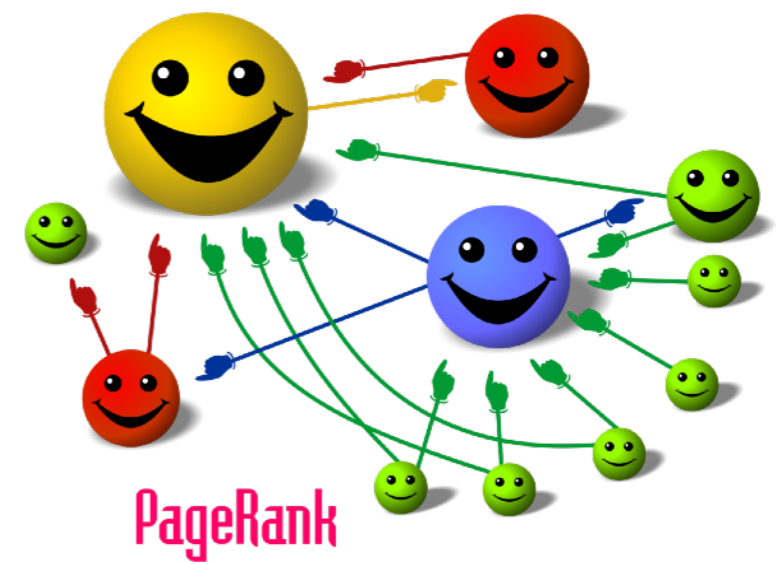


The PageRank Algorithm



Why PageRank?

- Suppose you have a directed graph
 - Websites linking to one another
 - Recommendation votes on eBay, or AirBnB, NetFlix, etc.
 - Scientists referring to each other's works
 - Neighborhoods in cities connected by movement of pedestrians
 - Recommendation for leadership of communities
- How do you associate a good “popularity” or “rank” value to each node in the graph?
- This is what the PageRank Algorithm is about.

PageRank

- The PageRank Algorithm as invented by Larry Page in 1998 when he was a graduate student at Stanford
- He started a research project called “BackRub”
- Sergey Brin joined the project pretty much right away
- They went on to write the paper on the right.
- Goal was to “bring order into the Web”



The Anatomy of a Large-Scale Hypertextual Web Search Engine

Sergey Brin and Lawrence Page

*Computer Science Department,
Stanford University, Stanford, CA 94305, USA
sergey@cs.stanford.edu and page@cs.stanford.edu*

Abstract

In this paper, we present Google, a prototype of a large-scale search engine which makes heavy use of the structure present in hypertext. Google is designed to crawl and index the Web efficiently and produce much more satisfying search results than existing systems. The prototype with a full text and hyperlink database of at least 24 million pages is available at <http://google.stanford.edu/>. To engineer a search engine is a challenging task. Search engines index tens to hundreds of millions of web pages involving a comparable number of distinct terms. They answer tens of millions of queries every day. Despite the importance of large-scale search engines on the web, very little academic research has been done on them. Furthermore, due to rapid advance in technology and web proliferation, creating a web search engine today is very different from three years ago. This paper provides an in-depth description of our large-scale web search engine -- the first such detailed public description we know of to date. Apart from the problems of scaling traditional search techniques to data of this magnitude, there are new technical challenges involved with using the additional information present in hypertext to produce better search results. This paper addresses this question of how to build a practical large-scale system which can exploit the additional information present in hypertext. Also we look at the problem of how to effectively deal with uncontrolled hypertext collections where anyone can publish anything they want.

Inventor

- Larry Page patented the procedure
 - US Patent 6,285,999
 - Filed Jan 9, 1998
 - Granted Sep 4, 2001
 - Owner is Stanford University
- Probably one of the most lucrative patents of all times



US006285999B1

United States Patent (10) Patent No.: **US 6,285,999 B1**
Page (45) Date of Patent: **Sep. 4, 2001**

(54) **METHOD FOR NODE RANKING IN A LINKED DATABASE**
(75) Inventor: **Lawrence Page, Stanford, CA (US)**
(73) Assignee: **The Board of Trustees of the Leland Stanford Junior University, Stanford, CA (US)**
(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.
(21) Appl. No.: **09/004,827**
(22) Filed: **Jan. 9, 1998**
Related U.S. Application Data
(60) Provisional application No. 60/035,205, filed on Jan. 10, 1997.
(51) Int. Cl.⁷ **G06F 17/30**
(52) U.S. Cl. **707/5; 707/7; 707/501**
(58) **Field of Search** **707/100, 5, 7, 707/513, 1-3, 10, 104, 501; 345/440; 382/226, 229, 230, 231**

(56) **References Cited**
U.S. PATENT DOCUMENTS
4,953,106 * 8/1990 Gansner et al. 345/440
5,450,535 * 9/1995 Nestib 395/140
5,748,954 5/1998 Mandala 395/610
5,752,241 * 5/1998 Cohen 707/3
5,832,494 * 11/1998 Egger et al. 707/102
5,848,407 * 12/1998 Ishikawa et al. 707/2
6,014,678 * 1/2000 Inoue et al. 707/501

OTHER PUBLICATIONS
S. Jeromy Carricre et al., "Web Query: Searching and Visualizing the Web through Connectivity", Computer Networks and ISDN Systems 29 (1997), pp. 1257-1267.*
Wang et al "Prefetching in World Wide Web", IEEE 1996, pp. 28-32.*
Ramer et al "Similarity, Probability and Database Organization: Extended Abstract", 1996, pp. 272.276.*

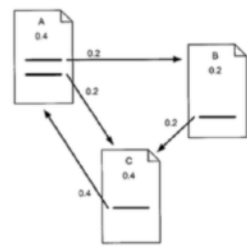
Craig Boyle "To link or not to link: An empirical comparison of Hypertext linking strategies", ACM 1992, pp. 221-231.*
L. Katz, "A new status index derived from sociometric analysis," 1953, Psychometrika, vol. 18, pp. 39-43.
C.H. Hubbell, "An input-output approach to clique identification sociometry," 1965, pp. 377-399.
Mizuchi et al., "Techniques for disaggregating centrality scores in social networks," 1996, Sociological Methodology, pp. 26-48.
E. Garfield, "Citation analysis as a tool in journal evaluation," 1972, Science, vol. 178, pp. 471-479.
Pinski et al., "Citation influence for journal aggregates of scientific publications: Theory, with application to the literature of physics," 1976, Inf. Proc. And Management, vol. 12, pp. 297-312.
N. Geller, "On the citation influence methodology of Pinski and Narin," 1978, Inf. Proc. And Management, vol. 14, pp. 93-95.
P. Doreian, "Measuring the relative standing of disciplinary journals," 1988, Inf. Proc. And Management, vol. 24, pp. 45-56.

(List continued on next page.)

Primary Examiner—Thomas Black
Assistant Examiner—Uyen Le
(74) **Attorney, Agent, or Firm**—Harrity & Snyder L.L.P.

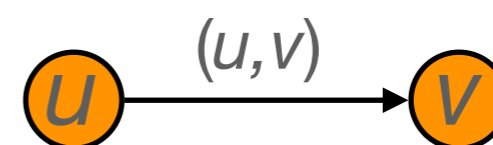
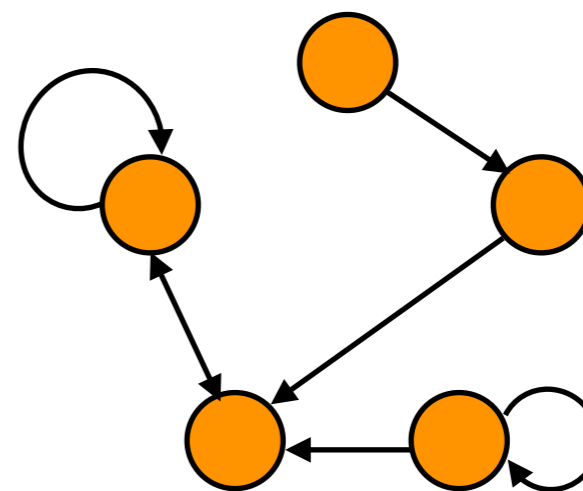
(57) **ABSTRACT**
A method assigns importance ranks to nodes in a linked database, such as any database of documents containing citations, the world wide web or any other hypermedia database. The rank assigned to a document is calculated from the ranks of documents citing it. In addition, the rank of a document is calculated from a constant representing the probability that a browser through the database will randomly jump to the document. The method is particularly useful in enhancing the performance of search engine results for hypermedia databases, such as the world wide web, whose documents have a large variation in quality.

29 Claims, 3 Drawing Sheets



Directed Graphs

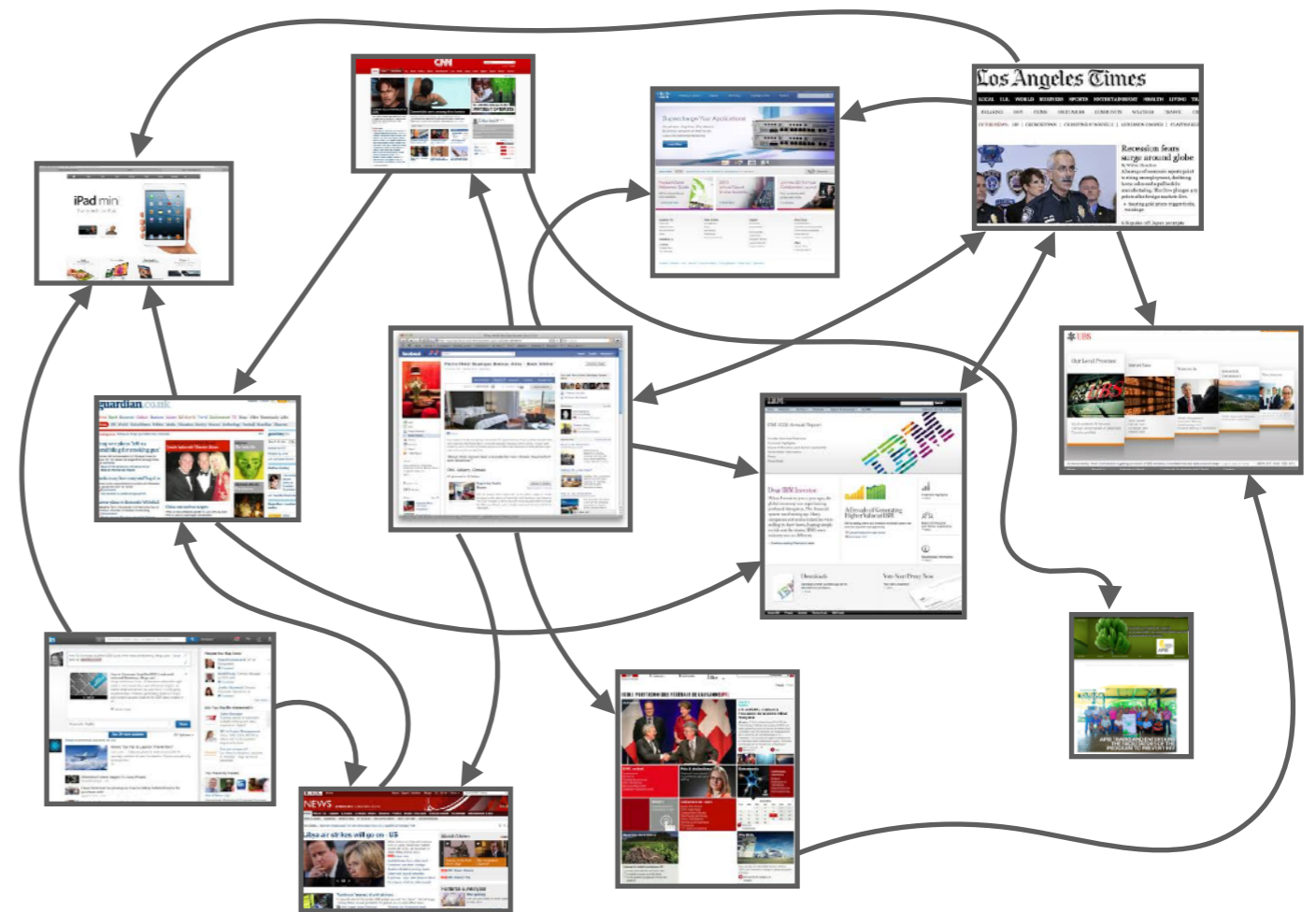
- A *directed graph* is a set V of *vertices* and a set E of *edges*, $E \subset V \times V$.
 - $(u, v) \in E$ connects vertices $u, v \in V$.
 - u is the *starting point* and v the *endpoint* of the edge
- A directed graph on a set V is also called a *relation* on V .



Example of a Directed Graph

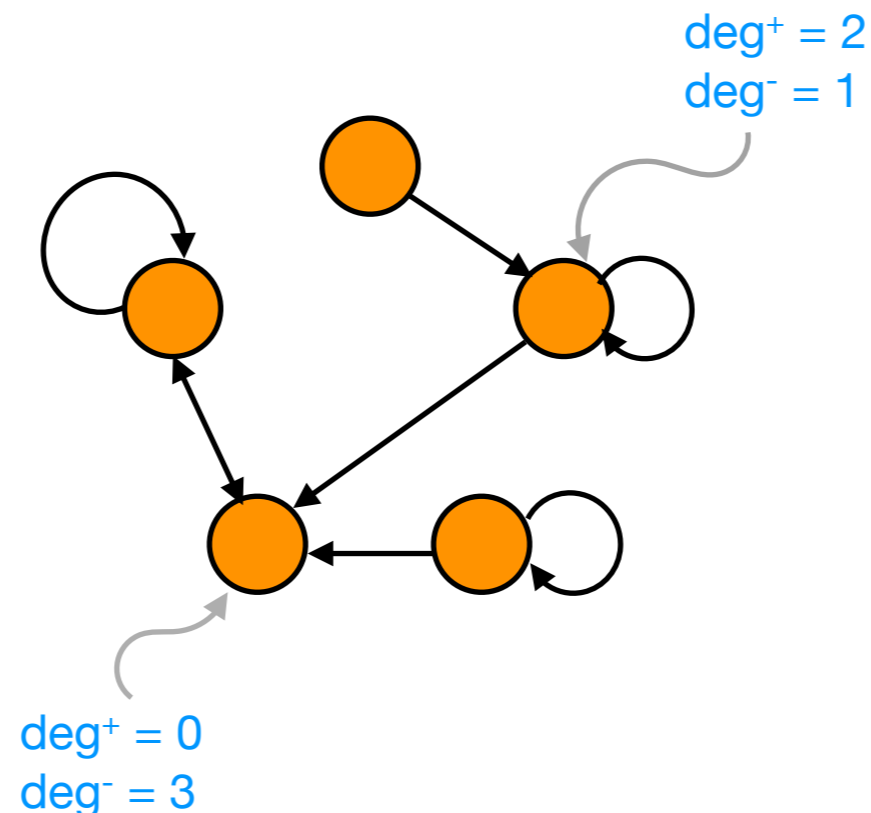
Vertices:
Websites

Relationship:
Directed edge between website A and website B if there is a link from website A to website B



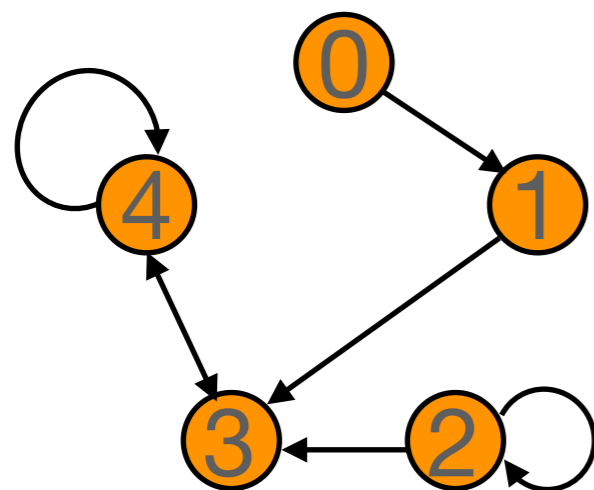
Degrees (again)

- The *in-degree* $\deg^-(v)$ of a node v is the number of edges ending in the node; the *out-degree* $\deg^+(v)$ is the number of edges starting at the node.
- Formally:
 - $\deg^+(u) = |\{(u,v) \in E\}|$
 - $\deg^-(u) = |\{(v,u) \in E\}|$



Adjacency Matrix

- $G = (V, E)$ directed graph, $V = \{v_1, \dots, v_n\}$. An *adjacency matrix* for G is an $n \times n$ -matrix $A = (a_{ij})$ such that
 - $a_{ij} = 1$ if $(v_i, v_j) \in E$, and $a_{ij} = 0$ otherwise.
- Note that the adjacency matrix depends on the ordering of the elements of V (hence is not unique).



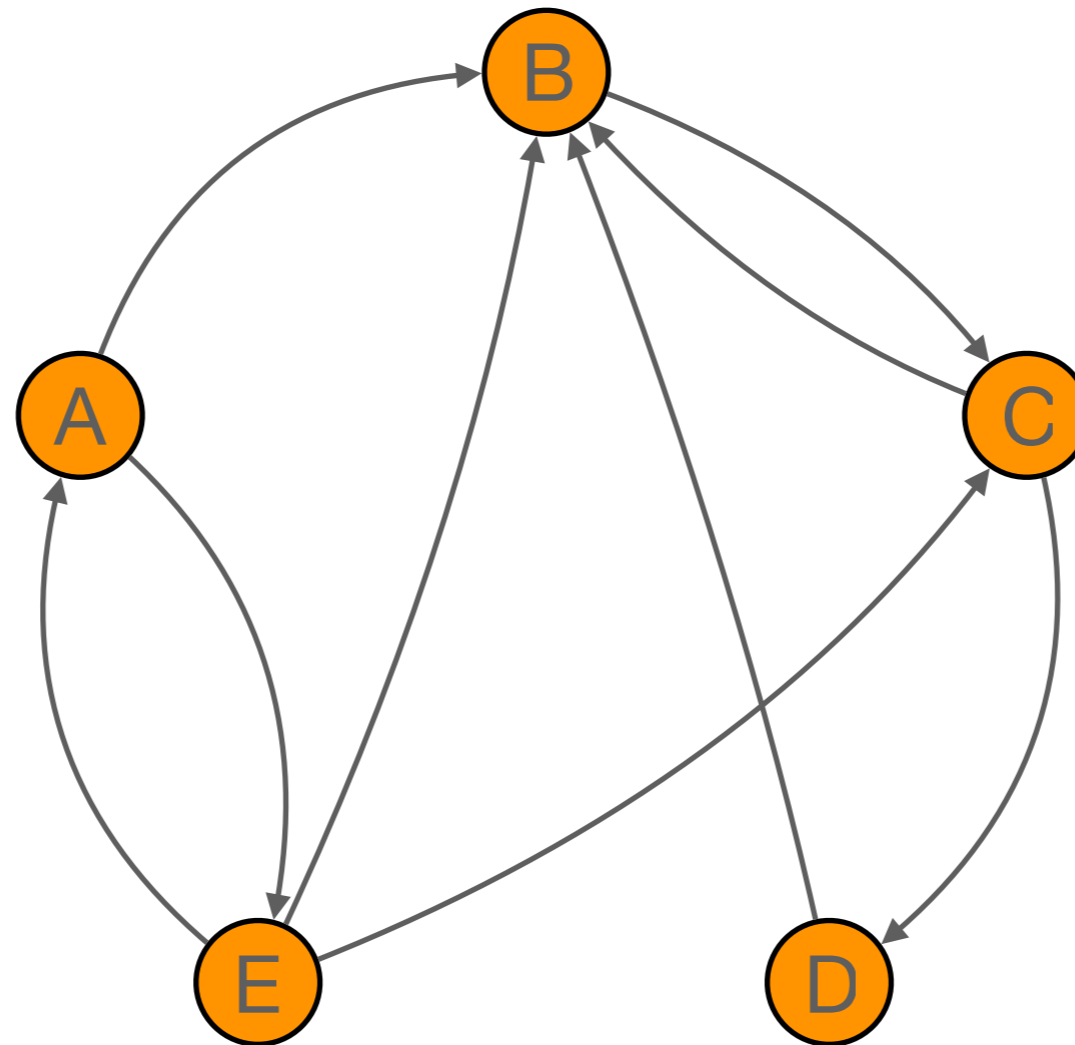
	0	1	2	3	4
0	0	1	0	0	0
1	0	0	0	1	0
2	0	0	1	1	0
3	0	0	0	0	1
4	0	0	0	1	1

Sum of entries in row i is the out-degree of node v_i

Matrix is not symmetric in general

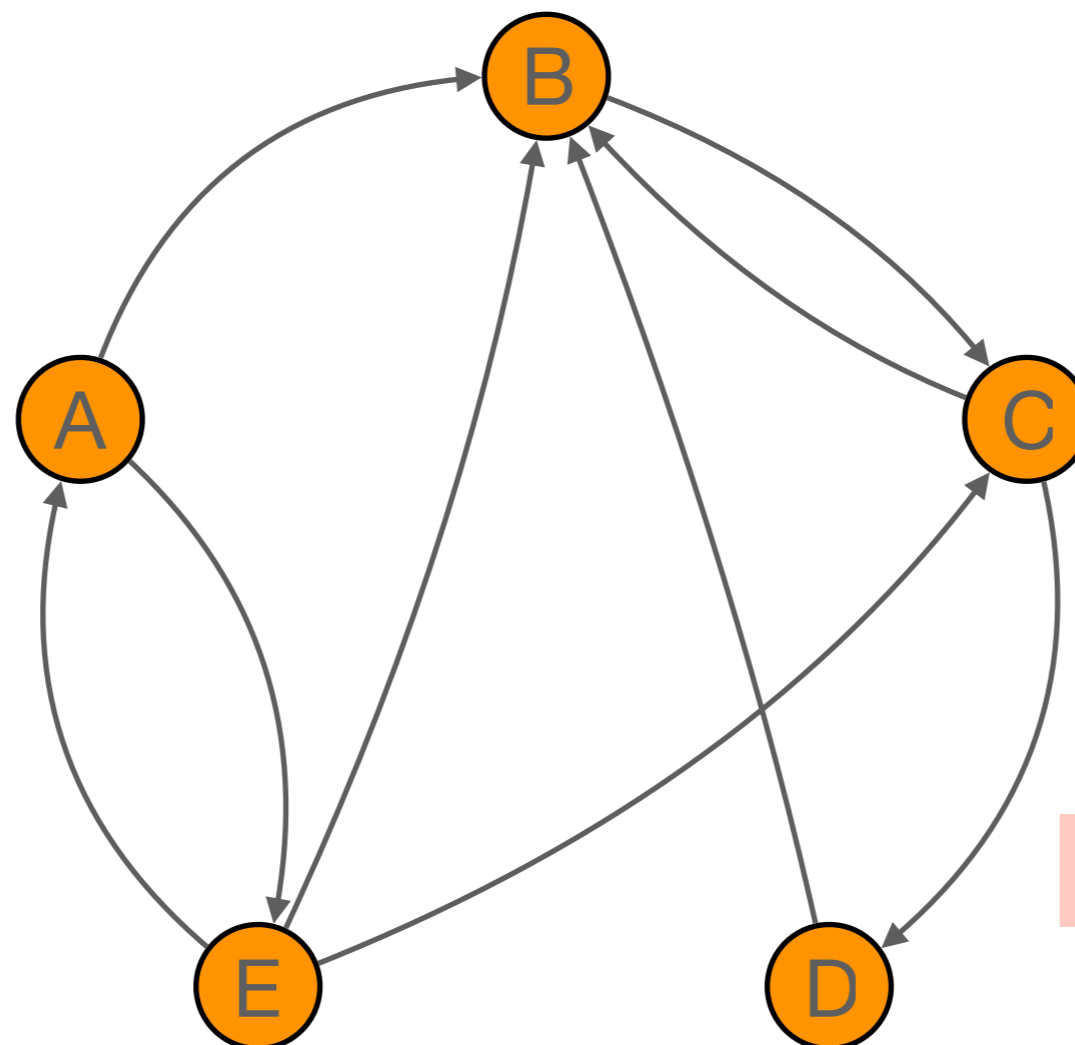
Sum of entries in column i is the in-degree of node v_i

Back to PageRank: Example



First Idea

Use the in-degree as a measure of popularity

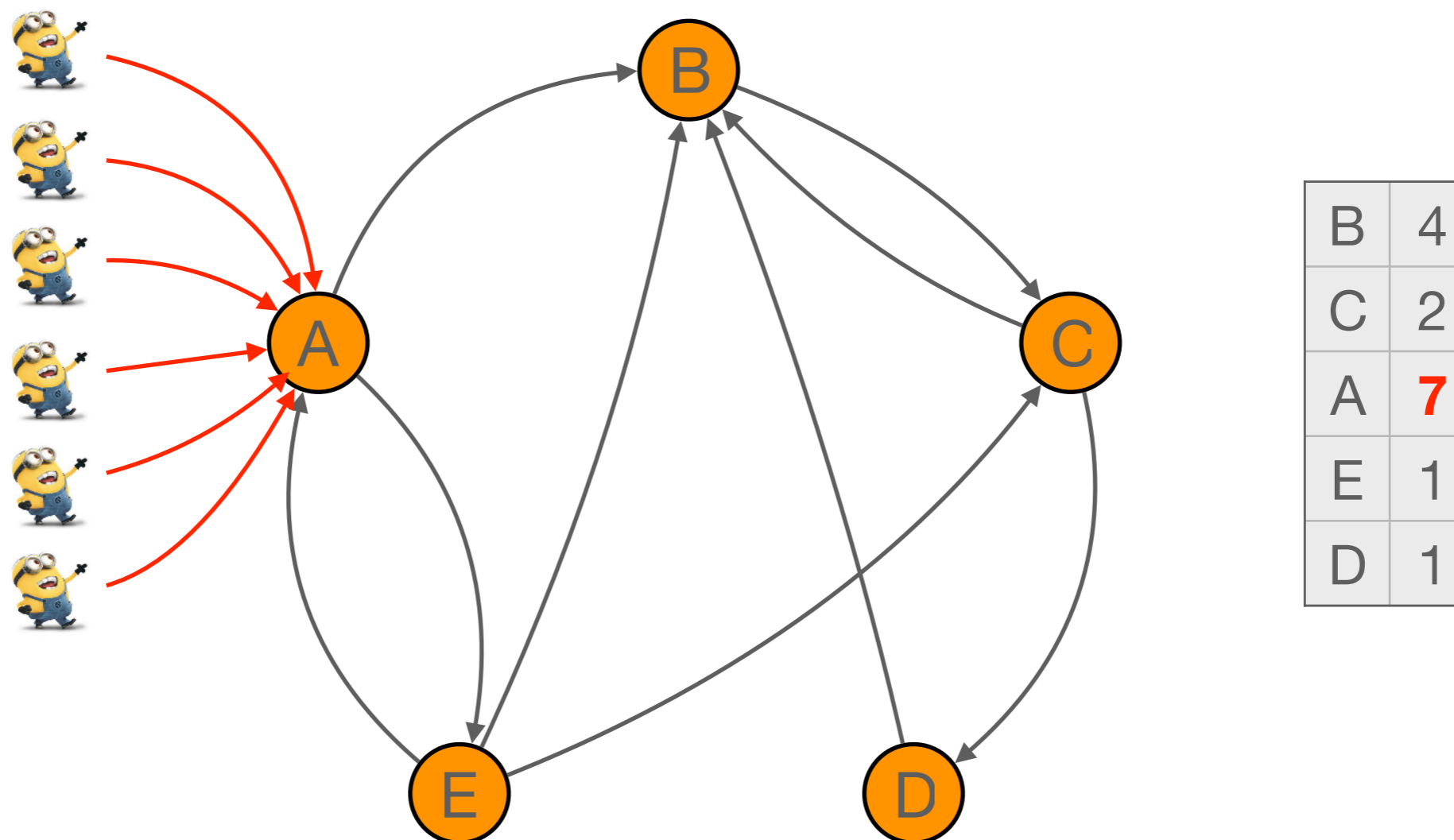


B	4
C	2
A	1
E	1
D	1

B wins the popularity contest

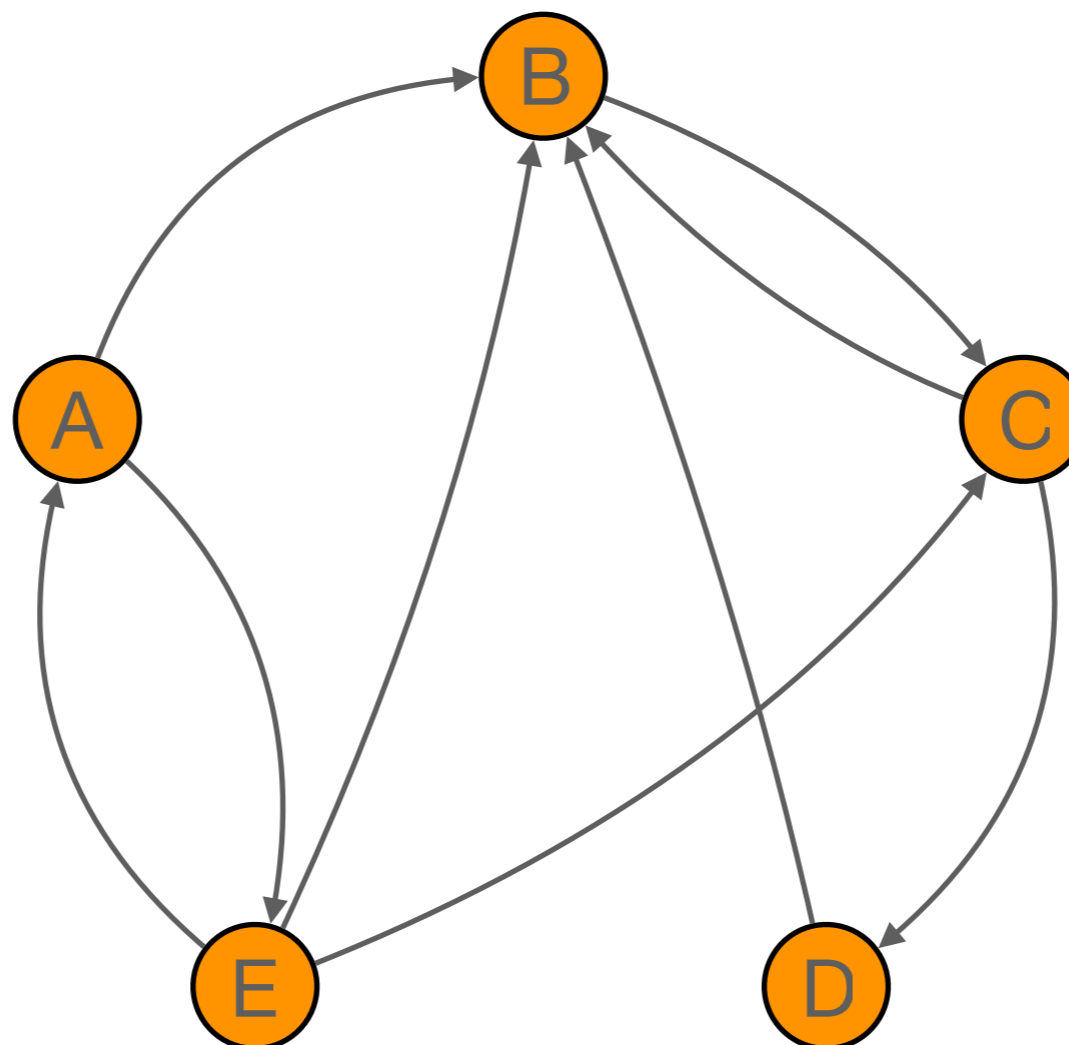
Really that Good?

- No.
- Can be very easily rigged.



Can we do Better?

- But if B is popular, and B is pointing to C, then C should also be popular
- But then D should also be popular, since C is popular and thinks that D is popular as well

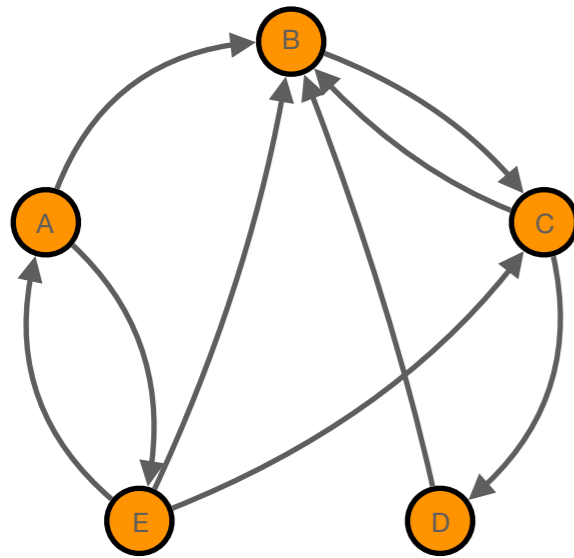


B	4
C	2
A	1
E	1
D	1

A Different Way: Continuous Voting

- Distribute a fixed number of votes to every player at the start

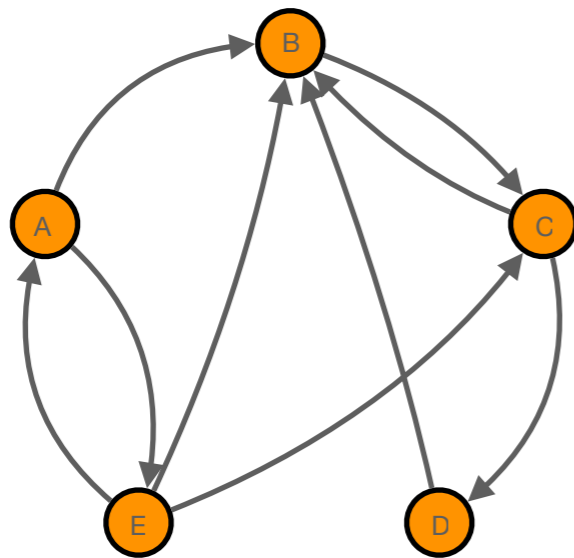
Example



$$\begin{array}{l}
 A \\
 B \\
 C \longrightarrow \\
 D \\
 E
 \end{array}
 \longrightarrow
 \begin{array}{l}
 E/3 \\
 A/2 + C/2 + D + E/3 \\
 B + E/3 \\
 C/2 \\
 A/2
 \end{array}$$

	Round 1	Round 2	Round 3	Round 4	Round 5	Round 6	Round 7	Round 8	Round 9	Round 10
A	0.200	0.067	0.033	0.011	0.006	0.002	0.001	0.000	0.000	0.000
B	0.200	0.467	0.300	0.411	0.417	0.369	0.419	0.395	0.395	0.407
C	0.200	0.267	0.500	0.311	0.417	0.419	0.369	0.420	0.395	0.395
D	0.200	0.100	0.133	0.250	0.156	0.208	0.209	0.185	0.210	0.197
E	0.200	0.100	0.033	0.017	0.006	0.003	0.001	0.000	0.000	0.000

Adjacency Matrix Form



$$\begin{array}{l}
 A \\
 B \\
 C \\
 D \\
 E
 \end{array}
 \longrightarrow
 \begin{array}{l}
 E/3 \\
 A/2+C/2+D+E/3 \\
 B+E/3 \\
 C/2 \\
 A/2
 \end{array}$$

	A	B	C	D	E
A	0	0	0	0	1/3
B	1/2	0	1/2	1	1/3
C	0	1	0	0	1/3
D	0	0	1/2	0	0
E	1/2	0	0	0	0

*

A
B
C
D
E

=

E/3
A/2+C/2+D+E/3
B+E/3
C/2
A/2

Recursion

$$V_0 = \begin{bmatrix} 0.2 \\ 0.2 \\ 0.2 \\ 0.2 \\ 0.2 \end{bmatrix} \quad V_{k+1} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1/3 \\ 1/2 & 0 & 1/2 & 1 & 1/3 \\ 0 & 1 & 0 & 0 & 1/3 \\ 0 & 0 & 1/2 & 0 & 0 \\ 1/2 & 0 & 0 & 0 & 0 \end{bmatrix} * V_k =: A * V_k$$

$$V_k = A^k * V_0$$

Does this recursion converge to a fixed point?

Diagonalization

$$A = \begin{pmatrix} 0 & 0 & 0 & 0 & 1/3 \\ 1/2 & 0 & 1/2 & 1 & 1/3 \\ 0 & 1 & 0 & 0 & 1/3 \\ 0 & 0 & 1/2 & 0 & 0 \\ 1/2 & 0 & 0 & 0 & 0 \end{pmatrix} = T \cdot \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & -(1+i)/2 & 0 & 0 & 0 \\ 0 & 0 & (-1+i)/2 & 0 & 0 \\ 0 & 0 & 0 & -\sqrt{6}/6 & 0 \\ 0 & 0 & 0 & 0 & \sqrt{6}/6 \end{pmatrix} \cdot T^{-1}$$

Absolute value of these eigenvalues is < 1.

$$A^k \rightarrow T \cdot \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \cdot T^{-1}$$

$$V_k = A^k * V_0$$

Recursion converges!

How to Find the Solution

Fixed point w satisfies

$$w = A * w$$

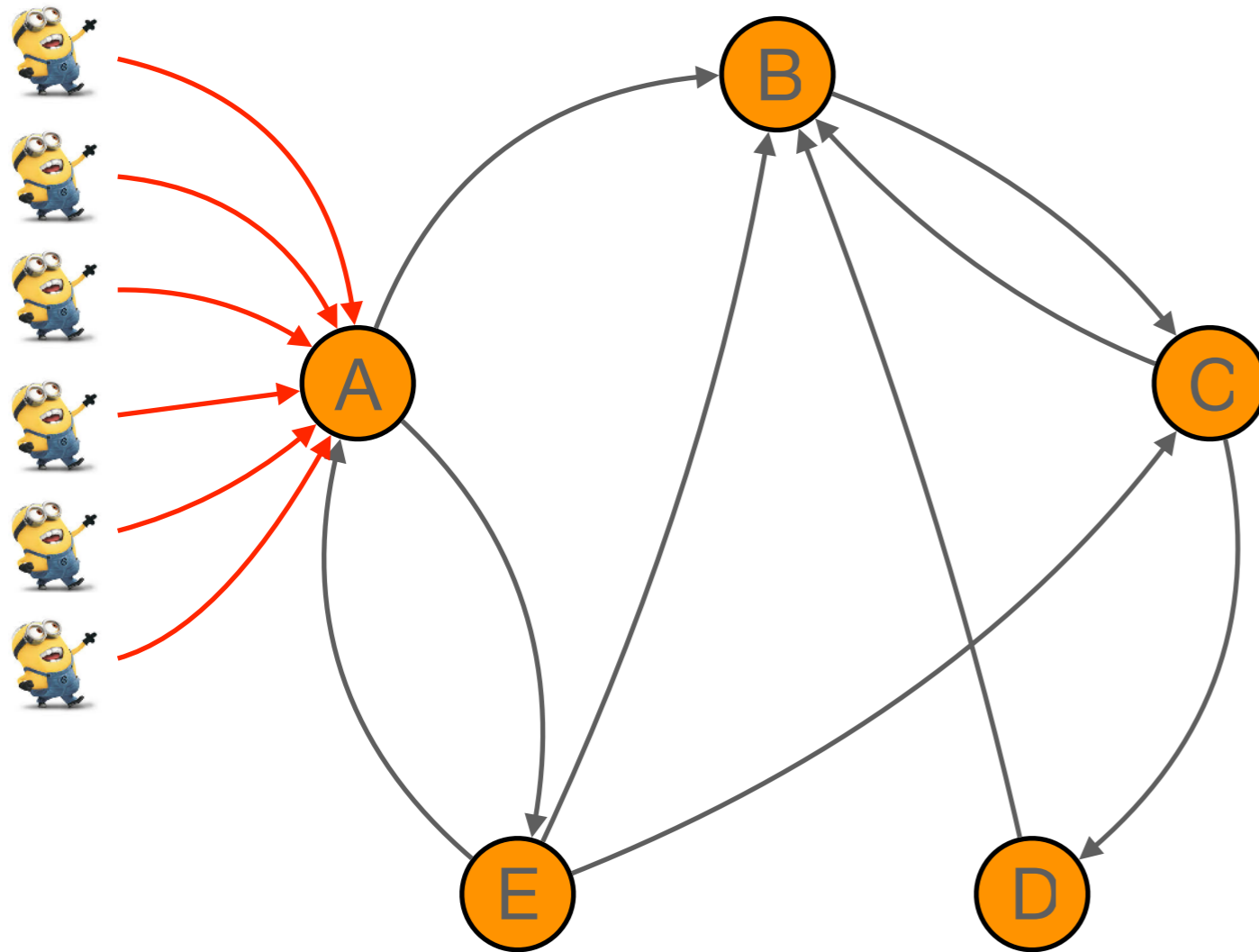
So, w is an eigenvector with eigenvalue 1 =

Vector unique subject to sum of entries = 1

0
0.4
0.4
0.2
0

	Round 1	Round 2	Round 3	Round 4	Round 5	Round 6	Round 7	Round 8	Round 9	Round 10
A	0.200	0.067	0.033	0.011	0.006	0.002	0.001	0.000	0.000	0.000
B	0.200	0.467	0.300	0.411	0.417	0.369	0.419	0.395	0.395	0.407
C	0.200	0.267	0.500	0.311	0.417	0.419	0.369	0.420	0.395	0.395
D	0.200	0.100	0.133	0.250	0.156	0.208	0.209	0.185	0.210	0.197
E	0.200	0.100	0.033	0.017	0.006	0.003	0.001	0.000	0.000	0.000

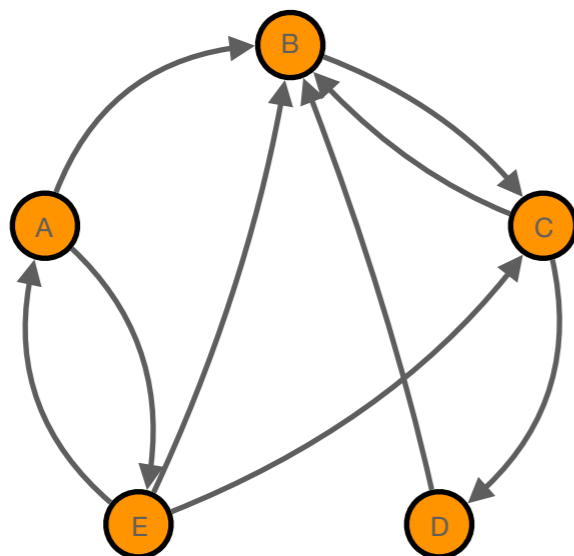
Rigging



	A	B	C	D	E						
A	0	0	0	0	1/3	1	1	1	1	1	1
B	1/2	0	1/2	1	1/3	0	0	0	0	0	0
C	0	1	0	0	1/3	0	0	0	0	0	0
D	0	0	1/2	0	0	0	0	0	0	0	0
E	1/2	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0

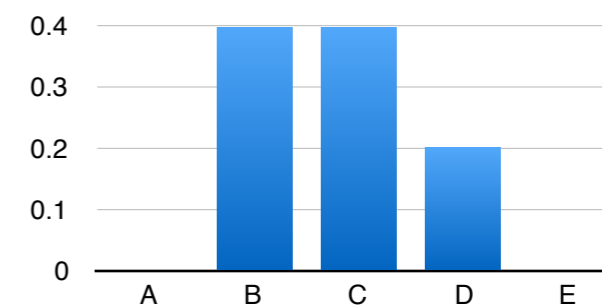
- Same eigenvector for eigenvalue 1
- Rigging would not work

Cooperative Rigging

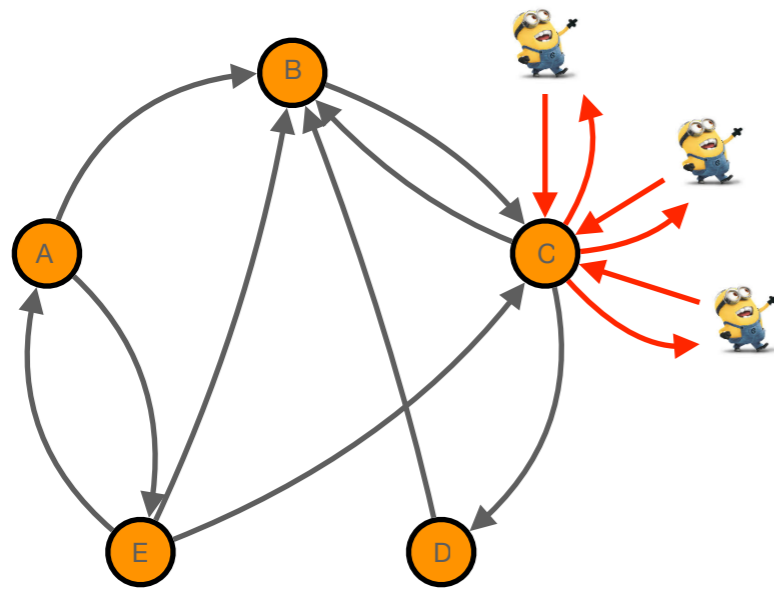


	A	B	C	D	E
A	0	0	0	0	1/3
B	1/2	0	1/5	1	1/3
C	0	1	0	0	1/3
D	0	0	1/5	0	0
E	1/2	0	0	0	0

A	B	C	D	E
0	2/5	2/5	1/5	0
	40%	40%	20%	

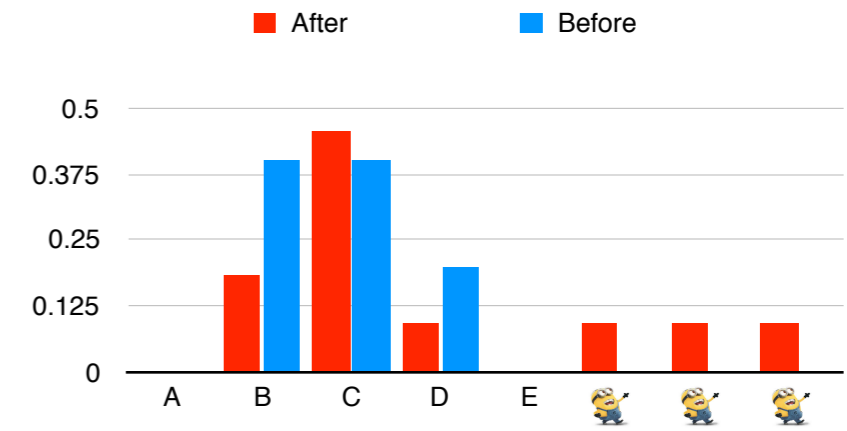


Cooperative Rigging



	A	B	C	D	E	Minion 1	Minion 2	Minion 3
A	0	0	0	0	1/3	0	0	0
B	1/2	0	1/5	1	1/3	0	0	0
C	0	1	0	0	1/3	1	1	1
D	0	0	1/5	0	0	0	0	0
E	1/2	0	0	0	0	0	0	0
Minion 1	0	0	1/5	0	0	0	0	0
Minion 2	0	0	1/5	0	0	0	0	0
Minion 3	0	0	1/5	0	0	0	0	0

A	B	C	D	E	Minion 1	Minion 2	Minion 3
0	2/11	5/11	1/11	0	1/11	1/11	1/11
	18.18%	45.46%	9.1%		9.1%	9.1%	9.1%



Perron-Frobenius Theorem

- Theorem about the eigenvectors and eigenvalues of “non-negative” matrices
 - First proved by Perron for “positive” matrices in 1907
 - ➔ Matrices having strictly positive entries
 - Later generalized by Frobenius to non-negative matrices of a particular type in 1912
 - ➔ Matrices having non-negative entries
 - ➔ Such that the underlying directed graph is strongly connected



Oskar Perron
1880-1975



Ferdinand Georg Frobenius
1849-1917

Definitions

- A matrix is called *non-negative* if all of its entries are ≥ 0
- A matrix is called *irreducible* if for any of its entries (i,j) there is a k such that the (i,j) -entry of A^k is positive.
 - This means that the underlying directed graph is *strongly connected*
 - ➔ This means that for any two nodes in the graph there is a directed path connecting them

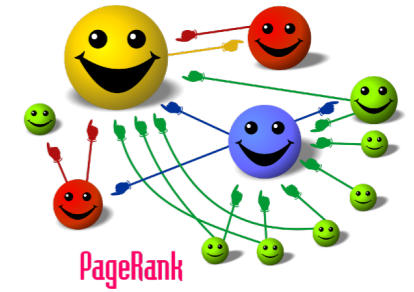
Perron-Frobenius Theorem (Abridged Version)

- A non-negative irreducible matrix
- Then A has a positive (real) eigenvalue λ_{\max} and for all other eigenvalues λ we have

$$|\lambda| \leq \lambda_{\max}$$

- Moreover, if the sum of the entries of the columns of A is 1 for every column, then $\lambda_{\max} = 1$
 - ➔ This last part is a corollary and not really a part of the theorem
- The theorem can be used to prove convergence of the iteration
 - Caveat: the matrices we obtain are not always irreducible

PageRank



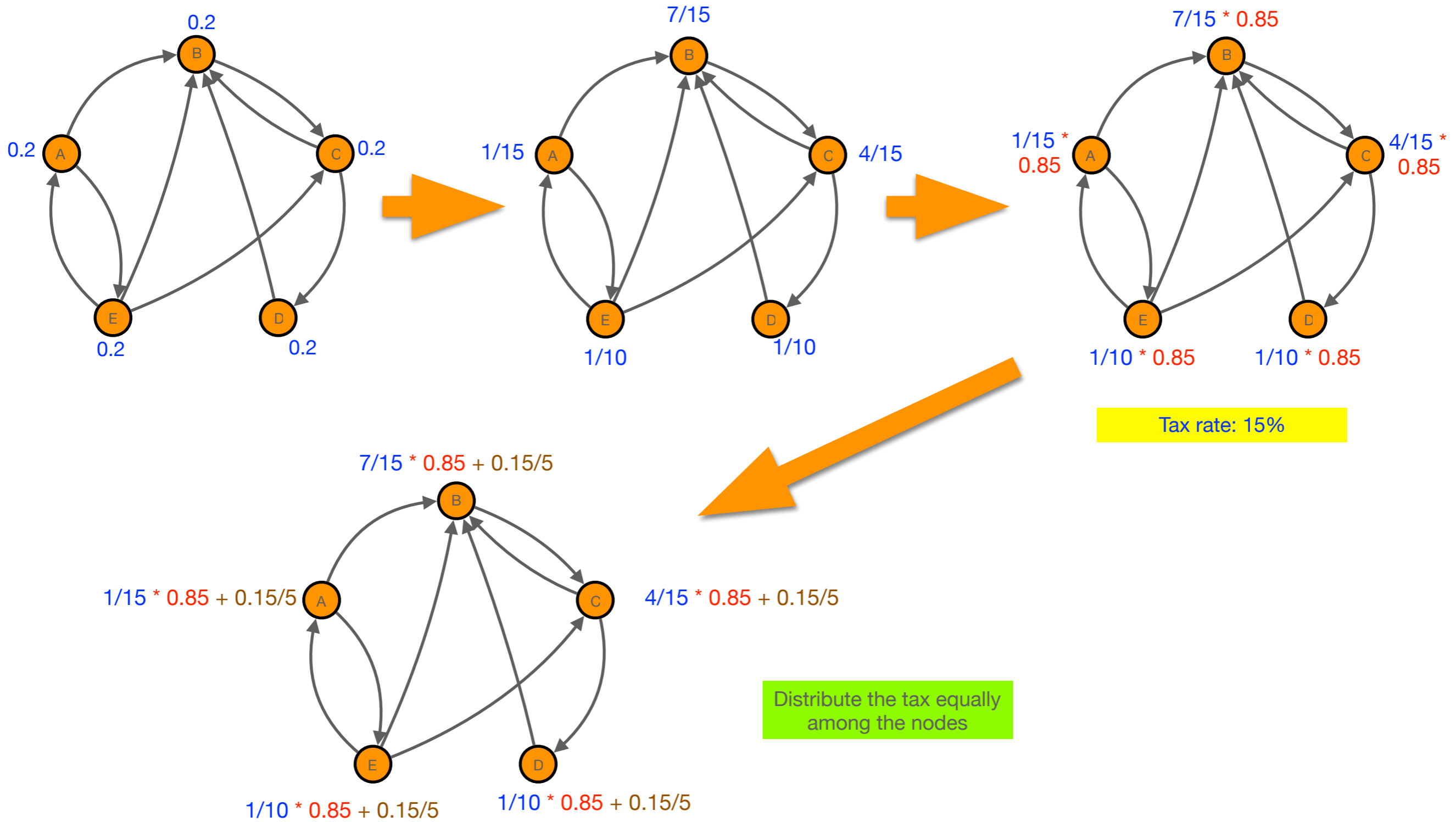
- Basic Idea: Taxation

- Imagine the votes being money transferred from one node to another
- At every iteration, the amount of money at each node is taxed at the rate of $t < 1$.
- The money raised this way is equally distributed among all the nodes in the graph for the next iteration.

PageRank

- What does it mean for websites?
 - For websites: if people start clicking on outgoing links, then at each stage they have a certain probability of getting bored and moving to another random webpage
 - ➔ Typical tax rate is 15%
- What does it mean for payments or votes?
 - Through taxation, even unpopular members can have some chance of survival
 - ➔ Tax rate should depend on the preferred outcome

PageRank



PageRank: Mathematical Formulation

N = total number of nodes

$$v_0 = \begin{pmatrix} 1/N \\ 1/N \\ \vdots \\ 1/N \end{pmatrix}$$

At the beginning:
All nodes receive equal votes

$$v_{k+1} = (1 - t)A \cdot v_k + \begin{pmatrix} t/N \\ t/N \\ \vdots \\ t/N \end{pmatrix}$$

Tax rate: t

Distribute the tax equally
among the nodes

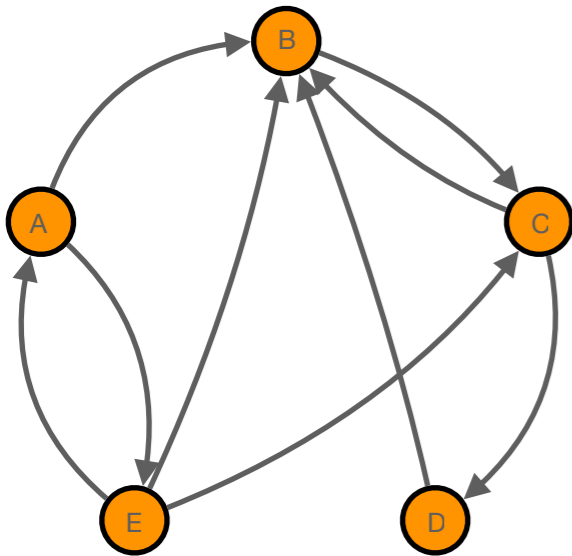
Fixed Point w

$$w = (1 - t)Aw + tv_0$$

$$w = t(I - (1 - t)A)^{-1} \cdot v_0$$

Convergence guaranteed by Perron-Frobenius

Example



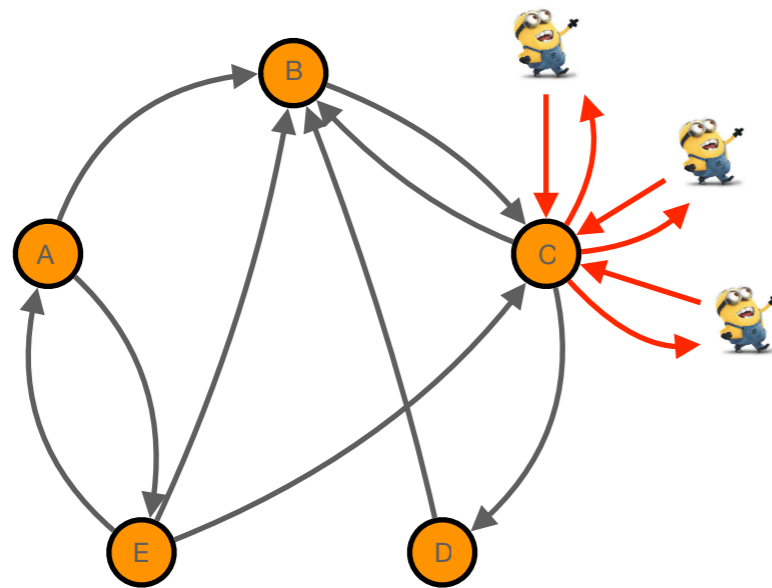
$$A = \begin{bmatrix} 0 & 0 & 0 & 0 & 1/3 \\ 1/2 & 0 & 1/2 & 1 & 1/3 \\ 0 & 1 & 0 & 0 & 1/3 \\ 0 & 0 & 1/2 & 0 & 0 \\ 1/2 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$(I - 0.85 * A)^{-1} = \begin{bmatrix} 1.13690194220748 & -0. & -0. & -0. & 0.322122216958787 \\ 2.19400616455640 & 3.01488599962314 & 2.37045411720369 & 2.56265309967967 & 2.14748144639191 \\ 2.00180718208042 & 2.56265309967967 & 3.01488599962314 & 2.17825513472772 & 2.14748144639191 \\ 0.850768052384179 & 1.08912756736386 & 1.28132654983983 & 1.92575843225928 & 0.912679614716564 \\ 0.483183325438181 & 0. & 0. & 0. & 1.13690194220748 \end{bmatrix}$$

$$W = \begin{array}{c} \begin{array}{ccccc} \text{A} & \text{E} & \text{C} & \text{D} & \text{E} \end{array} \\ \begin{bmatrix} 0.0438 & 0.3687 & 0.3572 & 0.1818 & 0.0486 \end{bmatrix} \end{array}$$



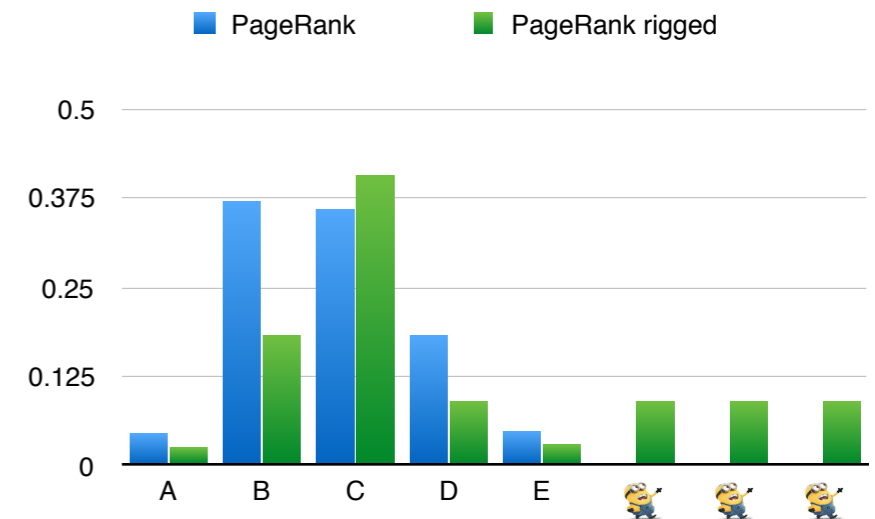
Rigging



$A =$

0	0	0	0	1/3	0	0	0
1/2	0	1/5	1	1/3	0	0	0
0	1	0	0	1/3	1	1	1
0	0	1/5	0	0	0	0	0
1/2	0	0	0	0	0	0	0
0	0	1/5	0	0	0	0	0
0	0	1/5	0	0	0	0	0
0	0	1/5	0	0	0	0	0

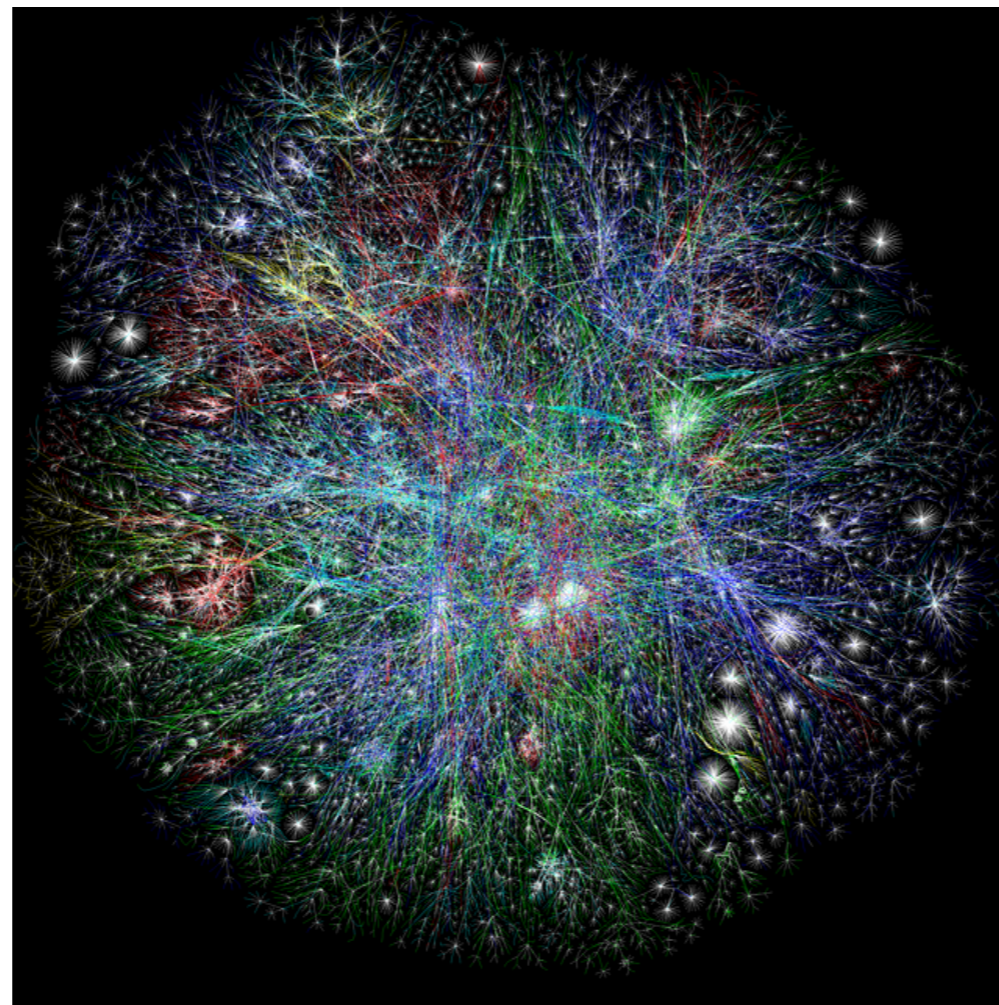
0.0274	0.1830	0.4073	0.0880	0.0304	0.0880	0.0880	0.0880



Rigging is still possible

Rigging

- Cooperative rigging becomes exceedingly difficult (but not impossible) as the graph grows
 - Only a small part of the graph is modified
- but other countermeasures are needed



Implementation

- In reality, we don't compute eigenvectors of matrices or their inverses
- Computation is done via “simulation” or “iteration”
- If the eigenvalues of the matrix are small, then iteration can converge quickly to desired solution