text{pairord}(T, \quad O, \quad P) = \frac{|P| - \sum_{(i,j) \in P} I(T, \quad O, \quad i, \quad j)}{|P|}$$

Precision:

$$\text{prec}(T, \quad O) = \frac{\left|\{i \in X : T(i) > \delta \text{ and } O(i) = 1\}\right|}{\left|\{i \in X : T(i) > \delta\}\right|}$$

Recall:

$$\text{rec}(T, \quad O) = \frac{\left|\{i \in X : T(i) > \delta \text{ and } O(i) = 1\}\right|}{\left|\{i \in X : O(i) = 1\}\right|}$$

Assessing Trust

Seed set: $S \subseteq V$ with L elements.
$S^+ = \{i \in S: O(i) = 1\}$ and $S^- = \{i \in S: O(i) = 0\}$.

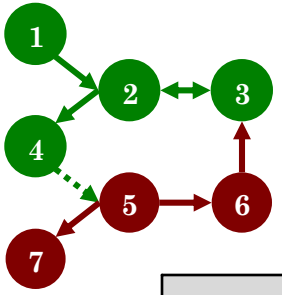
Ignorant trust function:

$$T_0(i) = \begin{cases} 1, & i \in S^+ \\ 0, & i \in S^- \\ \frac{1}{2}, & i \in V - S. \end{cases}$$

M-step trust function:

$$T_M(i) = \begin{cases} 1, & i \in S^+ \text{ or } i \notin S \text{ and } \exists j \in S^+ \text{ s.t. } d(i,j) \le M \text{ and } d(i,k) > M \ \forall k \in S^- \\ 0, & i \in S^- \\ \frac{1}{2}, & i \in V - S. \end{cases}$$

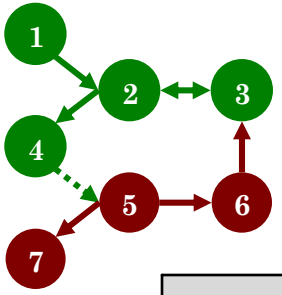
Back to Example



Suppose $S = \{1, 3, 6\}$. Then, $S^+ = \{1, 3\}$ and $S^- = \{6\}$.

Vertex i	Oracle $O(i)$	Ignorant Trust $T_0(i)$	1-step Trust $T_1(i)$	2-step Trust $T_2(i)$	3-step Trust $T_3(i)$
1					
2					
3					
4					
5					
6					
7					

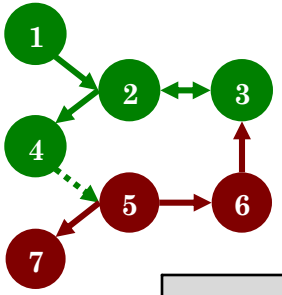
Back to Example



Suppose $S = \{1, 3, 6\}$. Then, $S^+ = \{1, 3\}$ and $S^- = \{6\}$.

Vertex i	Oracle $O(i)$	Ignorant Trust $T_0(i)$	1-step Trust $T_1(i)$	2-step Trust $T_2(i)$	3-step Trust $T_3(i)$
1	1				
2	1				
3	1				
4	1				
5	0				
6	0				
7	0				

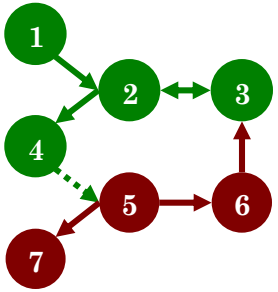
Back to Example



Suppose $S = \{1, 3, 6\}$. Then, $S^+ = \{1, 3\}$ and $S^- = \{6\}$.

Vertex i	Oracle $O(i)$	Ignorant Trust $T_0(i)$	1-step Trust $T_1(i)$	2-step Trust $T_2(i)$	3-step Trust $T_3(i)$
1	1	1	1	1	1
2	1	1/2	1	1	1
3	1	1	1	1	1
4	1	1/2	1/2	1	1
5	0	1/2	1/2	1/2	1
6	0	0	0	0	0
7	0	1/2	1/2	1/2	1/2

Back to Example



Now, suppose $X = V$. Then, $|P| = 7(6) = 42$.

M	pairoid(T_M, O, P)	prec(T_M, O) if $\delta = 1/2$	rec(T_M, O) if $\delta = 1/2$
0	$(42 - 8)/42 = 17/21$	1	1/2
1	$(42 - 4)/42 = 19/21$	1	3/4
2	$(42 - 0)/42 = 1$	1	1
3	$(42 - 8)/42 = 17/21$	4/5	1

Summary of Paper

- Formalizes the problem of Web spam and spam detection algorithms
- Introduces Inverse PageRank to select seed sets
- First use of an oracle to assess webpages
- Introduces the TrustRank algorithm (which is the PageRank algorithm with a carefully chosen personalization vector)
- Provides empirical results
- Defines trust assessing metrics