Social Network Analysis

Information Networks and the World Wide Web

Course Outline

- Graph Theory and Social Networks
- Visualizing Social Networks
- Game Theory
- **Information Networks and the World Wide Web**
- Network Dynamics
- Applications of SNA in various domains

Information Network & WWW

- Information Network
- History of WWW
- Structure of WWW
- Web 2.0

Information Network

- Similarity with Social Networks:
  - Both graphs
  - Have some similar properties (Connected Components, Power Laws)
- Differences with Social Network:
  - Nodes represent information, not people
  - Links mostly one-directional
- Which Social Networks are mostly unidirectional?
  - Name Recognition Network
  - Twitter Follower Network

History of WWW

- Information Networks have existed before:
  - Encyclopedia
  - Citation Network
    - Papers static
    - Links Unidirectional

- Precursor of WWW: *Hypertext*
  - Since middle of the twentieth century
  - Replace the traditional linear structure of text with a network structure, in which any portion of the text can link directly to any other part.

Memex

- Vannevar Bush and his seminal 1945 article in the Atlantic Monthly entitled "*As We May Think*"
- Traditional methods for storing information in a book, a library, or a computer memory are highly linear — they consist of a collection of items sorted in some sequential order.
- Our conscious experience of thinking, on the other hand, exhibits associative memory like a semantic network represents
  - You think of one thing; it reminds you of another; you see a novel connection; some new insight is formed.
- Bush called for the creation of information systems that mimicked this style of memory Memex
  - Functioned very much like the Web, consisting of digitized versions of all human knowledge connected by associative links
- Bush's article foreshadowed not only the Web itself, but also many of the dominant metaphors that are now used to think about the Web: the Web as universal encyclopedia; the Web as giant socio-economic system; the Web as global brain.
- Vannever Bush's vision was so accurate is not in any sense coincidental:
  - Bush occupied a prominent position in the U.S. government's scientific funding establishment
- Tim Berners-Lee invoked Bush's ideas when he set out to develop the Web

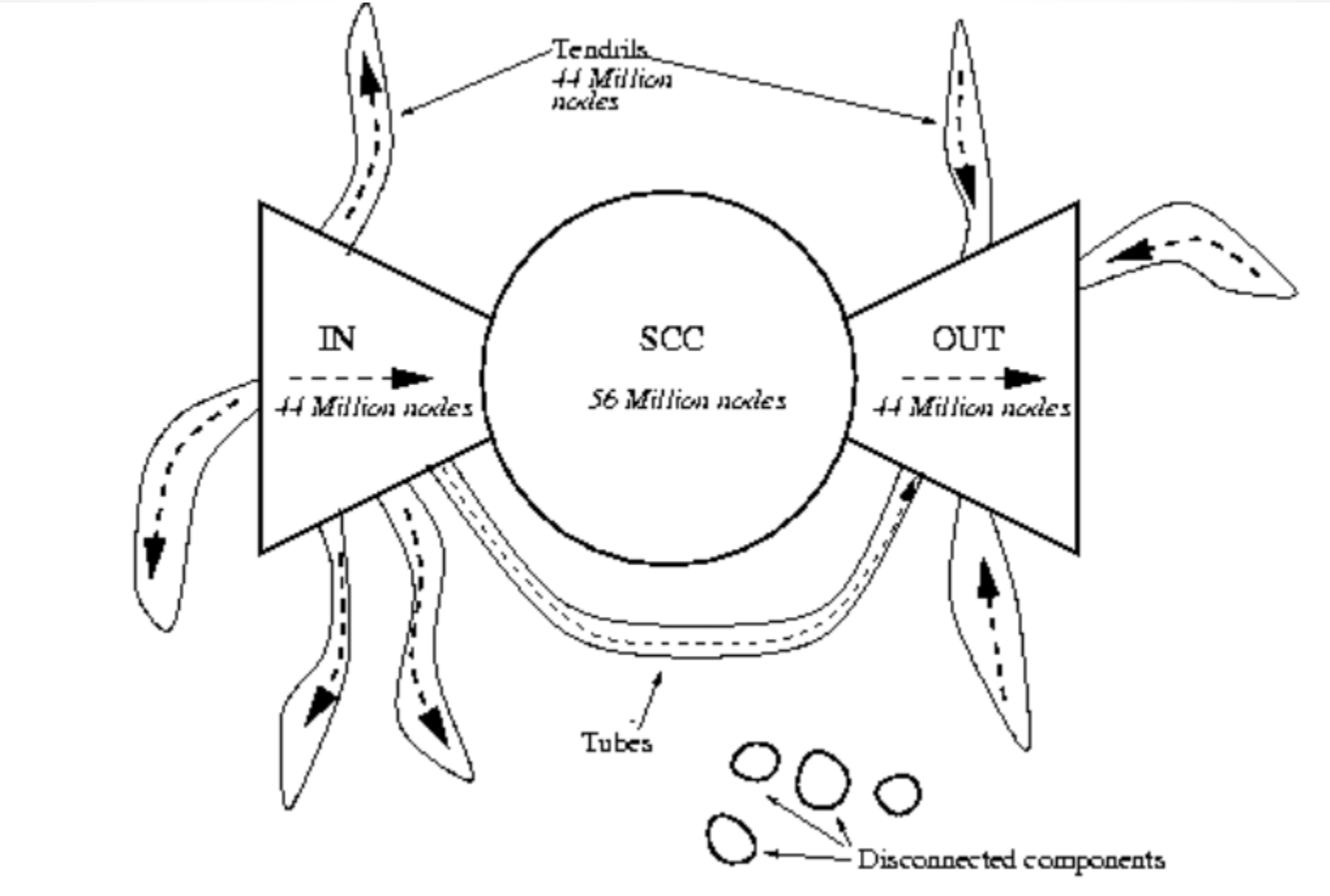
Strongly Connected Component

- (SCC) in a directed graph is a subset of the nodes such that:
  1. every node in the subset has a path to every other
  2. the subset is not part of some larger set with the property that every node can reach every other.
- Web contains a giant strongly connected component
  - Many naturally occurring undirected graphs have a giant connected component
  - Major "directory" sites -> home pages of major educational institutions, large companies, and governmental agencies -> pages in these sites ->Directory Pages
  - Thus, all these pages can mutually reach one another, and hence all belong to the same strongly connected component.
  - This SCC contains (at least) the home pages of many of the major commercial, governmental, and non-profit organizations in the world, it is easy to believe that it is a giant SCC.
  - If there were two giant SCCs —X & Y a single link from any node in X to any node Y and another link from any node in Y to any node in X, X and Y would merge

Structure of the Web

- Andrei Broder et al
  - Andrei Broder, Ravi Kumar, Farzin Maghoul, Prabhakar Raghavan, Sridhar Rajagopalan, Raymie Stata, Andrew Tomkins, and Janet Wiener. Graph structure in the Web. In Proc. 9th International World Wide Web Conference, pages 309–320, 2000
- Crawl of Alta Vista Search Engine
- Only Navigational Pages
- The influential study has since been replicated on other, even larger snapshots of the Web:
  - An early index of Google's search engine
  - Large research collections of Web pages
- Similar analyses have been carried out for particular well-defined pieces of the Web:
  - The links among articles on Wikipedia
  - Complex directed graph structures arising in other domains, such as the network of interbank loans

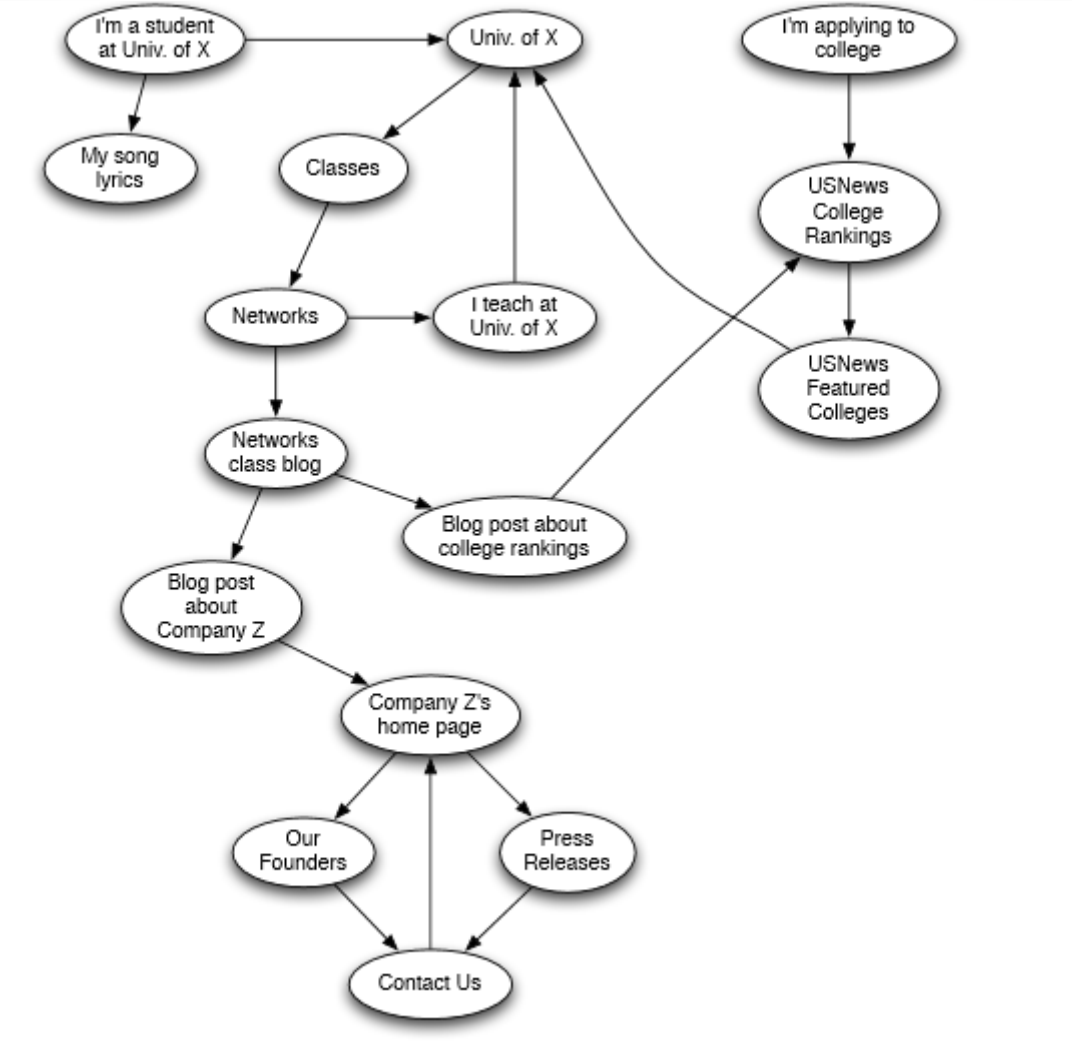
Bow-tie Structure of Web



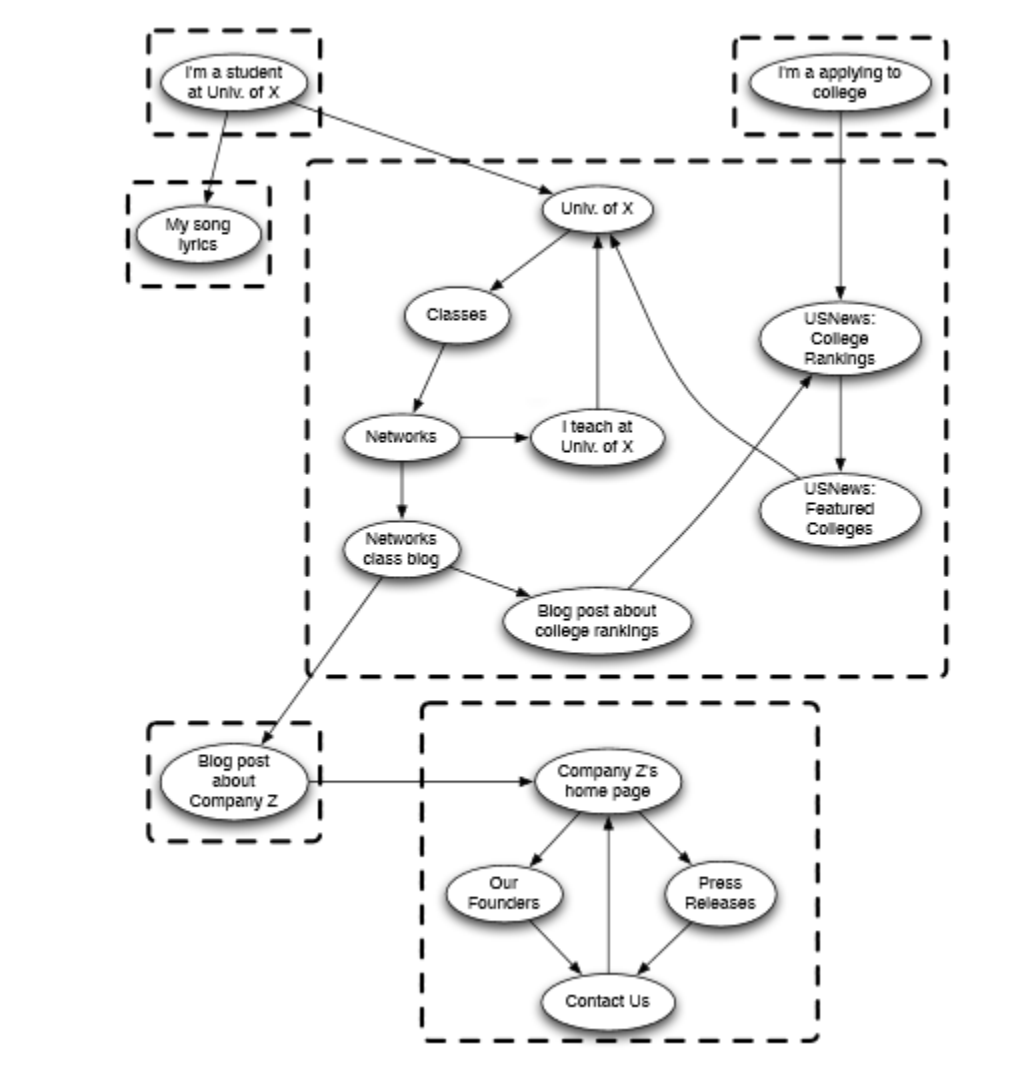
Components in the Bow-tie Structure

- IN: nodes that can reach the giant SCC but cannot be reached from it
- OUT: nodes that can be reached from the giant SCC but cannot reach it
- Tendrils: The "tendrils" of the bow-tie consist of
  a. The nodes reachable from IN that cannot reach the giant SCC,
  b. The nodes that can reach OUT but cannot be reached from the giant SCC. SCC
- Tube: A node that satisfy both (a) and (b) - travels from IN to OUT without touching the giant SCC
- Disconnected: Nodes that would not have a path to the giant SCC even if we completely ignored the directions of the edges
- Structure is dynamic

Exercise – Find Structure



SCCs


I'm a student at Univ. of X
I'm a applying to college
My song lyrics
Univ. of X
Classes
USNews: College Rankings
Networks
I teach at Univ. of X
USNews: Featured Colleges
Networks class blog
Blog post about college rankings
Blog post about Company Z
Company Z's home page
Our Founders
Press Releases
Contact Us

Web 2.0

- In 2000s changes in Web:
  I. The growth of Web authoring styles that enabled many people to collectively create and maintain shared content;
  II. the movement of people's personal on-line data (including e-mail, calendars, photos, and videos) from their own computers to services offered and hosted by large companies;
  III. the growth of linking styles that emphasize on-line connections between people, not just between documents
- Showcases several "social phenomena":
  - Software that gets better the more people use It
  - The wisdom of crowd
  - The Long Tail

Web Search Engines

- Information Retrieval – Fundamentals
- Problems of Web Search
- Crawling
- Hubs & Authorities
- Page Rank
- Beyond Page Rank
  - HillTop Algorithm
  - Trust Rank
- Link Analysis in other scenarios

Query-document matching scores

- We need a way of assigning a score to a query/document pair
- Let's start with a one-term query
- If the query term does not occur in the document: score should be 0
- The more frequent the query term in the document, the higher the score (should be)

Summary: tf x idf (or tf.idf)

- Assign a tf.idf weight to each term *i* in each document *d*

$$w_{t,d} = tf_{t,d} \times \log(N / df_t)$$

$tf_{t,d}$ = frequency of term t in document d

N = total number of documents

df_t = the number of documents that contain term t

- Weight increases with the number of occurrences *within* a doc
- And increases with the rarity of the term *across* the whole corpus

Term-document count matrices

- Consider the number of occurrences of a term in a document:
  - Each document is a count vector in $\mathbb{N}^{v}$: a column below

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	157	73	0	0	0	0
Brutus	4	157	0	1	0	0
Caesar	232	227	0	2	1	1
Calpurnia	0	10	0	0	0	0
Cleopatra	57	0	0	0	0	0
mercy	2	0	3	5	5	1
worser	2	0	1	1	1	0

Bag of words model

- Vector representation doesn't consider the ordering of words in a document
- *John is quicker than Mary* and *Mary is quicker than John* have the same vectors
- This is called the bag of words model.
- Limitation: The positional index was able to distinguish these two documents.

Term frequency tf

- The term frequency $tf_{t,d}$ of term *t* in document *d* is defined as the number of times that *t* occurs in *d*.
- We want to use tf when computing query-document match scores. But how?
- Raw term frequency is not what we want:
  - A document with 10 occurrences of the term is more relevant than a document with 1 occurrence of the term.
  - But not 10 times more relevant.
- Relevance does not increase proportionally with term frequency.

NB: frequency = count in IR

Log-frequency weighting

- The log frequency weight of term t in d is

$$w_{t,d} = \begin{cases} 1 + \log_{10} \mathrm{tf}_{t,d}, & \text{if } \mathrm{tf}_{t,d} > 0 \\ 0, & \text{otherwise} \end{cases}$$

- 0 → 0, 1 → 1, 2 → 1.3, 10 → 2, 1000 → 4, etc.
- Score for a document-query pair: sum over terms *t* in both *q* and *d*:
- score

$$= \sum_{t \in q \cap d} (1 + \log \mathrm{tf}_{t,d})$$

- The score is 0 if none of the query terms is present in the document.

Document frequency

- Rare terms are more informative than frequent terms
  - Recall stop words
- Consider a term in the query that is rare in the collection (e.g., *arachnocentric*)
- A document containing this term is very likely to be relevant to the query *arachnocentric*
- → We want a high weight for rare terms like *arachnocentric*.

Document frequency, continued

- Frequent terms are less informative than rare terms
- Consider a query term that is frequent in the collection (e.g., *high, increase, line*)
- A document containing such a term is more likely to be relevant than a document that doesn't
- But it's not a sure indicator of relevance.
- → For frequent terms, we want high positive weights for words like *high, increase, and line*
- But lower weights than for rare terms.
- We will use document frequency (df) to capture this.

idf weight

- df_t is the document frequency of *t*: the number of documents that contain *t*
  - df_t is an inverse measure of the informativeness of *t*
  - $df_t \leq N$
- We define the idf (inverse document frequency) of *t* by
  - We use log (N/df_t) instead of N/df_t to "dampen" the effect of idf.

$$\text{idf}_t = \log_{10}(N/\text{df}_t)$$

idf example, suppose N = 1 million

term	df_t	idf_t
calpurnia	1	
animal	100	
sunday	1,000	
fly	10,000	
under	100,000	
the	1,000,000	

$$\text{idf}_t = \log_{10}(N/\text{df}_t)$$

There is one idf value for each term t in a collection.

Effect of idf on ranking

- Does idf have an effect on ranking for one-term queries, like
  - iPhone
- idf has no effect on ranking one term queries
  - idf affects the ranking of documents for queries with at least two terms
  - For the query capricious person, idf weighting makes occurrences of capricious count for much more in the final document ranking than occurrences of person.

tf-idf weighting

- The tf-idf weight of a term is the product of its tf weight and its idf weight.

$$w_{t,d} = \log(1 + \mathrm{tf}_{t,d}) \times \log_{10}(N / \mathrm{df}_t)$$

- Best known weighting scheme in information retrieval
  - Note: the "-" in tf-idf is a hyphen, not a minus sign!
  - Alternative names: tf.idf, tf x idf
- Increases with the number of occurrences within a document
- Increases with the rarity of the term in the collection

Documents as vectors

- So we have a |V|-dimensional vector space
- Terms are axes of the space
- Documents are points or vectors in this space
- Very high-dimensional: tens of millions of dimensions when you apply this to a web search engine
- These are very sparse vectors - most entries are zero.

Queries as vectors

- Key idea 1: Do the same for queries: represent them as vectors in the space
- Key idea 2: Rank documents according to their proximity to the query in this space
- proximity = similarity of vectors
- proximity ≈ inverse of distance
- Recall: We do this because we want to get away from the you're-either-in-or-out Boolean model.
- Instead: rank more relevant documents higher than less relevant documents

The Vector Space Model

Queries are just short documents

- Take the freetext query as short document
- Return the documents ranked by the closeness of their vectors to the query vector.

Formalizing vector space proximity

- First cut: distance between two points
  - (= distance between the end points of the two vectors)
- Euclidean distance?
- Euclidean distance is a bad idea . . .
- . . . because Euclidean distance is large for vectors of different lengths.

Use angle instead of distance

- Thought experiment: take a document *d* and append it to itself. Call this document *d′*.
- "Semantically" d and d′ have the same content
- The Euclidean distance between the two documents can be quite large
- The angle between the two documents is 0, corresponding to maximal similarity.

- Key idea: Rank documents according to angle with query.

From angles to cosines

- The following two notions are equivalent.
  - Rank documents in decreasing order of the angle between query and document
  - Rank documents in increasing order of cosine(query,document)
- Cosine is a monotonically decreasing function for the interval [0°, 180°]

Length normalization

- A vector can be (length-) normalized by dividing each of its components by its length – for this we use the L_2 norm:

$$\|\vec{x}\|_2 = \sqrt{\sum_i x_i^2}$$

- Dividing a vector by its L_2 norm makes it a unit (length) vector (on surface of unit hypersphere)
- Effect on the two documents d and d′ (d appended to itself) from earlier slide: they have identical vectors after length-normalization.
  - Long and short documents now have comparable weights

cosine(query,document)

Dot product

Unit vectors

$$\cos(\vec{q},\vec{d}) = \frac{\vec{q}\bullet\vec{d}}{|\vec{q}||\vec{d}|} = \frac{\vec{q}}{|\vec{q}|}\bullet\frac{\vec{d}}{|\vec{d}|} = \frac{\sum_{i=1}^{|V|} q_i d_i}{\sqrt{\sum_{i=1}^{|V|} q_i^2}\sqrt{\sum_{i=1}^{|V|} d_i^2}}$$

q_i is the tf-idf weight of term i in the query
d_i is the tf-idf weight of term i in the document

$\cos(\vec{q},\vec{d})$ is the cosine similarity of $\vec{q}$ and $\vec{d}$ … or, equivalently, the cosine of the angle between $\vec{q}$ and $\vec{d}$.

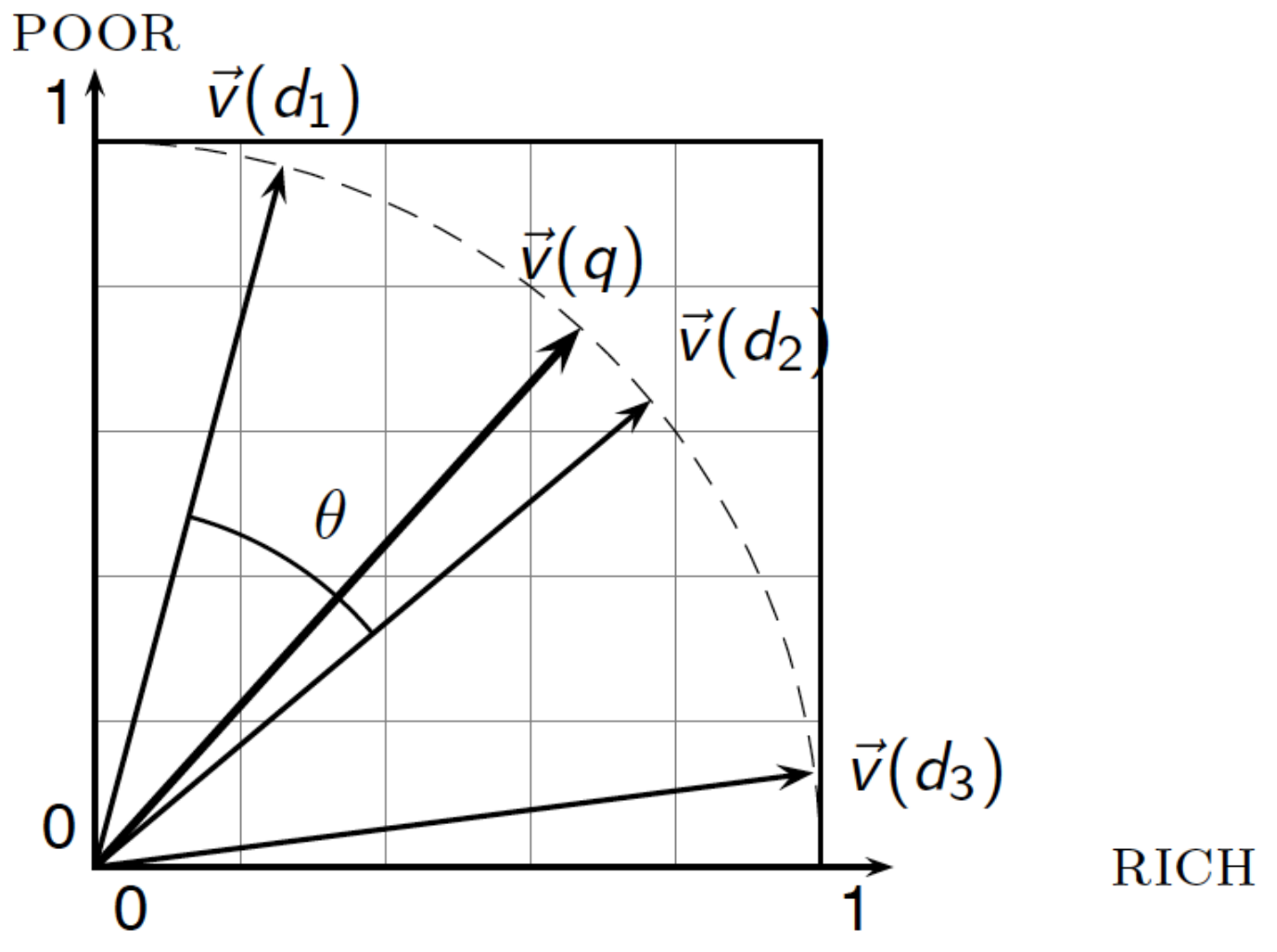
Cosine for length-normalized vectors

- For length-normalized vectors, cosine similarity is simply the dot product (or scalar product):

$$\cos(\vec{q},\vec{d}) = \vec{q} \bullet \vec{d} = \sum_{i=1}^{|V|} q_i d_i$$

for q, d length-normalized.

Cosine similarity illustrated



Cosine similarity amongst 3 documents

How similar are the novels

SaS: *Sense and Sensibility*

PaP: *Pride and Prejudice*, and

WH: *Wuthering Heights*?

term	SaS	PaP	WH
affection	115	58	20
jealous	10	7	11
gossip	2	0	6
wuthering	0	0	38

Term frequencies (counts)

Note: To simplify this example, we don't do idf weighting.

3 documents example contd.

Log frequency weighting

term	SaS	PaP	WH
affection	3.06	2.76	2.30
jealous	2.00	1.85	2.04
gossip	1.30	0	1.78
wuthering	0	0	2.58

After length normalization

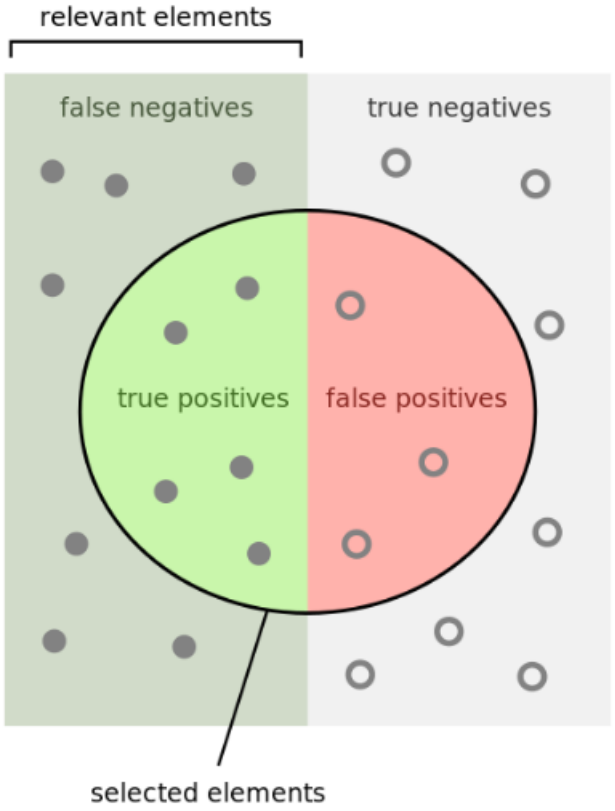
term	SaS	PaP	WH
affection	0.789	0.832	0.524
jealous	0.515	0.555	0.465
gossip	0.335	0	0.405
wuthering	0	0	0.588

cos(SaS,PaP) ≈
0.789 × 0.832 + 0.515 × 0.555 + 0.335 × 0.0 + 0.0 × 0.0
≈ 0.94
cos(SaS,WH) ≈ 0.79
cos(PaP,WH) ≈ 0.69

Computing cosine scores

```
CosineScore(q)
 1  float Scores[N] = 0
 2  float Length[N]
 3  for each query term t
 4  do calculate w_{t,q} and fetch postings list for t
 5      for each pair(d, tf_{t,d}) in postings list
 6      do Scores[d] += w_{t,d} × w_{t,q}
 7  Read the array Length
 8  for each d
 9  do Scores[d] = Scores[d]/Length[d]
10  return Top K components of Scores[]
```

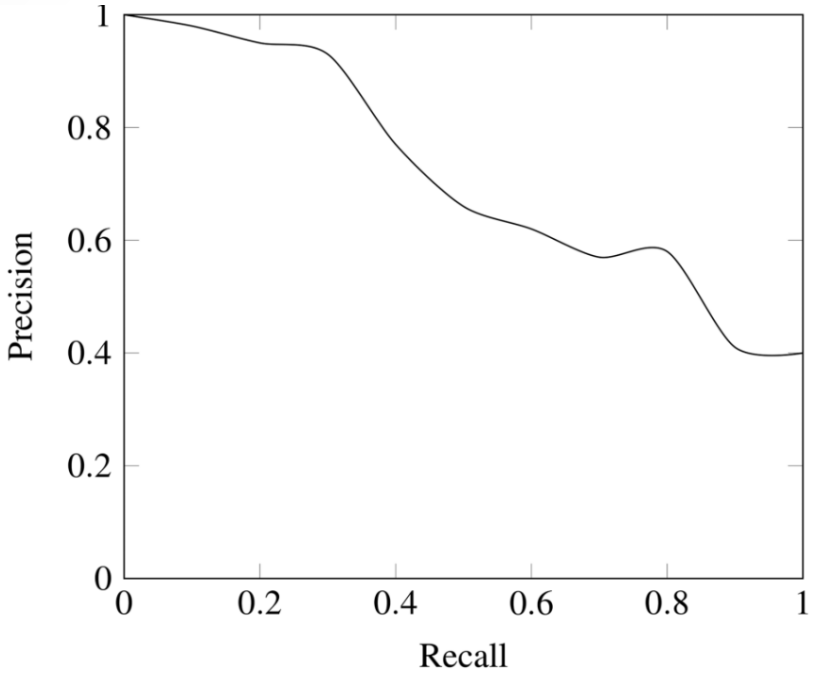
Measuring IR Systems



$$\text{Precision} = \frac{tp}{tp + fp}$$

$$\text{Recall} = \frac{tp}{tp + fn}$$

Measuring IR Systems



$$F_1 = 2 * \frac{precision * recall}{precision + recall}$$

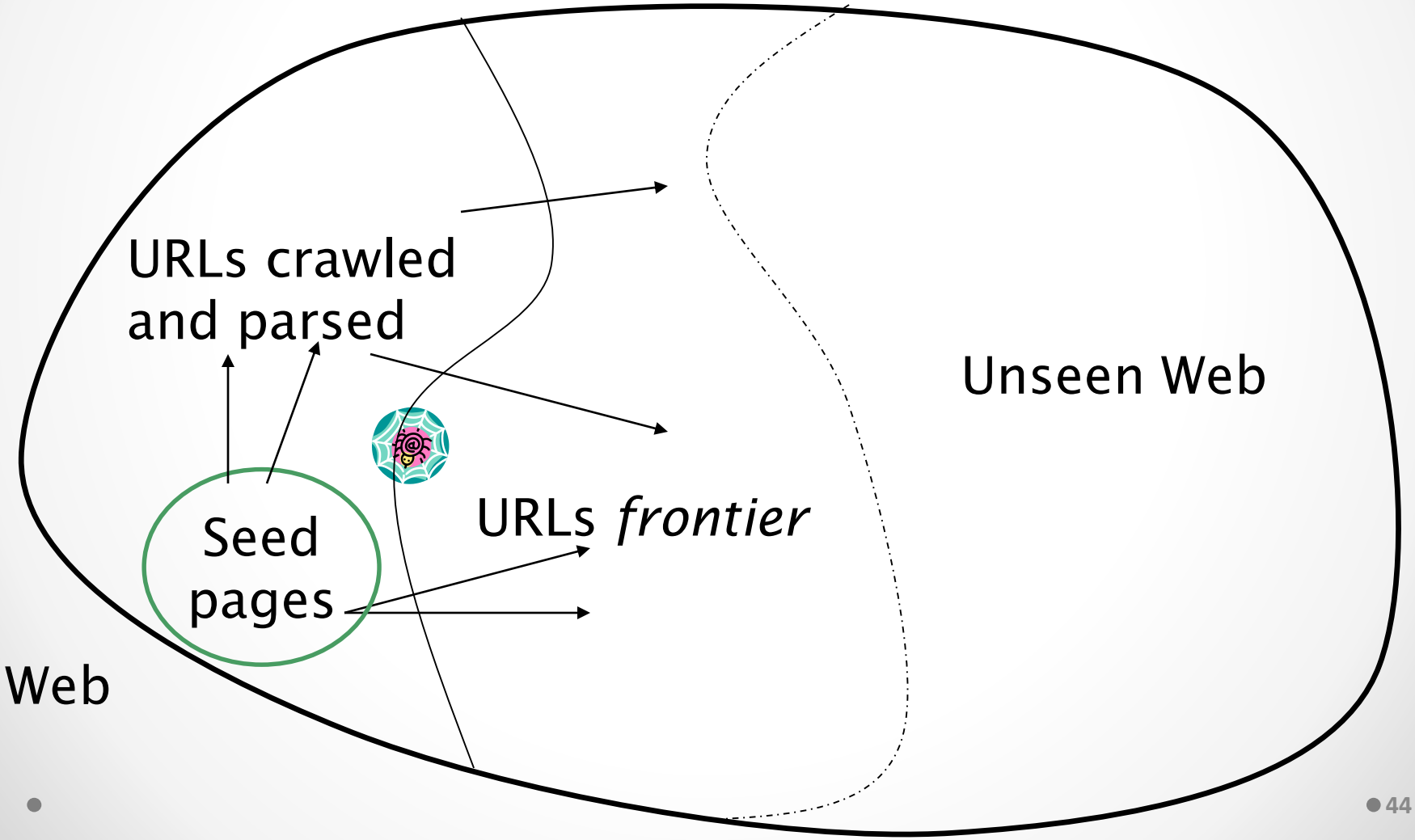
Information Retrieval - Limitations

- Problems with Keyword-based Search
  - Expressing what user wants with just keywords
    - Synonyms (Automobile)
    - Polysemy (Jaguar)
- Problem with Web Search
  - Dynamic nature
    - Can't search for latest information
  - Problem of scarcity to a problem of abundance
    - **Ranking becomes critical**

Basic crawler operation

- Begin with known "seed" URLs
- Fetch and parse them
  - Extract URLs they point to
  - Place the extracted URLs on a queue
- Fetch each URL on the queue and repeat

Crawling picture



Complications

- Web crawling isn't feasible with one machine
  - All of the above steps distributed
- Malicious pages
  - Spam pages
  - Spider traps – incl dynamically generated
  - Recent Trends: Fake news, Hate propagation
- Even non-malicious pages pose challenges
  - Latency/bandwidth to remote servers vary
  - Webmasters' stipulations
    - How "deep" should you crawl a site's URL hierarchy?
  - Site mirrors and duplicate pages
- Politeness – don't hit a server too often

What any crawler *must* do

- Be Polite: Respect implicit and explicit politeness considerations
  - Only crawl allowed pages
  - Respect *robots.txt* (more on this shortly)
- Be Robust: Be immune to spider traps and other malicious behavior from web servers

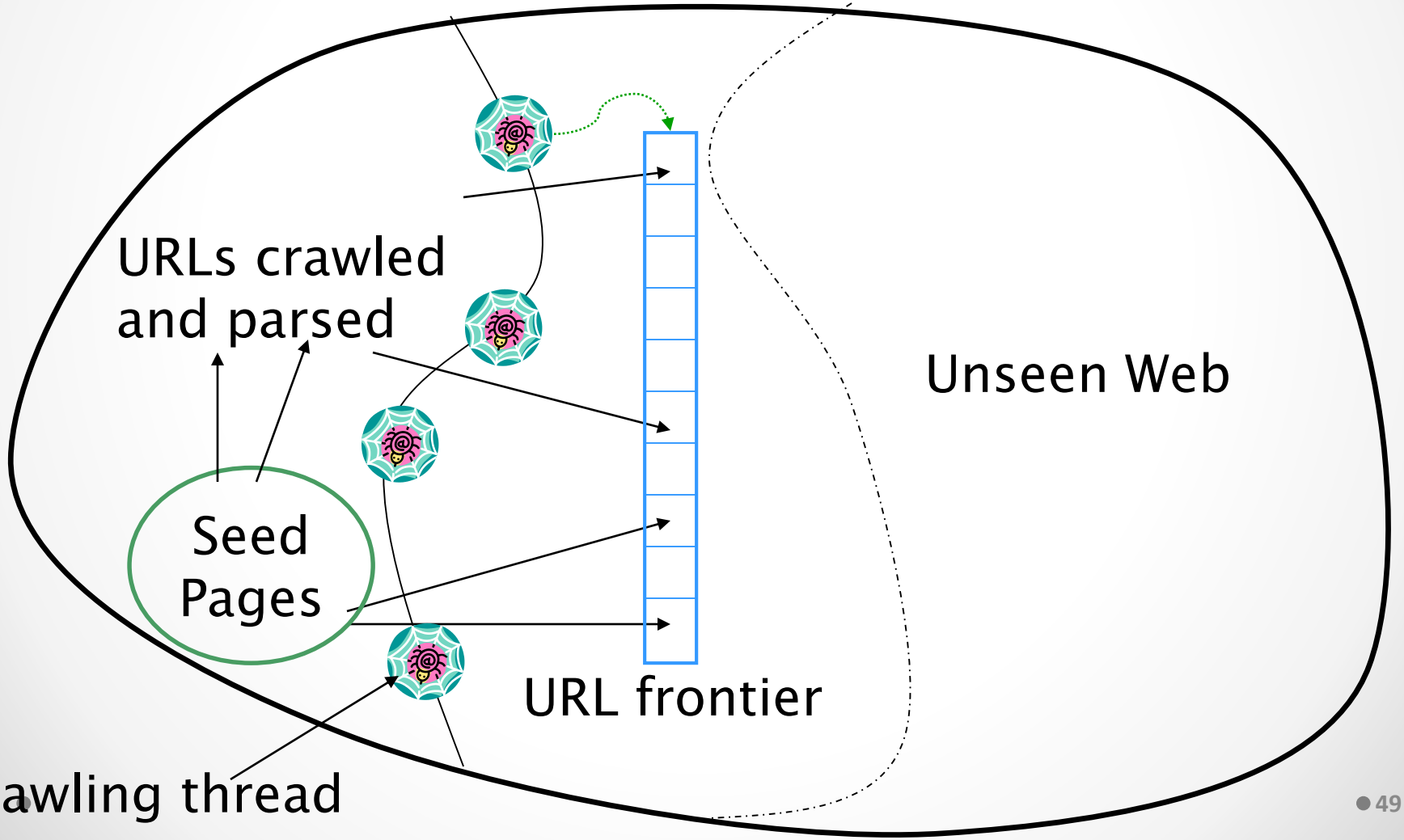
What any crawler *should* do

- Be capable of distributed operation: designed to run on multiple distributed machines
- Be scalable: designed to increase the crawl rate by adding more machines
- Performance/efficiency: permit full use of available processing and network resources

What any crawler *should* do

- Fetch pages of "higher quality" first
- Continuous operation: Continue fetching fresh copies of a previously fetched page
- Extensible: Adapt to new data formats, protocols

Updated crawling picture



URL frontier

- Can include multiple pages from the same host
- Must avoid trying to fetch them all at the same time
- Must try to keep all crawling threads busy

Explicit and implicit politeness

- Explicit politeness: specifications from webmasters on what portions of site can be crawled
  - robots.txt
- Implicit politeness: even with no specification, avoid hitting any site too often

Processing steps in crawling

- Pick a URL from the frontier
- Fetch the document at the URL
- Parse the URL
  - Extract links from it to other docs (URLs)
- Check if URL has content already seen
  - If not, add to indexes
- For each extracted URL
  - Ensure it passes certain URL filter tests
  - Check if it is already in the frontier (duplicate URL elimination)

Breadth-First Traversal

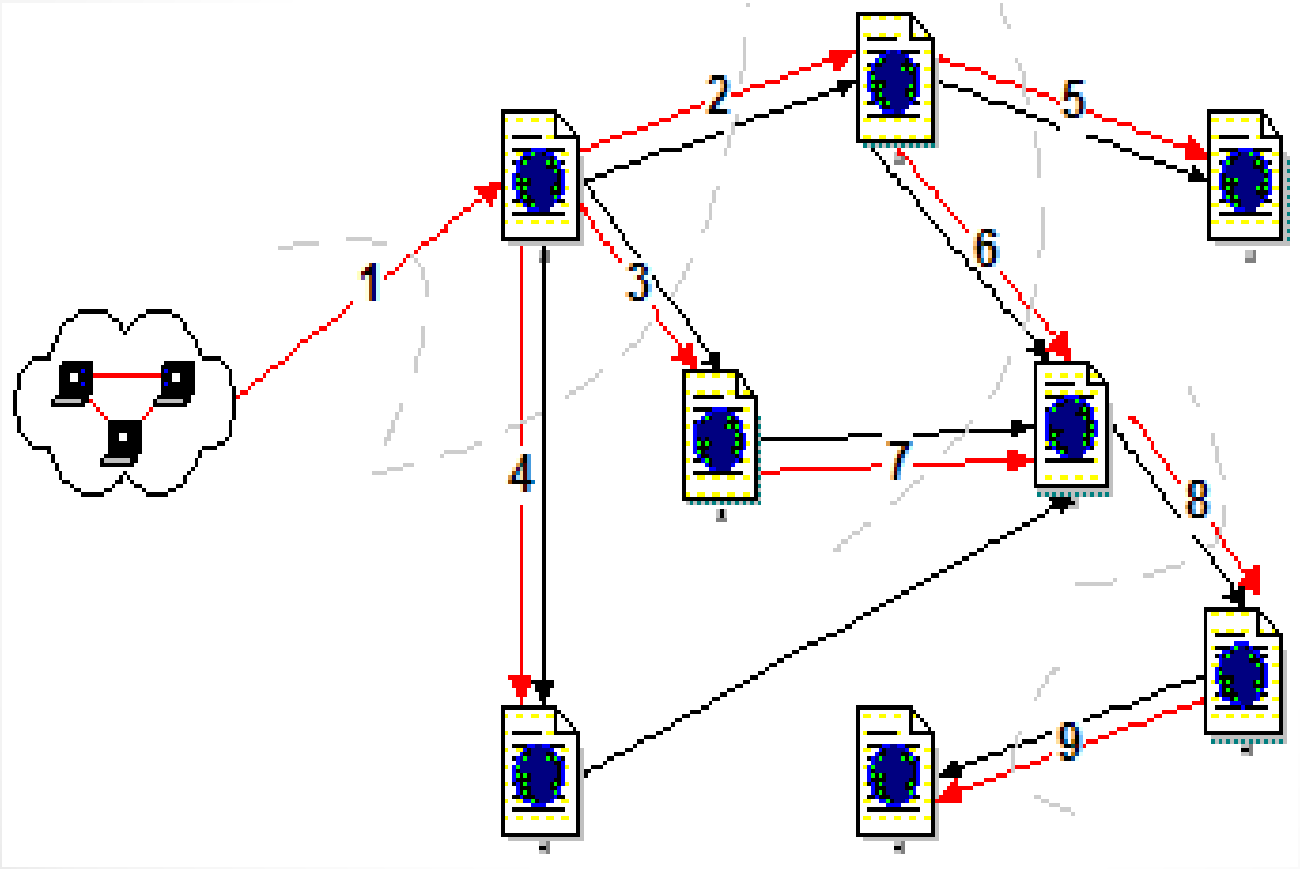
Given any graph and a set of seeds at which to start, the graph can be traversed using the algorithm

1. Put all the given seeds into the queue;
2. Prepare to keep a list of "visited" nodes (initially empty);
3. As long as the queue is not empty:
    a. Remove the first node from the queue;
    b. Append that node to the list of "visited" nodes
    c. For each edge starting at that node:

i. If the node at the end of the edge already appears on the list of "visited" nodes or it is already in the queue, then do nothing more with that edge;

ii. Otherwise, append the node at the end of the edge to the end of the queue.

Breadth First Crawlers

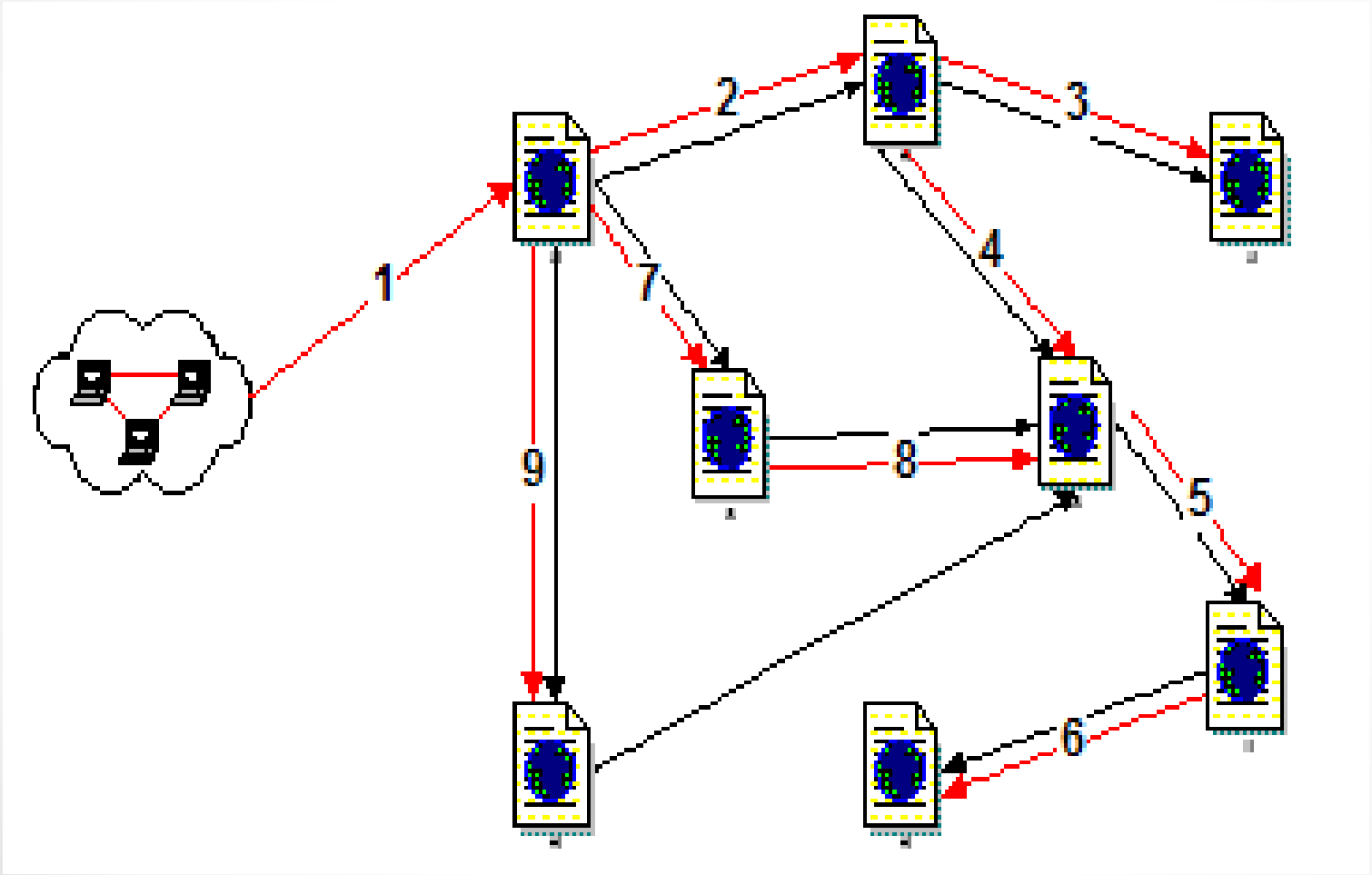


Depth First Crawlers

Use depth first search (DFS) algorithm

- Get the 1st link not visited from the start page
- Visit link and get 1st non-visited link
- Repeat above step till no non-visited links
- Go to next non-visited link in the previous level and repeat 2nd step

Depth first traversal



Depth-First vs. Breadth-First

- depth-first goes off into one branch until it reaches a leaf node
  - not good if the goal node is on another branch
  - neither complete nor optimal
  - uses much less space than breadth-first
    - much fewer visited nodes to keep track of
    - smaller fringe
- breadth-first is more careful by checking all alternatives
  - complete and optimal
  - very memory-intensive

Hubs & Authorities

- Hubs: Large directories that were not actually authoritative in the information that they held, but were used as compilations of a broad catalog of information that led users direct to other authoritative pages. In other words, a good hub represented a page that pointed to many other pages
- Authorities: Good source of information on a topic. Good authority represented a page that was linked by many different hubs
- HITS Algorithm: To find good hubs & authorities (Kleinberg)
  - Jon Kleinberg. Authoritative sources in a hyperlinked environment. Journal of the ACM, 46(5):604–632, 1999

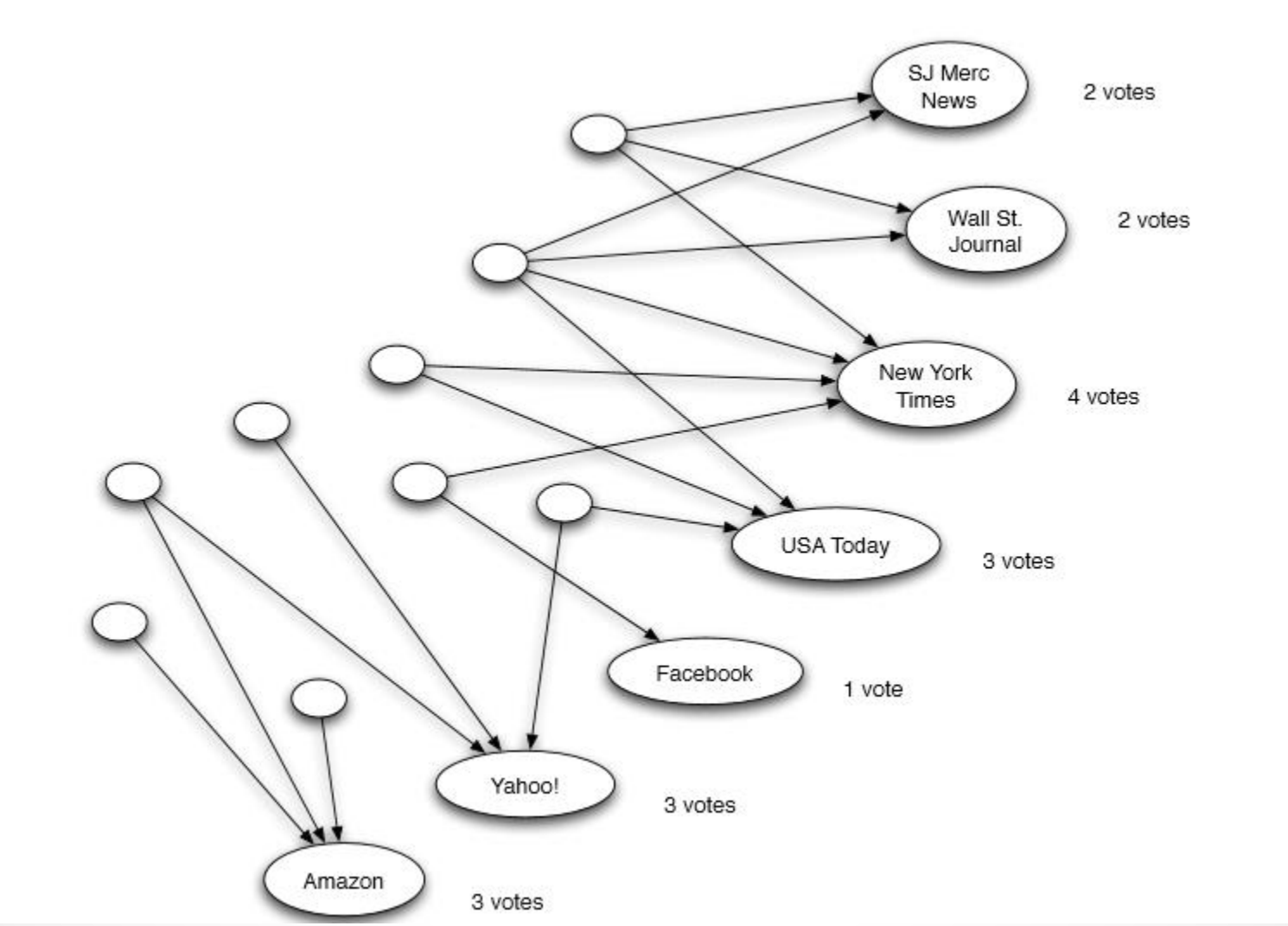
HITS Algorithm

- Retrieve the most relevant pages to the search query(*Root set*)
- Obtain a *base set* is generated by augmenting the root set with all the web pages that are linked from it and some of the pages that link to it. The web pages in the base set and all hyperlinks among those pages form a focused subgraph.
- Authority and hub values are defined in terms of one another in a mutual recursion.
  - An authority value is computed as the sum of the scaled hub values that point to that page.
  - A hub value is the sum of the scaled authority values of the pages it points to. Some implementations also consider the relevance of the linked pages.
- The algorithm performs a series of iterations, each consisting of two basic steps:
  - **Authority Update**: Update each node's *Authority score* to be equal to the sum of the *Hub Scores* of each node that points to it. That is, a node is given a high authority score by being linked from pages that are recognized as Hubs for information.
  - **Hub Update**: Update each node's *Hub Score* to be equal to the sum of the *Authority Scores* of each node that it points to. That is, a node is given a high hub score by linking to nodes that are considered to be authorities on the subject.

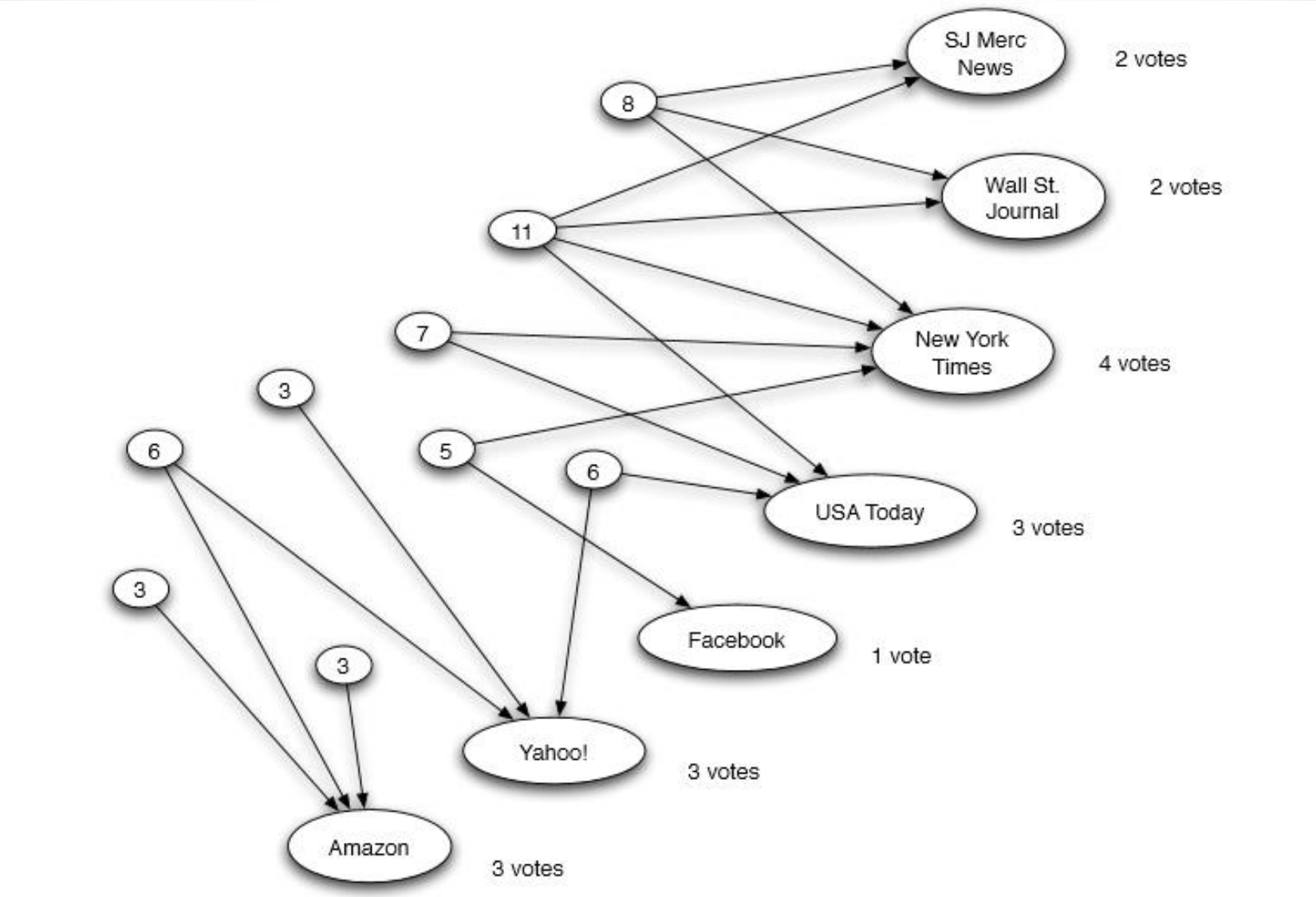
HITS Algorithm - Steps

- The Hub score and Authority score for a node is calculated with the following algorithm:
  I. Start with each node having a hub score and authority score of 1.
  II. Run the Authority Update Rule
  III. Run the Hub Update Rule
  IV. Normalize the values by dividing each Hub score by square root of the sum of the squares of all Hub scores, and dividing each Authority score by square root of the sum of the squares of all Authority scores.
  V. Repeat from the second step as necessary *k* times or until the value changes are very small

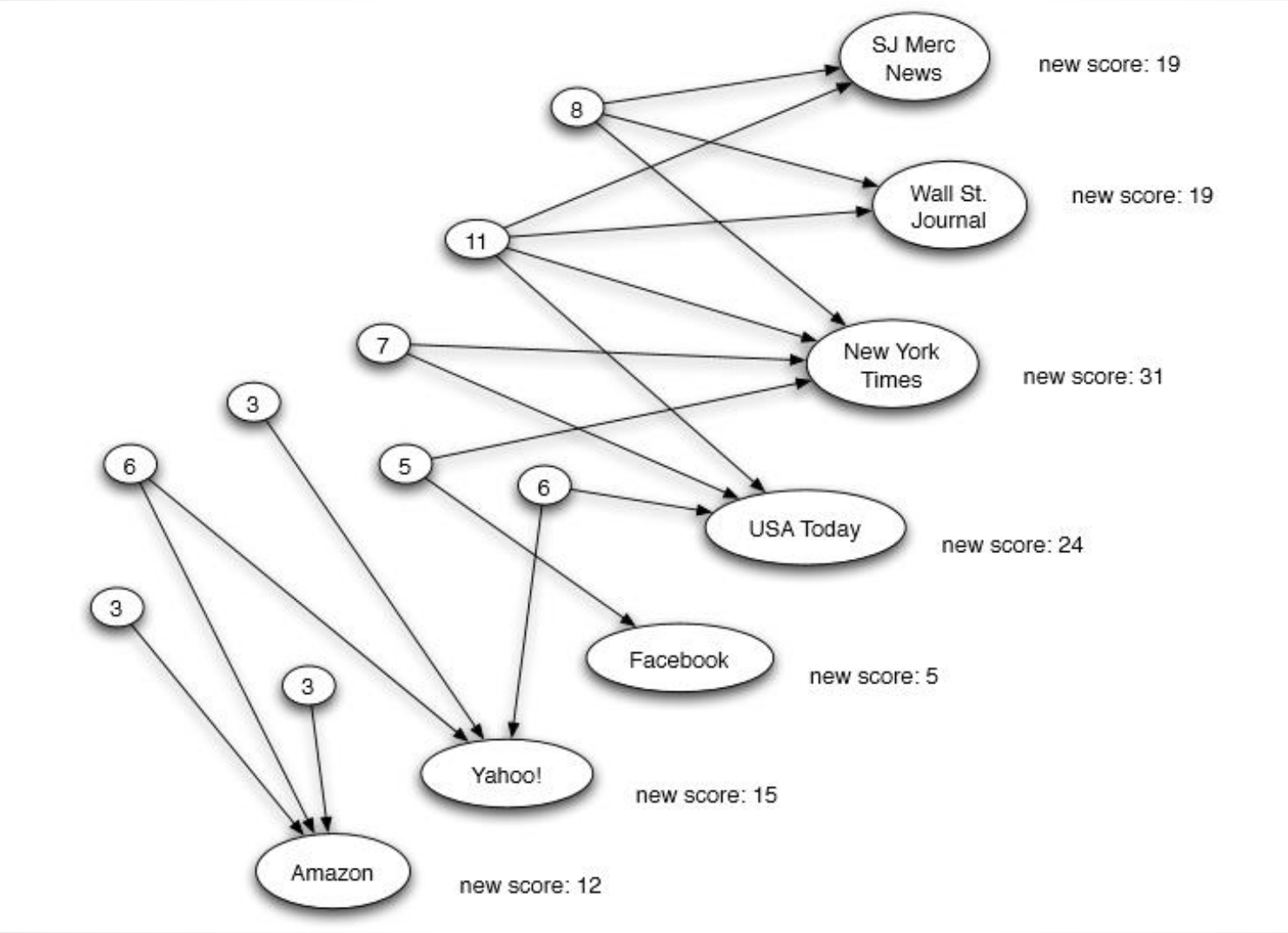
HITS Algorithm – 1st Authority Update



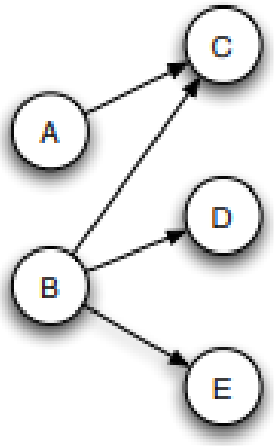
HITS Algorithm – 1st Hub Update



HITS Algorithm – 2nd Authority Update



HITS Algorithm – Exercise



Hub & Authority Scores after 2 iterations of HITS Algorithm

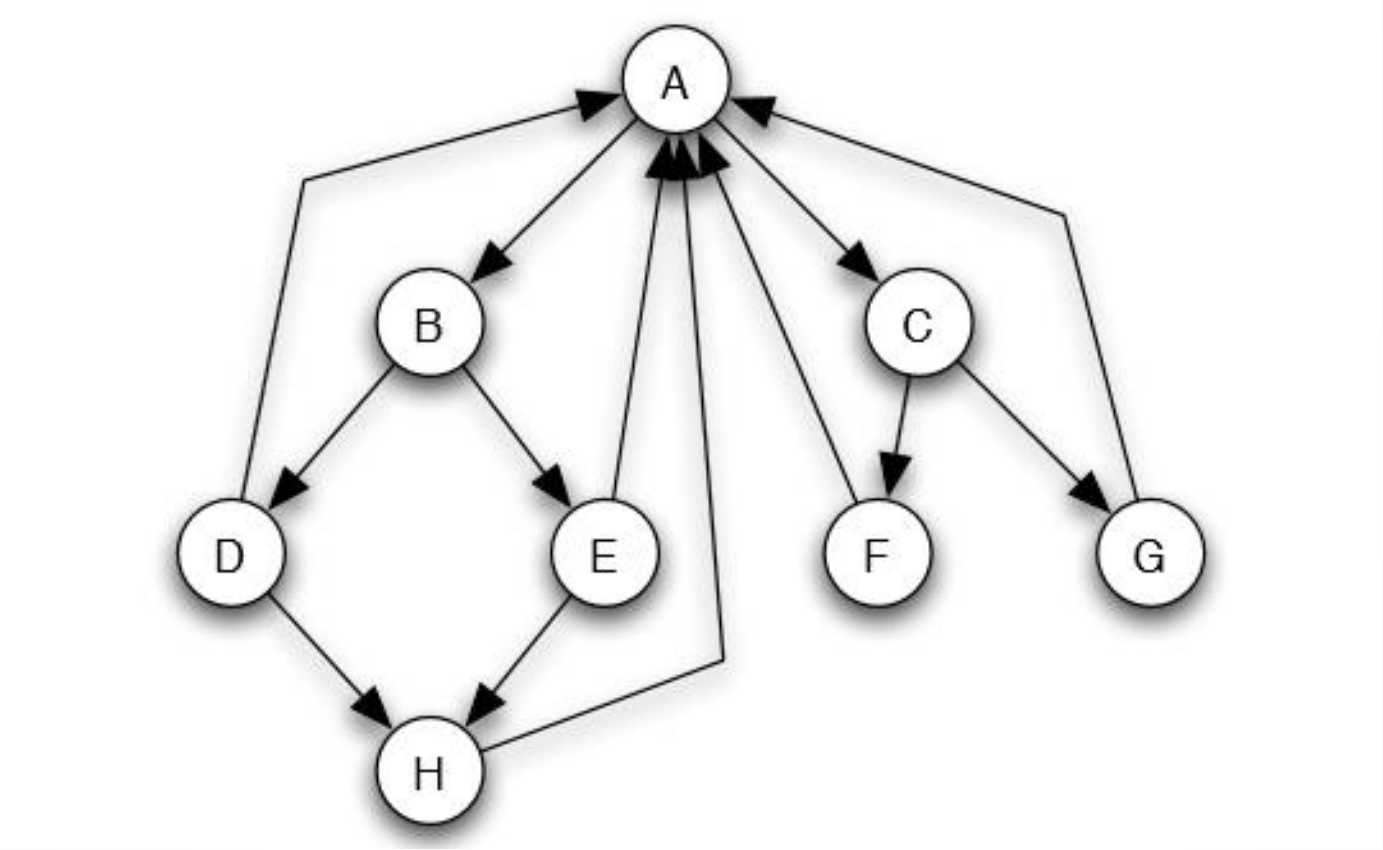
HITS Algorithm - Features

- Works on *Principle of Repeated Improvement*
- The normalized values actually converge to limits as *k* goes to infinity
- Except in a few rare cases (characterized by a certain kind of degenerate property of the link structure), we reach the same limiting values no matter what we choose as the initial hub and authority values, provided only that all of them are positive
  - The limiting hub and authority values are a property purely of the link structure, not of the initial estimates

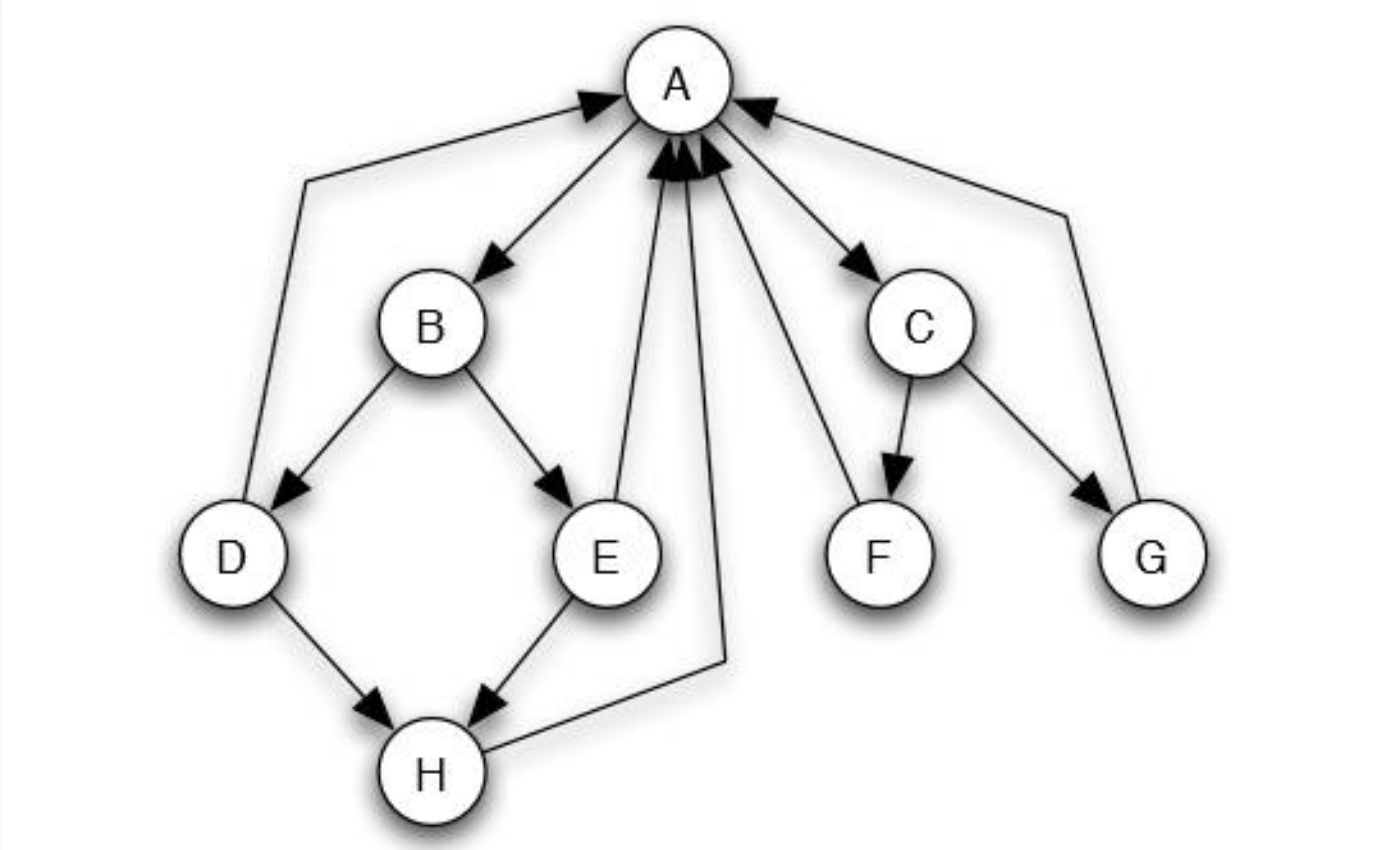
Page Rank

- In a network with *n* nodes, we assign all nodes the same initial PageRank, set to be *1/n*
- We then perform a sequence of k updates to the PageRank values, using the following rule for each update:
  - Each page divides its current PageRank equally across its out-going links and passes these equal shares to the pages it points to. (If a page has no out-going links, it passes all its current PageRank to itself.)
  - Each page updates its new PageRank to be the sum of the shares it receives
- Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual Web search engine. In Proc. 7th International World Wide Web Conference, pages 107–117, 1998.

Exercise – PageRank after 1st update

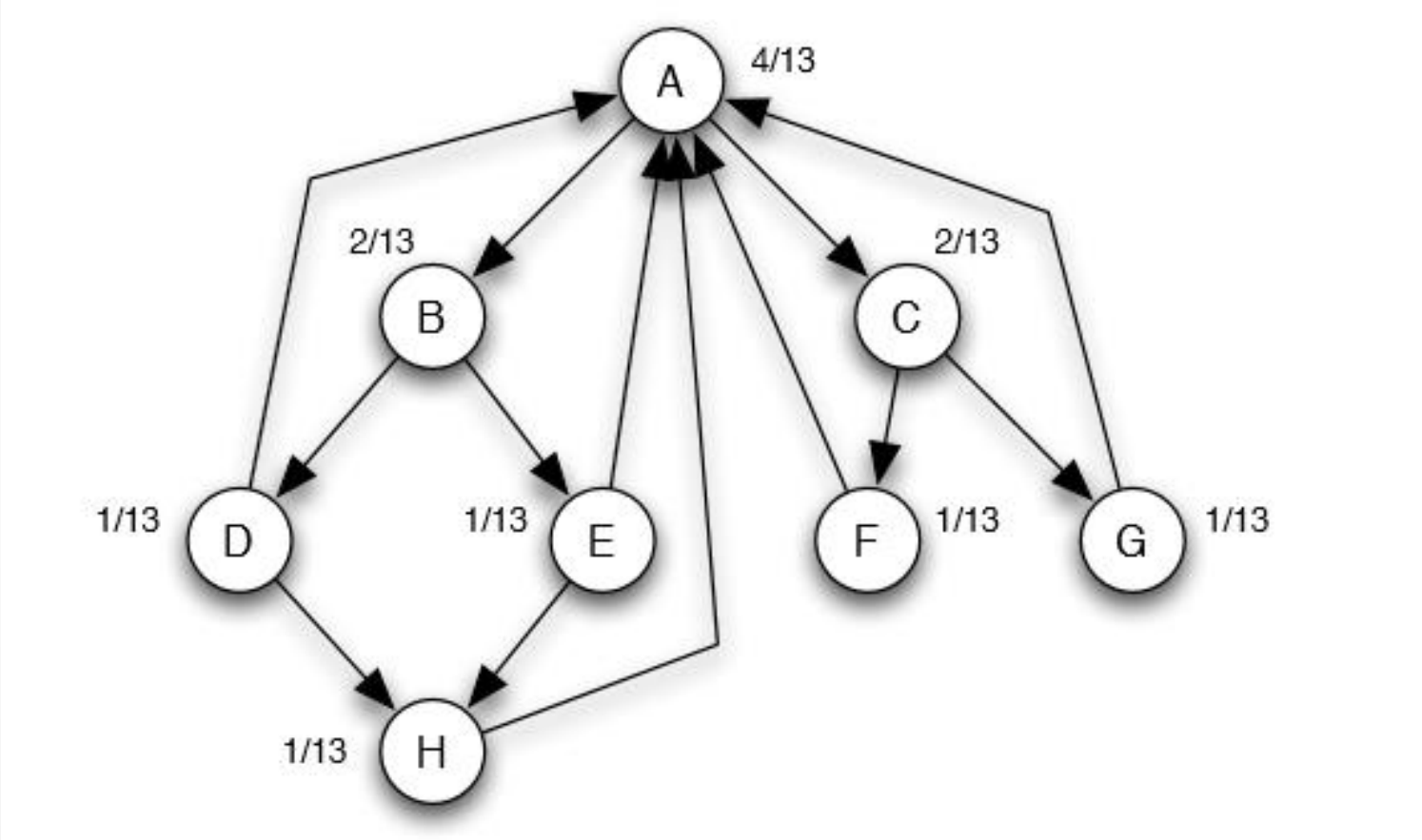


Exercise – PageRank after 2 updates



Step	A	B	C	D	E	F	G	H
1	1/2	1/16	1/16	1/16	1/16	1/16	1/16	1/8
2	5/16	1/4	1/4	1/32	1/32	1/32	1/32	1/16

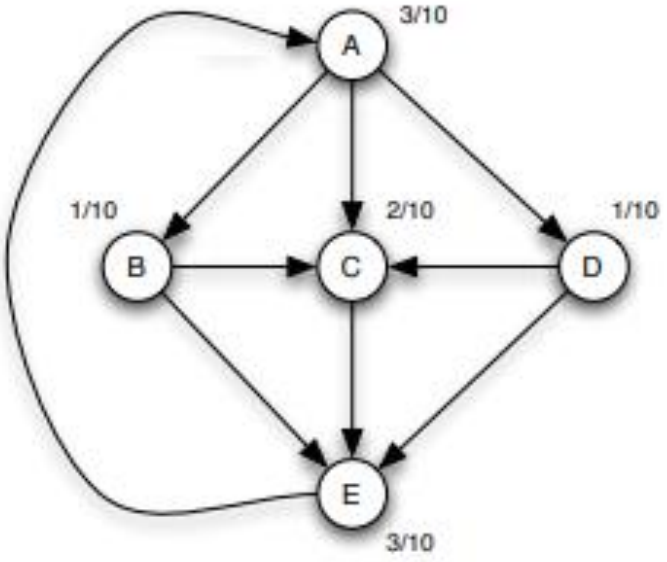
Final PageRank



Page Rank - Features

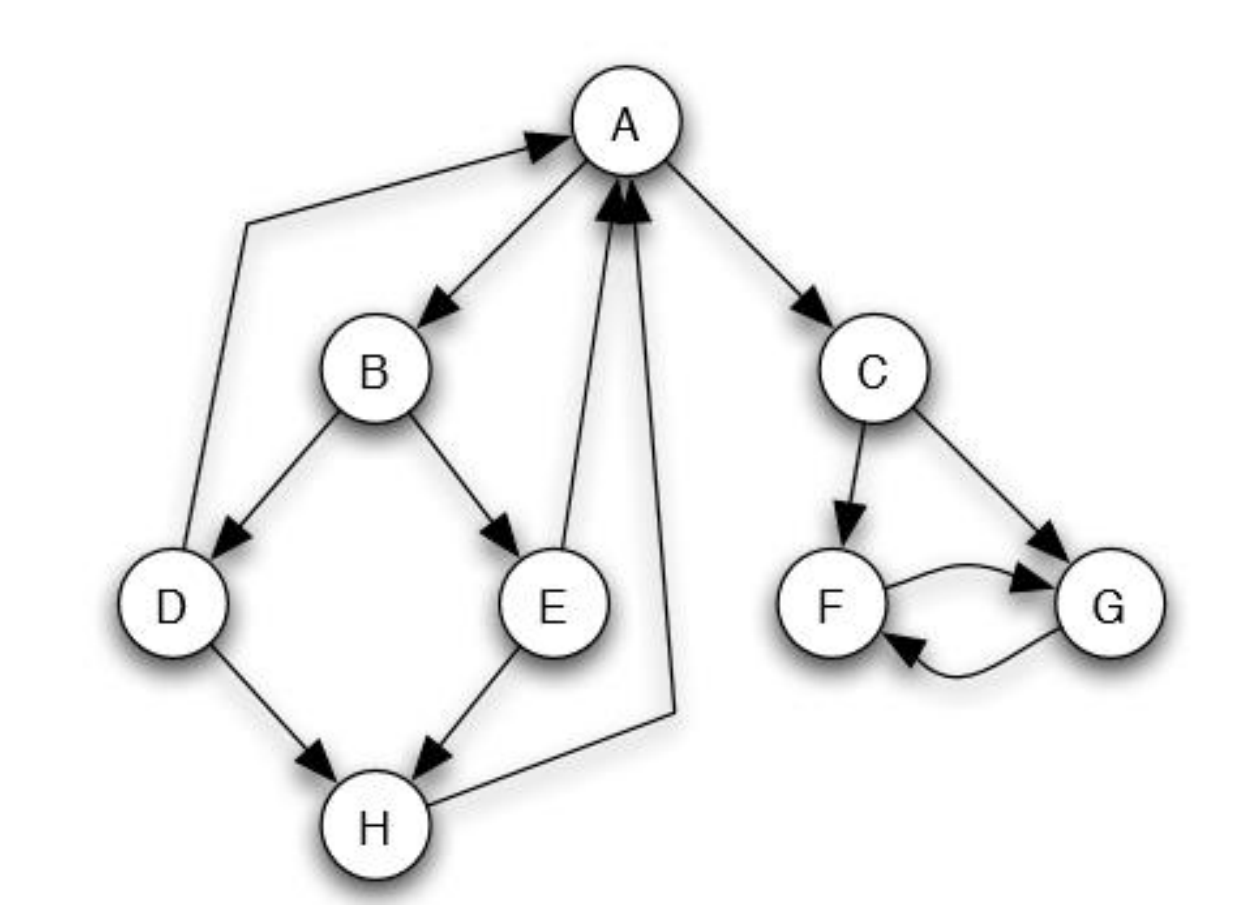
- The values actually converge to limits as *k* goes to infinity
- PageRank is conserved throughout the computation — with the total PageRank in the network equal to one
- Applying one step of the Basic PageRank Update Rule the limiting PageRank values and, then the values at every node remain the same
- If the network is strongly connected then there is a unique set of equilibrium value

Exercise



Are these the correct final PageRank values for this network?

Problem with basic Page Rank



Scaled Page Rank

- Problem in almost any real network: as long as there are small sets of nodes that can be reached from the rest of the graph, but have no paths back, then PageRank will build up there
- Scaled PageRank Update Rule:
  - First apply the Basic PageRank Update Rule.
  - Then scale down all PageRank values by a factor of *s*. This means that the total PageRank in the network has shrunk from *1* to *s*
  - We divide the residual *1−s* units of PageRank equally over all nodes, giving *(1−s)/n* to each
- In practice *s=0.8-0.9*

Page Rank & Random Walk

- Random Walk: Start at a page and randomly move to any other page
- The probability of being at a page X after k steps of this random walk is precisely the PageRank of X after k applications of the Basic PageRank Update Rule
- *Scaled Page Rank:* With probability *s* the walker follows a random edge as before; and with probability *1−s* the walker jumps to a random node anywhere in the network, choosing each node with equal probability

Page Rank in Web Search

- Page Rank started by Google and used in many other Search engines
- Currently Google may be using a different technique – most probably combination of various algorithms
- An effective algorithm needs to combine multiple methods
  - Anchor Text (We can give more weights to those links)
  - User feedback
- Must guard against Web sites that may try to get on top of Search results
  - Pages created to fool search engines – SPAMDEXING.
  - Search engine business depends on successful filtering of SPAM pages

Some Spamming Techniques

- Changing a color scheme for keyword stuffing: e.g. white text on white background
- Link farms - Creating a number of bogus web pages that link to one page in order to give it a better rank
- Honey Pot - Provide some valuable content but contain links to SPAM pages
- Registering many keyword-rich domains
- Scraper sites – Scrape content from search engines and other websites
- Article spinning – Rewriting existing articles to escape duplicate content penalty
- Cloaking – Serving the page differently to the crawler than to the humans

How do we filter out SPAM?

- How do we separate the "good" ones from the "bad" ones?
- It turns out its hard to do it automatically
- The most reliable way is to use human experts but how do we do that for billions of pages?
- Is there a way to conclude something based on a small set of pages reviewed by experts?

Problem Definition

- How can we semi-automatically estimate which pages are good and which ones are bad (SPAM) provided that we have a limited number of experts?
- Can we reliably say which ones are probably good and/or probably bad based on a small "seed" of pages reviewed by experts?
- How can we do it effectively and efficiently?

Oracle and Trust Function

- The notion of human checking of a web page is represented by Oracle function:

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

- Oracle invocations are expensive, one should strive to minimize them
- Important empirical observation for trust: *approximate isolation* of the good set
- Good pages rarely link to bad ones
- The converse does not hold

- To evaluate the pages without calling O, it is necessary to estimate the probability that p is good
- The Trust function yields a range of values between 0 (bad) and 1 (good)
- Ideally, $T(p) = \Pr[O(p) = 1].$
- This is hardly ever true in practice
- A relaxed constraint is orderedness by pair, so that we can display search results based on that order

$$T(p) < T(q) \Leftrightarrow \Pr[O(p) = 1] < \Pr[O(q) = 1].$$
$$T(p) = T(q) \Leftrightarrow \Pr[O(p) = 1] = \Pr[O(q) = 1].$$

- Another method of relaxing the requirements of T is introducing a **threshold** value

$$T(p) > \delta \Leftrightarrow O(p) = 1$$

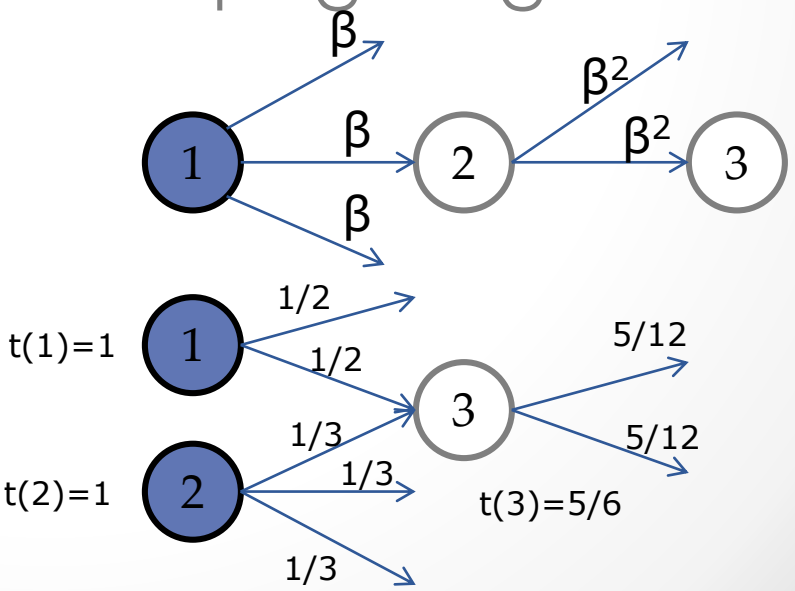
- If a page receives a score above δ we know it is good.
- Otherwise, we cannot say anything
- This does not necessarily provide ordering based on the likelihood of being good

Seed Selection

- Random selection is simplest but it may hinder TrustRank effectiveness
- Oracle invocations are expensive, because they require human effort
- Chosen pages should be useful in identifying additional good pages
- Seed set should be reasonably small to limit oracle invocations
- Trust flows out of the good seed pages, so give preference to pages which reach many other pages
- Select pages based on the number of outlinks
- Scheme closely related to PageRank
- Difference is that importance depends on outlinks not inlinks

Trust Attenuation

- We cannot be absolutely sure that pages reachable from good seeds are indeed good
- Further away we are from good seed, less certain we are that a page is good
- Trust dampening
  β – dampening factor
- Trust splitting
- Can be combined



Further Improvements

- During the seed selection it is not necessary to order all pages using inversed PageRank
- If we already have PageRank score, we can choose subset of highly ranked pages
- Users will be more interested in those pages anyway
- Anti-Trust Rank: The closer a site is to spam resources, the more likely it is to be spam as well.

Hilltop Algorithm

- Assumption: The number and quality of the sources referring to a page are a good measure of the page's quality.
- Novelty: Only considering "expert" sources - pages that have been created with the specific purpose of directing people towards resources.
- In response to a query, compute a list of the most relevant experts on the query topic.
- Identify relevant links within the selected set of experts, and follow them to identify target web pages.
- The targets are then ranked according to the number and relevance of non-affiliated experts that point to them.
- Thus, the score of a target page reflects the collective opinion of the best independent experts on the query topic.
- When such a pool of experts is not available, Hilltop provides no results.
- Thus, Hilltop is tuned for result accuracy and not query coverage.
- Algorithm consists of two broad phases:

i. Expert Lookup
ii. Target Ranking

Expert Lookup

- **A**n expert page is a page that is about a certain topic and has links to many non-affiliated pages on that topic.
  - Two pages are non-affiliated conceptually if they are authored by authors from non-affiliated organizations.
- In a pre-processing step, a subset of the pages crawled by a search engine are identified as experts
- Given an input query, a lookup is done on the expert-index to find and rank matching expert pages.
- This phase computes the best expert pages on the query topic as well as associated match information.

Target Ranking

- A page is an authority on the query topic if and only if some of the best experts on the query topic point to it.
- Of course in practice some expert pages may be experts on a broader or related topic.
  - If so, only a subset of the hyperlinks on the expert page may be relevant.
- By combining relevant out-links from many experts on the query topic we can find the pages that are most highly regarded by the community of pages related to the query topic
- Given the top ranked matching expert-pages and associated match information, we select a subset of the hyperlinks within the expert-pages.
- Specifically, we select links that we know to have all the query terms associated with them. This implies that the link matches the query.
- With further connectivity analysis on the selected links we identify a subset of their targets as the top-ranked pages on the query topic.
- The targets identified are those that are linked to by *at least two* non-affiliated expert pages on the topic.
- The targets are ranked by a ranking score which is computed by combining the scores of the experts pointing to the target.

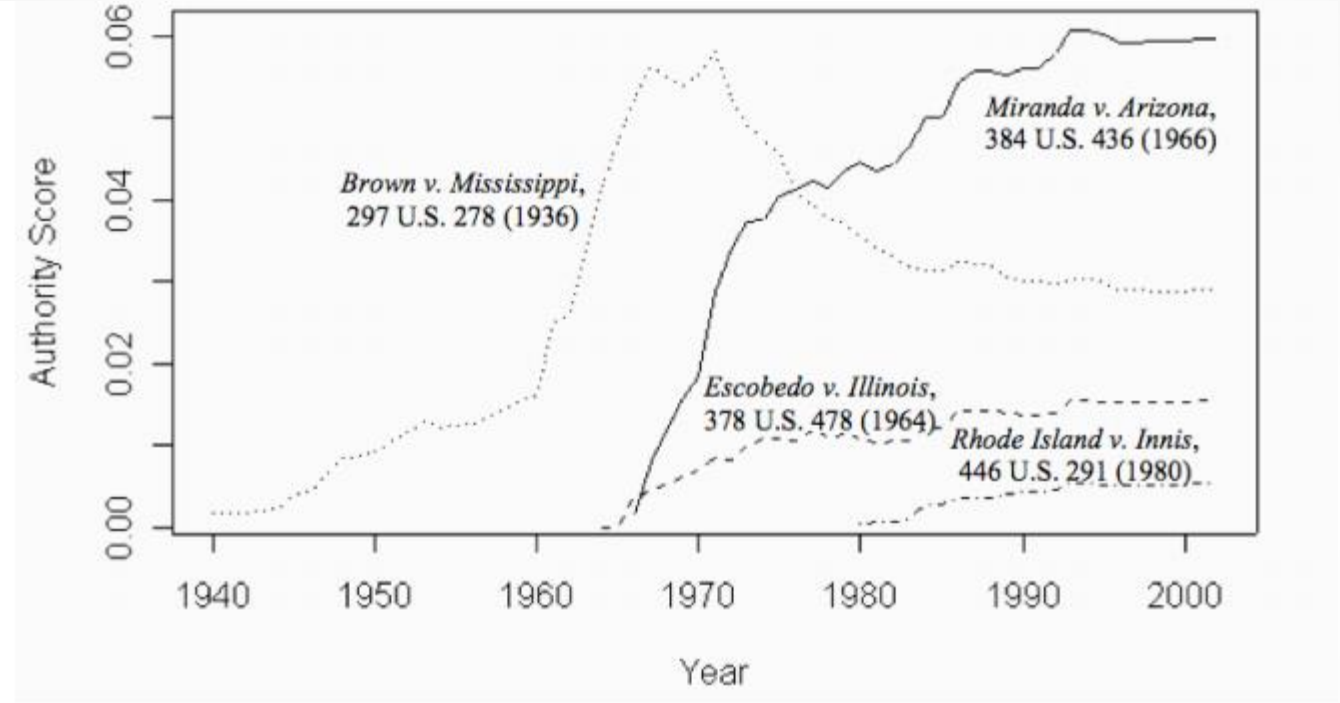
Citation Analysis

- Impact factor for a scientific journal is defined to be the average number of citations received by a paper in the given journal over the past two years.
- This type of voting by in-links can thus serve as a proxy for the collective attention that the scientific community pays to papers published in the journal.
- In the 1970s, Pinski and Narin extended the impact factor by taking into account the idea that not all citations should be counted equally — rather, citations from journals that are themselves high-impact should be viewed as more important

Citation Analysis in US Supreme Court

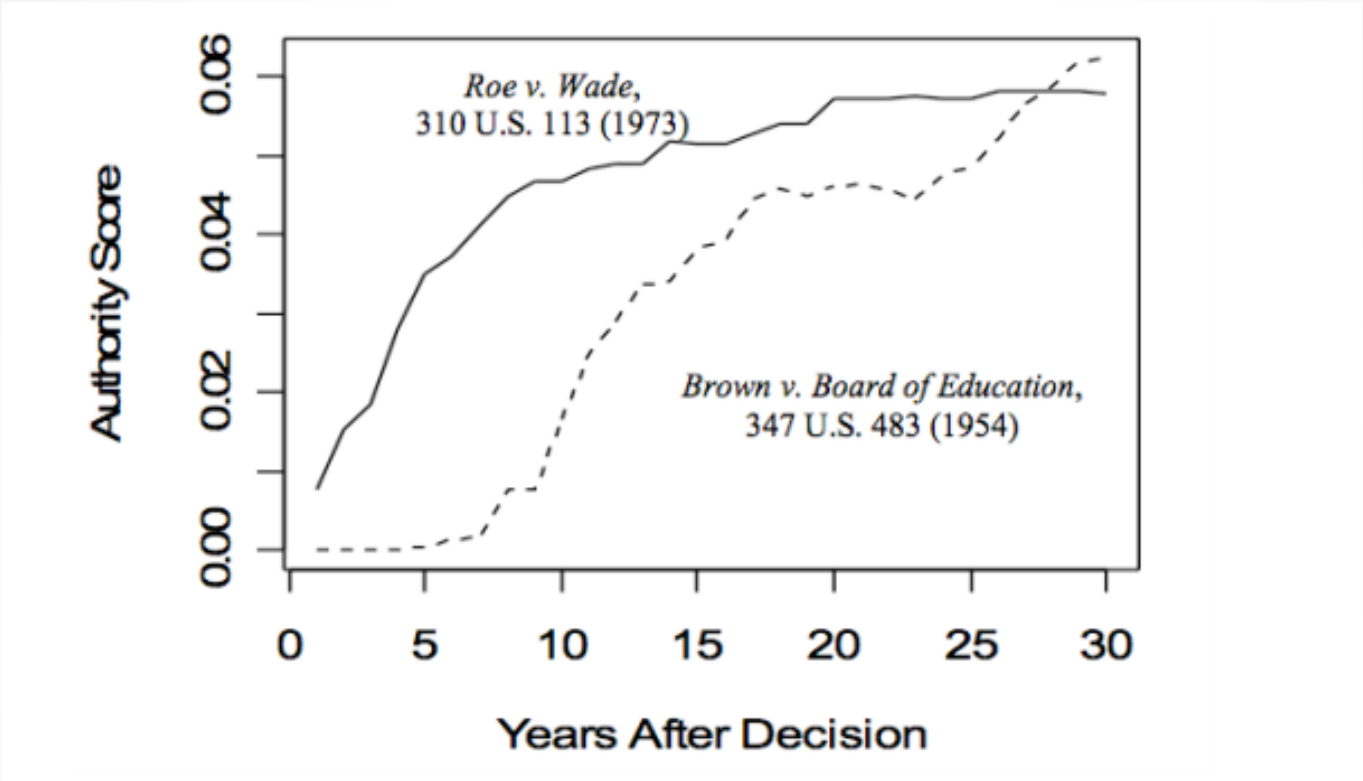
- Citations are crucial in legal writing, to ground a decision in precedent and to explain the relation of a new decision to what has come before.
- link analysis in this context can help in identifying cases that play especially important roles in the overall citation structure.
- In one example of this style of research, Fowler and Jeon applied hub and authority measures to the set of all U.S. Supreme Court decisions, a collection of documents that spans more than two centuries.
- They found that the set of Supreme Court decisions with high authority scores in the citation network align well with the more qualitative judgments of legal experts about the Court's most important decisions.

Citation Analysis in US Supreme Court



Authority for a particular topic can change over long time periods.

Citation Analysis in US Supreme Court



Significant decisions can vary widely in the rate at which they acquire authority.

Reading

1. https://www.cs.cornell.edu/home/kleinber/networks-book/
   - **Chapter 13 & 14.1-14.5**
2. Gyongyi, H. Garcia-Molina and J.Pedersen. Combating Web Spam with TrustRank. Tech. rep., Stanford University, 2004.
3. Krishna Bharat and George Mihaila Hilltop: A Search Engine based on Expert Documents (Poster). In *9th International WWW Conference (WWW9)*, May 2000, Amsterdam, Netherlands.