CS3245

Information Retrieval

11

Lecture 11: Web Search

Last Time

Chapter 11

1. Probabilistic Approach to Retrieval / Basic Probability Theory
2. Probability Ranking Principle
3. OKAPI BM25

Chapter 12

1. Language Models for IR

Today

Chapter 19

- Web search big picture
- Search Advertising
- Duplicate Detection

Chapter 20

- Crawling

Chapter 21

- Anchor Text
- PageRank

IR on the web vs. IR in general

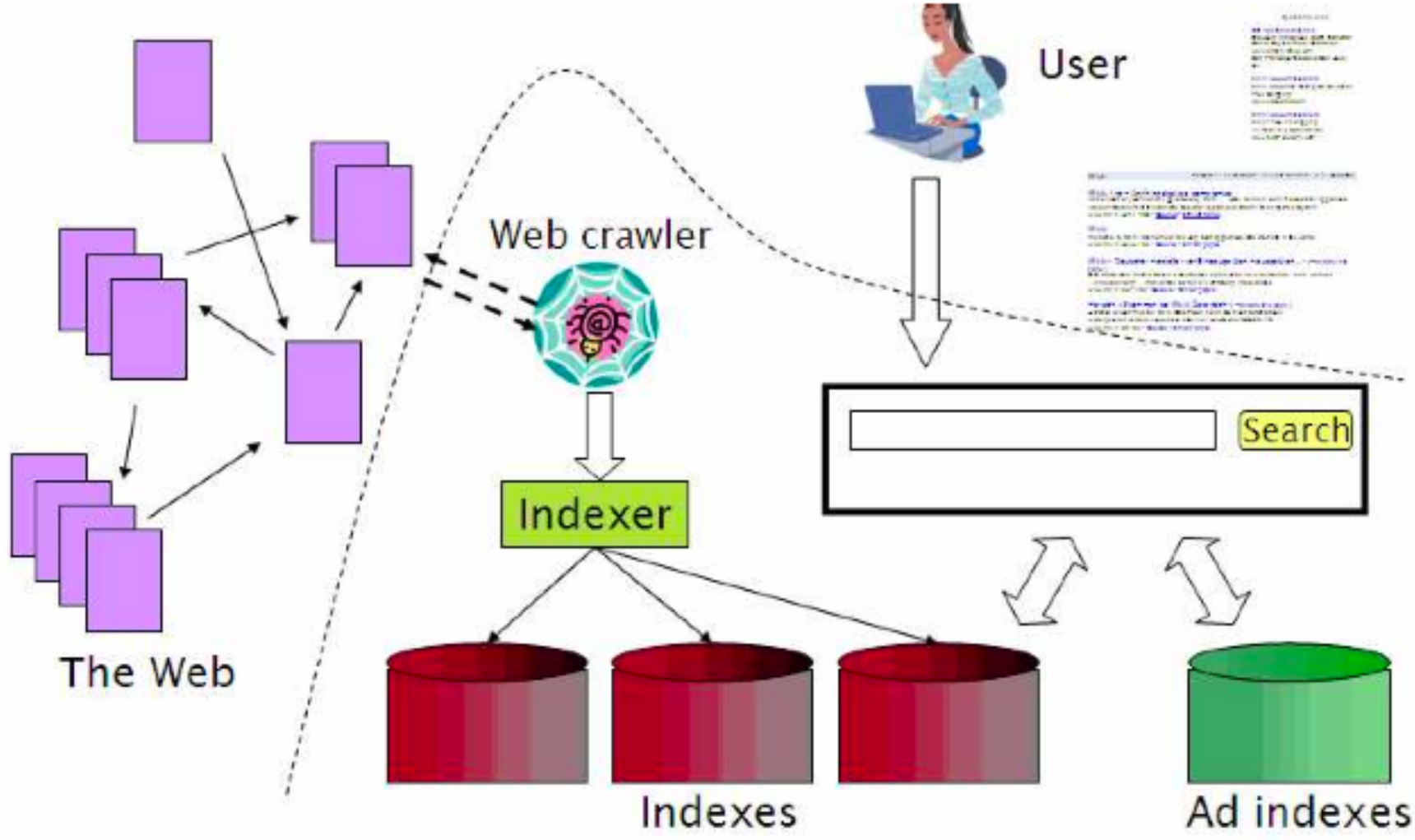
- On the web, search is not just a nice feature.
  - Search is a key enabler of the web: financing, content creation, interest aggregation, etc.

  → look at search ads
- The web is a chaotic und uncoordinated collection.

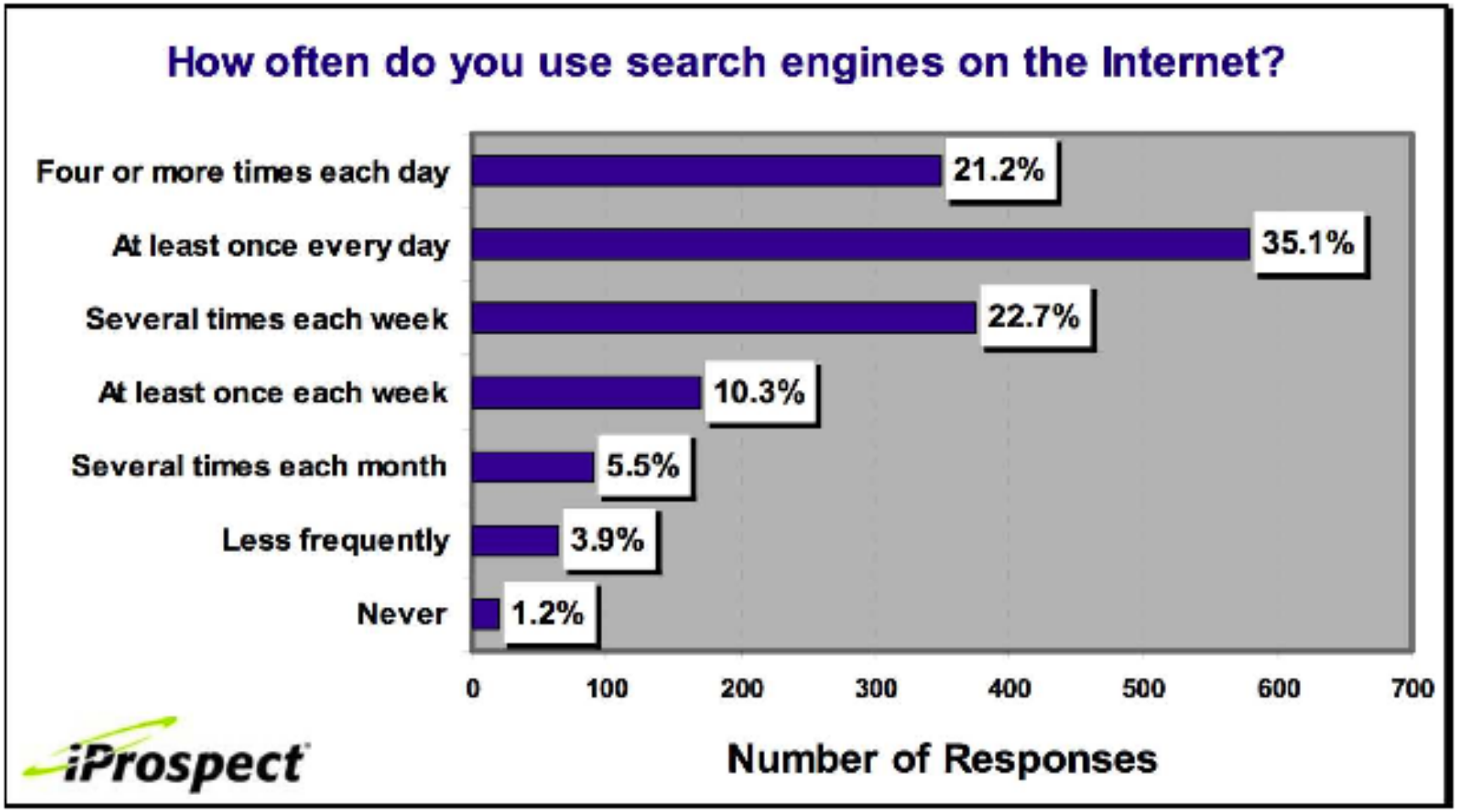
  → lots of duplicates – need to detect duplicates
- No control / restrictions on who can author content.

  → lots of spam – need to detect spam
- The web is very large. → need to know how big it is.

Web search overview



Search is the top activity on the web



Without search engines, the web wouldn't work

- Without search, content is hard to find.
- → Without search, there is no incentive to create content.
  - Why publish something if nobody will read it?
  - Why publish something if I don't get ad revenue from it?
- Somebody needs to pay for the web.
  - Servers, web infrastructure, content creation
  - A large part today is paid by search ads: Search pays for the web.

Interest aggregation

- Unique feature of the web: A small number of geographically dispersed people with similar interests can find each other.
  - Elementary school kids with hemophilia
  - People interested in translating R5R5 Scheme into relatively portable C (open source project)
  - Search engines are a key enabler for interest aggregation.
- The Long Tail

SEARCH ADVERTISING

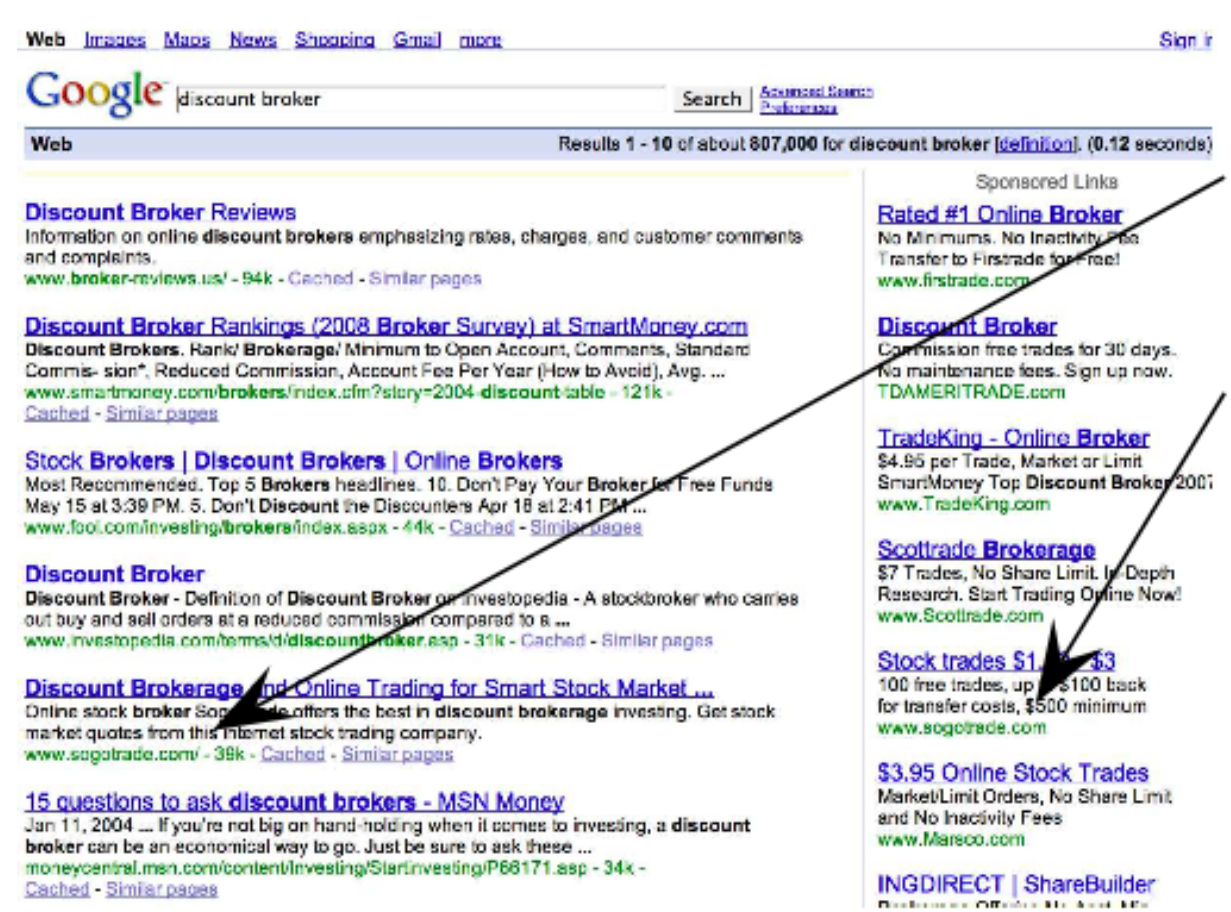
1st Generation of Search Ads: Goto (1996)

1st Generation of Search Ads: Goto (1996)

- Buddy Blake bid the maximum ($0.38) for this search.
- He paid $0.38 to Goto every time somebody clicked on the link.
- Pages were simply ranked according to bid – revenue maximization for Goto.
- No separation of ads/docs. Only one result list!
- Upfront and honest. No relevance ranking, . . .

  . . . but Goto did not pretend there was any.

2nd generation of search ads: Google (2000)



SogoTrade appears in search results.

SogoTrade appears in ads.

Do search engines rank advertisers higher than non-advertisers?

All major search engines claim "no".

Do ads influence editorial content?

- Similar problem at newspapers / TV channels
- A newspaper is reluctant to publish harsh criticism of its major advertisers.
- The line often gets blurred at newspapers / on TV.
- No known case of this happening with search engines yet?

- Leads to the job of white and black hat **search engine optimization** (organic) and **search engine marketing** (paid).

How are ads ranked?

- Advertisers bid for keywords – sale by auction.
- Open system: Anybody can participate and bid on keywords.
- Advertisers are only charged when somebody clicks on your ad (i.e., CPC)

How does the auction determine an ad's rank and the price paid for the ad?

- Basis is a second price auction, but with twists
- For the bottom line, this is perhaps the most important research area for search engines – computational advertising.
  - Squeezing an additional fraction of a cent from each ad means billions of additional revenue for the search engine.

How are ads ranked?

- First cut: according to bid price – a la Goto
  - Bad idea: open to abuse!
  - Example: query [does my husband cheat?] → ad for divorce lawyer
  - We don't want to show nonrelevant ads.

Instead: rank based on bid price and relevance

- Key measure of ad relevance: clickthrough rate
  - clickthrough rate = CTR = clicks per impressions
- Result: A nonrelevant ad will be ranked low.
  - Even if this decreases search engine revenue short-term
  - Hope: Overall acceptance of the system and overall revenue is maximized if users get useful information.
- Other ranking factors: location, time of day, quality and loading speed of landing page
- The main ranking factor: the query

Google's second price auction

advertiser	bid	CTR	ad rank	rank	paid
A	$4.00	0.01	0.04	4	(minimum)
B	$3.00	0.03	0.09	2	$2.68
C	$2.00	0.06	0.12	1	$1.51
D	$1.00	0.08	0.08	3	$0.51

- bid: maximum bid for a click by advertiser
- CTR: click-through rate: when an ad is displayed, what percentage of time do users click on it? CTR is a measure of relevance.
- ad rank: bid × CTR: this trades off (i) how much money the advertiser is willing to pay against (ii) how relevant the ad is
- rank: rank in auction
- paid: second price auction price paid by advertiser

Google's second price auction

advertiser	bid	CTR	ad rank	rank	paid
A	$4.00	0.01	0.04	4	(minimum)
B	$3.00	0.03	0.09	2	$2.68
C	$2.00	0.06	0.12	1	$1.51
D	$1.00	0.08	0.08	3	$0.51

- Second price auction: The advertiser pays the minimum amount necessary to maintain their position in the auction (plus 1 cent) – related to the Vickrey Auction
- $price_1 \times CTR_1 = bid_2 \times CTR_2$ (this will result in $rank_1 = rank_2$)
- $price_1 = bid_2 \times CTR_2 / CTR_1$
- $p_1 = bid_2 \times CTR_2/CTR_1 = 3.00 \times 0.03/0.06 = 1.50$
- $p_2 = bid_3 \times CTR_3/CTR_2 = 1.00 \times 0.08/0.03 = 2.67$
- $p_3 = bid_4 \times CTR_4/CTR_3 = 4.00 \times 0.01/0.08 = 0.50$

Keywords with high bids

According to http://www.cwire.org/highest-paying-search-terms/

$69.1	mesothelioma treatment options
$65.9	personal injury lawyer michigan
$62.6	student loans consolidation
$61.4	car accident attorney los angeles
$59.4	online car insurance quotes
$59.4	arizona dui lawyer
$46.4	asbestos cancer
$40.1	home equity line of credit
$39.8	life insurance quotes
$39.2	refinancing
$38.7	equity line of credit
$38.0	lasik eye surgery new york city
$37.0	2nd mortgage
$35.9	free car insurance quote

Search ads: A win-win-win?

- The search engine company gets revenue every time somebody clicks on an ad.
- The user only clicks on an ad if they are interested in the ad.
  - Search engines punish misleading and nonrelevant ads.
  - As a result, users are often satisfied with what they find after clicking on an ad.
- The advertiser finds new customers in a cost-effective way.

Not a win-win-win: Keyword arbitrage

- Buy a keyword on Google
- Then redirect traffic to a third party that is paying much more than you are paying Google.
  - E.g., redirect to a page full of ads
- This rarely makes sense for the user.

- (Ad) spammers keep inventing new tricks.
- The search engines need time to catch up with them.

➔ Adversarial Information Retrieval

Not a win-win-win: Violation of trademarks

- Example: geico
- During part of 2005: The search term "geico" on Google was bought by competitors.
- Geico lost this case in the United States.
- Louis Vuitton lost a similar case in Europe.
- See http://google.com/tm_complaint.html
- It's potentially misleading to users to trigger an ad off of a trademark if the user can't buy the product on the site.

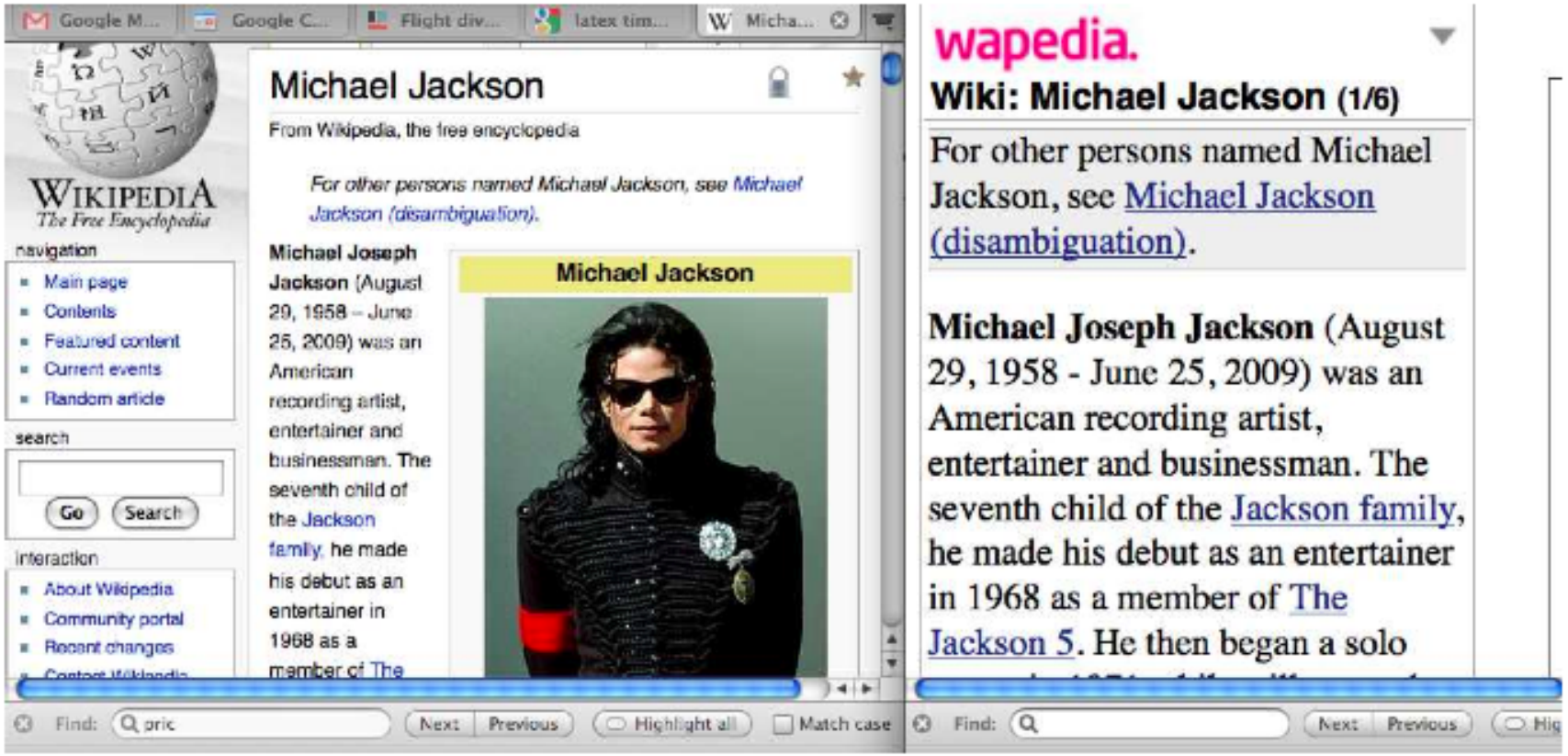
DUPLICATE DETECTION

Duplicate detection

- The web is full of duplicated content.
- More so than many other collections
- Exact duplicates
  - Easy to detect; use hash/fingerprint (e.g., MD5)
- Near-duplicates
  - More common on the web, difficult to eliminate

- For the user, it's annoying to get a search result with near-identical documents.
- Marginal relevance is zero: even a highly relevant document becomes nonrelevant if it appears below a (near-)duplicate.

Near-duplicates: Example



Detecting near-duplicates

- Compute similarity with an edit-distance measure
- We want "syntactic" (as opposed to semantic) similarity.
  - True semantic similarity (similarity in content) is too difficult to compute.
- We do not consider documents near-duplicates if they have the same content, but express it with different words.
- Use similarity threshold θ to make the call "is/isn't a near-duplicate".
- E.g., two documents are near-duplicates if similarity
- > θ = 80%.

Recall: Jaccard coefficient

- A commonly used measure of overlap of two sets
- Let *A* and *B* be two sets
- Jaccard coefficient:

$$\text{JACCARD}(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

$(A \neq \emptyset \text{ or } B \neq \emptyset)$

- JACCARD(*A*,*A*) = 1
- JACCARD(*A*,*B*) = 0 if $A \cap B = 0$
- *A* and *B* don't have to be the same size.
- Always assigns a number between 0 and 1.

Jaccard coefficient: Example

- Three documents:
- d_1: "Jack London traveled to Oakland"
- d_2: "Jack London traveled to the city of Oakland"
- d_3: "Jack traveled from Oakland to London"
- Based on shingles of size 2 (2-grams or bigrams), what are the Jaccard coefficients $J(d_1, d_2)$ and $J(d_1, d_3)$?
- $J(d_1, d_2) = 3/8 = 0.375$
- $J(d_1, d_3) = 0$
- **Note**: very sensitive to dissimilarity

A document as set of shingles

- A shingle is simply a word n-gram.
- Shingles are used as features to measure syntactic similarity of documents.
- For example, for $n = 3$, "a rose is a rose is a rose" would be represented as this set of shingles:
  - { a-rose-is, rose-is-a, is-a-rose }
- We define the similarity of two documents as the Jaccard coefficient of their shingle sets.

Fingerprinting

- We can map shingles to a large integer space $[1..2^m]$ (e.g., $m = 64$) by fingerprinting.
- We use s_k to refer to the shingle's fingerprint in $1..2^m$.

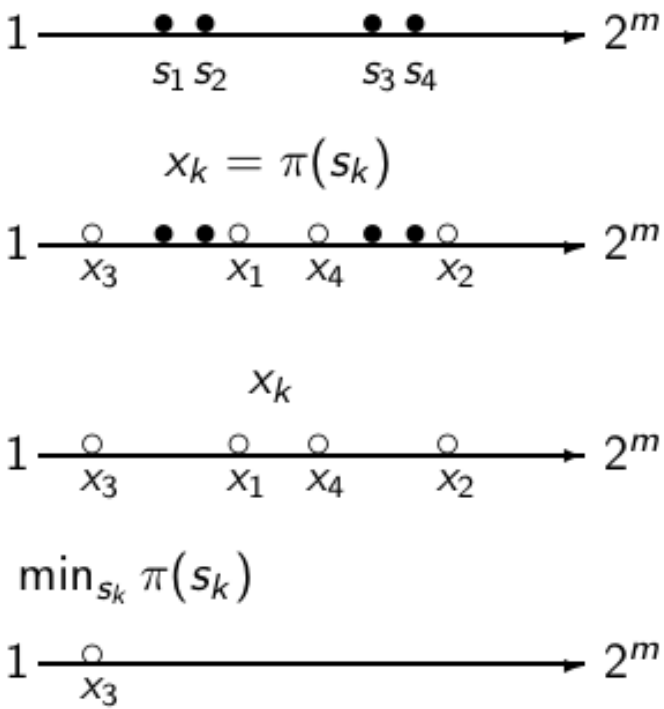
- This doesn't directly help us – we are just converting strings to large integers
- But it **will** help us compute an approximation to the actual Jaccard coefficient quickly

Documents as sketches

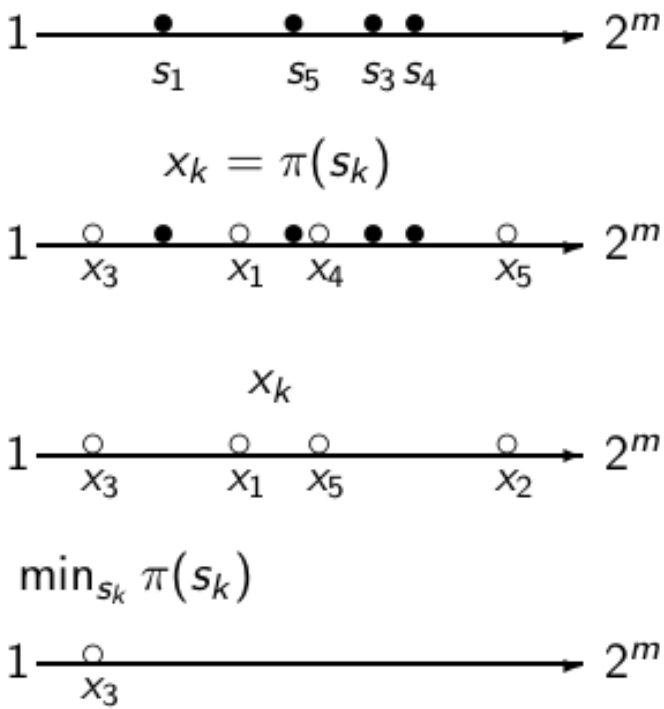
- The number of shingles per document is large, difficult to exhaustively compare
- To make it fast, we use a **sketch**, a subset of the shingles of a document.
- The size of a sketch is, say, n = 200 and is defined by a set of permutations $\pi_1 \ldots \pi_{200}$.
- Each π_i is a random permutation on $1..2^m$
- The sketch of d is defined as:
  - $< \min_{s \in d} \pi_1(s), \min_{s \in d} \pi_2(s), \ldots, \min_{s \in d} \pi_{200}(s) >$ (a vector of 200 numbers).

Deriving a sketch element: a permutation of the original hashes

document 1: $\{s_k\}$

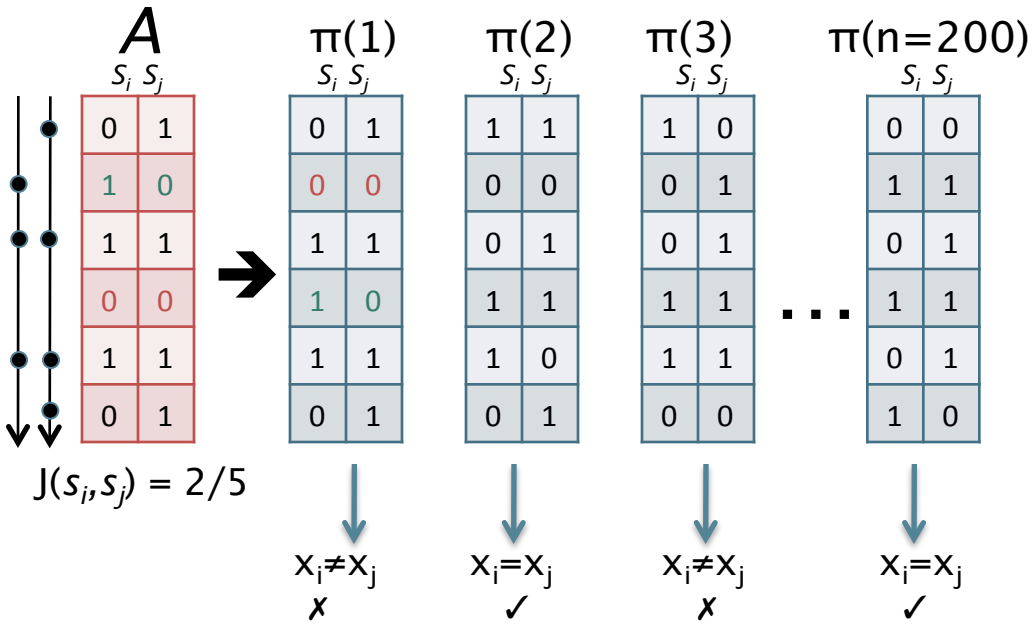


document 2: $\{s_k\}$



We use $\min_{s \in d1} \pi(s) = \min_{s \in d2} \pi(s)$ as a test for: are d_1 and d_2 near-duplicates? In this case: permutation π says: $d_1 \approx d_2$

Proof that $J(S(d_i),s(d_j)) \cong P(x_i^\pi = x_j^\pi)$



We view a matrix *A*:

- 1 column per set of hashes
- Element $A_{i,j}$ = 1 if element i in set S_j is present
- Permutation π(n) is a random reordering of the rows in A
- x_k^π is the first non-zero entry in $\pi(d_k)$, i.e., first shingle present in document k

- Let C_{00} = # of rows in A where both entries are 0, define C_{11}, C_{10}, C_{01} likewise.
- $J(s_i,s_j)$ is then equivalent to $C_{11} / C_{10} + C_{01} + C_{11}$
- $P(x_i=x_j)$ then is equivalent to $C_{11} / C_{10} + C_{01} + C_{11}$

Estimating Jaccard

- Thus, the proportion of successful permutations is the Jaccard coefficient.
  - Permutation π is successful iff $\min_{s \in d1} \pi(s) = \min_{s \in d2} \pi(s)$
- Picking a permutation at random and outputting 1 (successful) or 0 (unsuccessful) is a Bernoulli trial.
- Estimator of probability of success: proportion of successes in n Bernoulli trials. (n = 200)
- Our sketch is based on a random selection of permutations.
- Thus, to compute Jaccard, count the number k of successful permutations for $< d_1, d_2 >$ and divide by n = 200.
- $k/n = k/200$ estimates $J(d_1, d_2)$

Shingling: Summary

- Input: *N* documents
- Choose n-gram size for shingling, e.g., *n* = 5
- Pick 200 random permutations, represented as hash functions
- Compute *N* sketches: 200 × *N* matrix shown on previous slide, one row per permutation, one column per document
- Compute $\frac{N\cdot(N-1)}{2}$ pairwise similarities
- Transitive closure of documents with similarity > θ
- Index only one document from each equivalence class

CRAWLING

What any crawler should do

- Be capable of distributed operation
- Be scalable: need to be able to increase crawl rate by adding more machines
- Fetch pages of higher quality first
- Continuous operation: get fresh version of already crawled pages

How hard can crawling be?

- Web search engines must crawl their documents.
- Getting the content of the documents is easier for many other IR systems.
  - E.g., indexing all files on your hard disk: just do a recursive descent on your file system
- Ok: for web IR, getting the content of the documents takes longer . . .

  . . . because of latency.
- But is that really a design/systems challenge?

Basic crawler operation

- Initialize queue with URLs of known seed pages
- Repeat
  - Take URL from queue
  - Fetch and parse page
  - Extract URLs from page
  - Add URLs to queue
- Fundamental assumption: The web is well linked.

What's wrong with this crawler?

```
urlqueue := (some carefully selected set of
  seed urls)
while urlqueue is not empty:
myurl := urlqueue.getlastanddelete()
mypage := myurl.fetch()
fetchedurls.add(myurl)
newurls := mypage.extracturls()
for myurl in newurls:
if myurl not in fetchedurls and not in
  urlqueue:
urlqueue.add(myurl)
addtoinvertedindex(mypage)
```

What's wrong with the simple crawler

- Scale: we need to distribute.
- We can't index everything: we need to subselect. How?
- Duplicates: need to integrate duplicate detection
- Spam and spider traps: need to integrate spam detection
- Politeness: we need to be "nice" and space out all requests for a site over a longer period (hours, days)
- Freshness: we need to recrawl periodically.
  - Because of the size of the web, we can do frequent recrawls only for a small subset.
  - Again, subselection problem or prioritization

Magnitude of the crawling problem

- To fetch 20,000,000,000 pages in one month . . .

  . . . we need to fetch almost 8000 pages per second!
- Actually: many more since many of the pages we attempt to crawl will be duplicates, unfetchable, spam etc.

What a crawler must do

Be polite

- Don't hit a site too often
- Only crawl pages you are allowed to crawl: robots.txt

Be robust

- Be immune to spider traps, duplicates, very large pages, very large websites, dynamic pages etc

Robots.txt

- Protocol for giving crawlers ("robots") limited access to a website, originally from 1994
- Example:

  User-agent: *

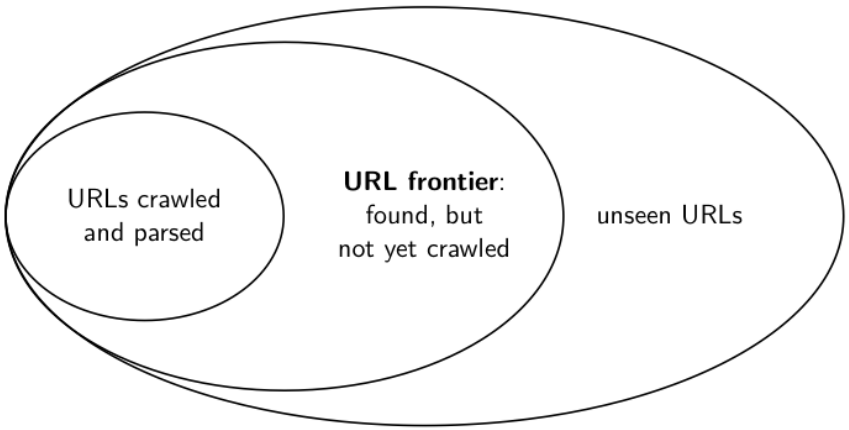
    Disallow: /yoursite/temp/

  User-agent: searchengine

    Disallow: /

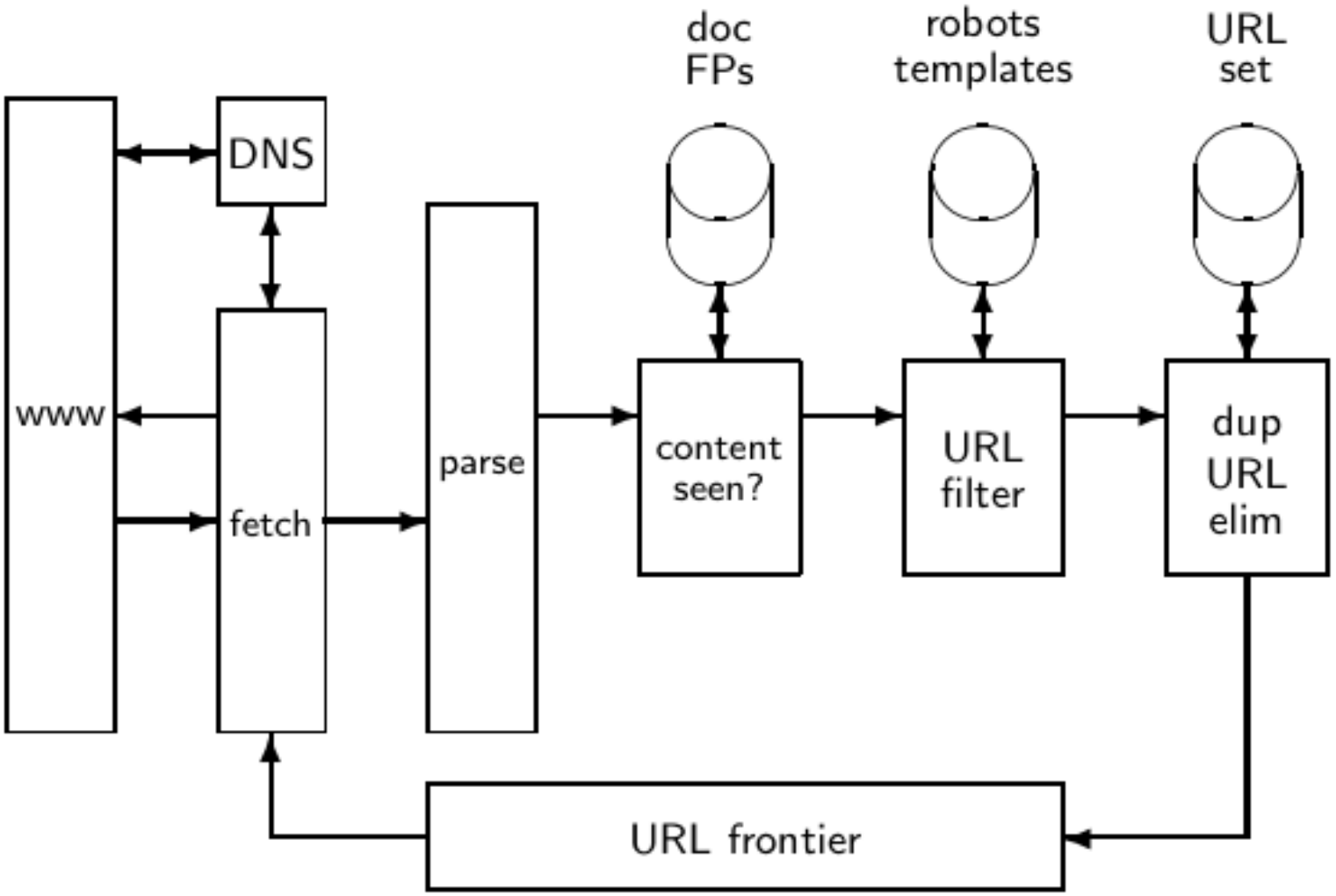
- **Important**: cache the robots.txt file of each site we are crawling

URL Frontier



- The URL frontier is the data structure that holds and manages URLs we've seen, but that have not been crawled yet.
- Can include multiple pages from the same host
- Must avoid trying to fetch them all at the same time
- Must keep all crawling threads busy

Basic Crawling Architecture



URL normalization

- Some URLs extracted from a document are relative URLs.
- E.g., at http://mit.edu, we may have aboutsite.html
  - This is the same as: http://mit.edu/aboutsite.html
- During parsing, we must normalize (expand) all relative URLs.

Content seen

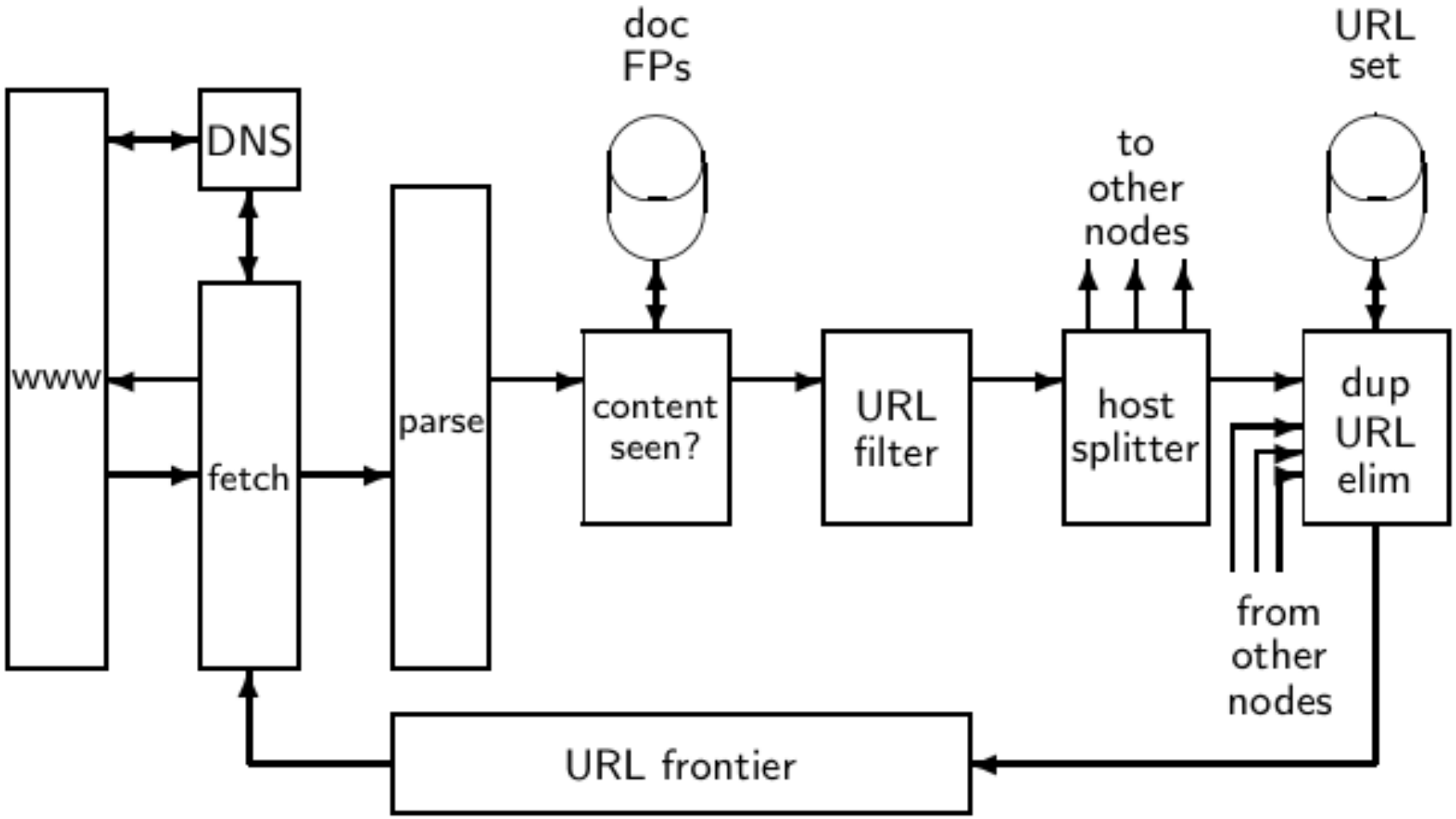
- For each page fetched: check if the content is already in the index
- Check this using document fingerprints or shingles
- Skip documents whose content has already been indexed

- Still need to consider **Freshness**: Crawl some pages (e.g., news sites) more often than others

Distributing the crawler

- Run multiple crawl threads, potentially at different nodes
  - Usually geographically distributed nodes
- Partition hosts being crawled into nodes

Distributed crawling architecture

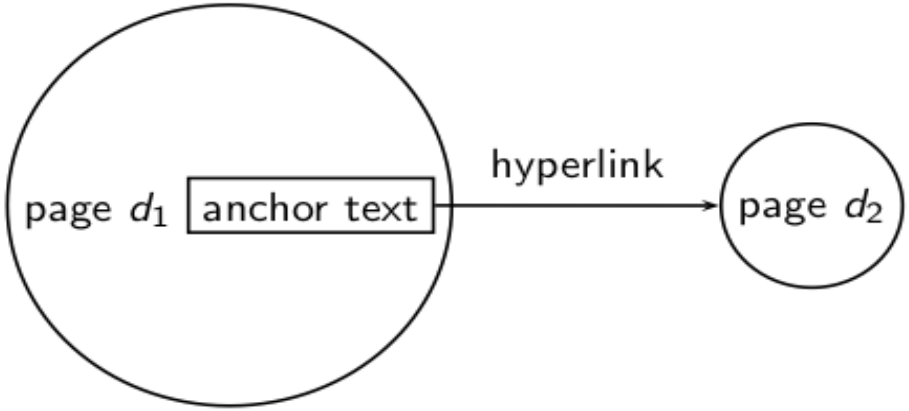


A Crawler Issue: Spider traps

- Malicious server that generates an infinite sequence of linked pages
- Sophisticated spider traps generate pages that are not easily identified as dynamic.

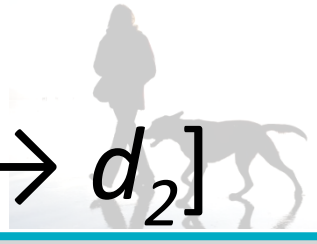
LINK ANALYSIS

The web as a directed graph



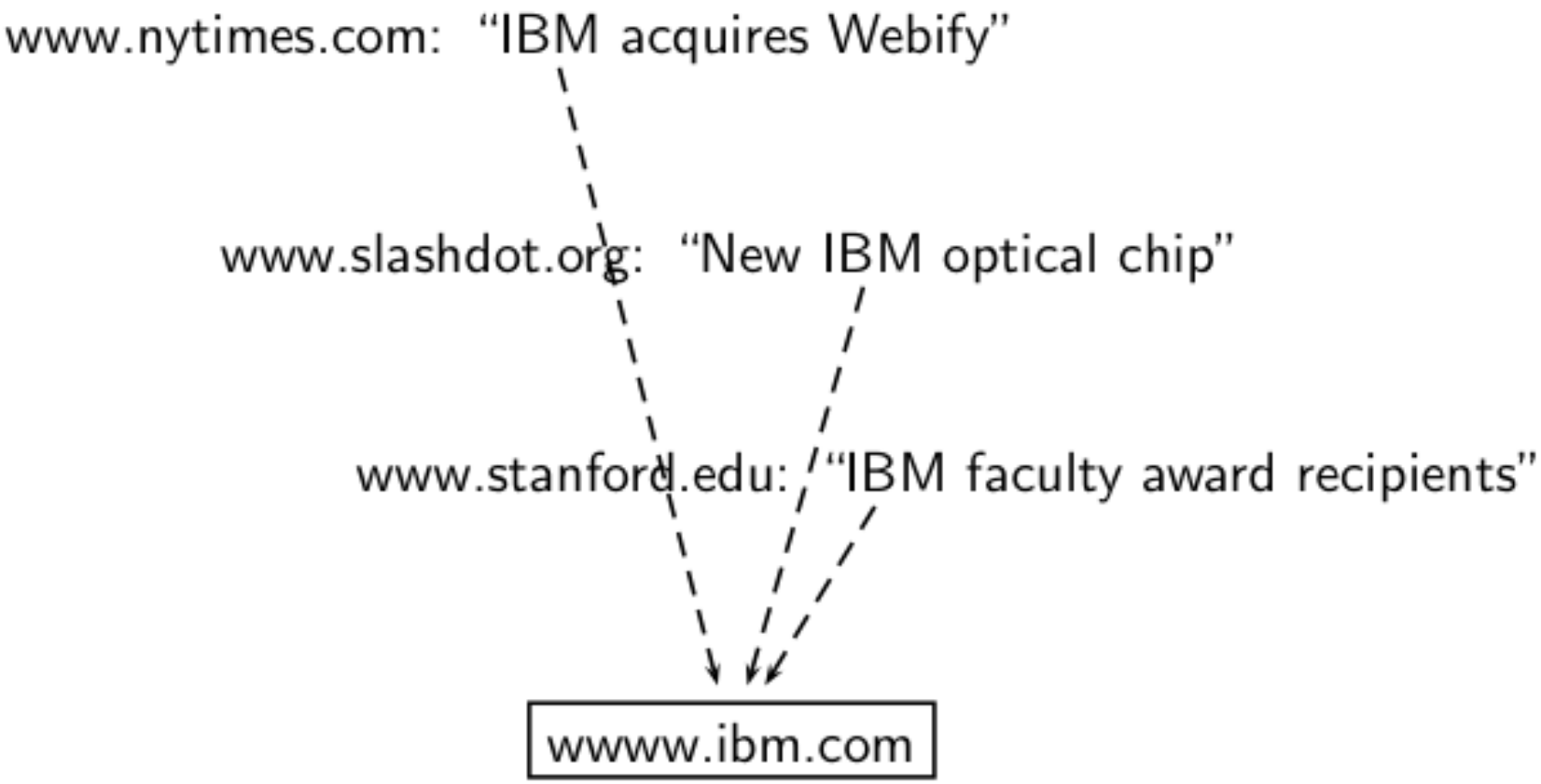
- Assumption 1: A hyperlink is a quality signal.
  - The hyperlink $d_1 \rightarrow d_2$ indicates that d_1's author deems d_2 high-quality and relevant.
- Assumption 2: The anchor text describes the content of d_2.
  - We use anchor text somewhat loosely here for: the text surrounding the hyperlink .
  - Example: "You can find cheap cars here. "
  - Anchor text: "You can find cheap cars here"

[text of d_2] only vs. [text of d_2] + [anchor text → d_2]



- Searching on [text of d_2] + [anchor text → d_2] is often more effective than searching on [text of d_2] only.
- Example: Query *IBM*
  - Matches IBM's copyright page
  - Matches many spam pages
  - Matches IBM wikipedia article
  - May not match IBM home page!
  - … if IBM home page is mostly graphics
- Searching on [anchor text → d_2] is better for the query *IBM*.
  - In this representation, the page with most occurences of *IBM* is www.ibm.com

Anchor text containing *IBM* pointing to www.ibm.com



Indexing anchor text

- Thus: Anchor text is often a better description of a page's content than the page itself.
- Anchor text can be weighted more highly than document text.

  (based on Assumption 1 & 2)

Assumptions underlying PageRank

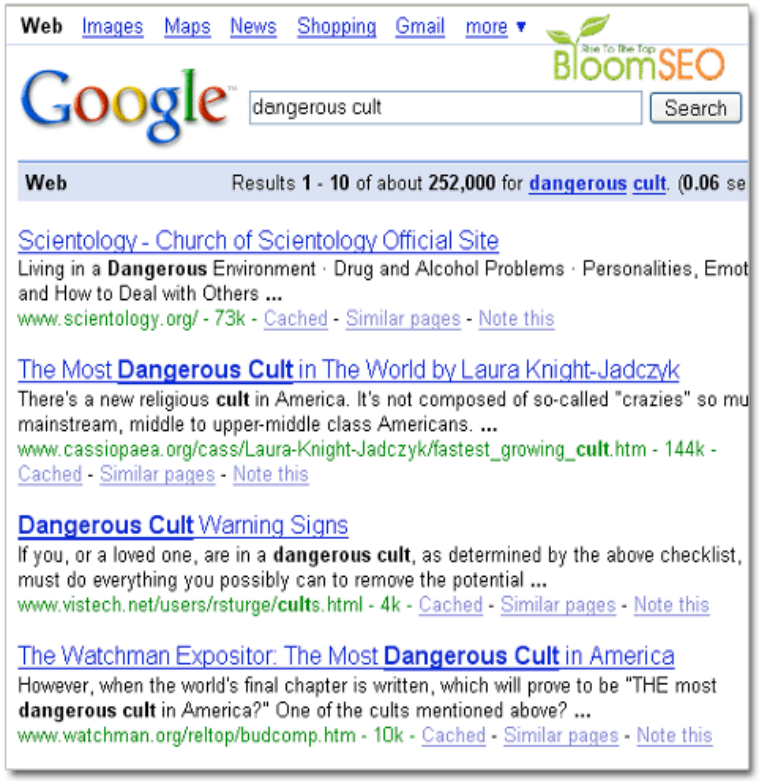
- Assumption 1: A link on the web is a quality signal – the author of the link thinks that the linked-to page is high-quality.
- Assumption 2: The anchor text describes the content of the linked-to page.

- Is Assumption 1 true in general?
- Is Assumption 2 true in general?

Google bombs

- Is a search with "bad" results due to maliciously manipulated anchor text.
- Still some remnants: [dangerous cult] on Google, Bing, Yahoo
  - Coordinated link creation by those who dislike the Church of Scientology
- Google introduced a new weighting function in January 2007

  that fixed many Google bombs.
- Defused Google bombs: [who is a failure?], [evil empire]



PAGERANK

Origins of PageRank: Citation analysis (1)

- Citation analysis: analysis of citations in the scientific literature.
- Example citation: "Miller (2001) has shown that physical activity alters the metabolism of estrogens."
- We can view "Miller (2001)" as a hyperlink linking two scientific articles.
- One application of these "hyperlinks" in the scientific literature:
  - Measure the similarity of two articles by the overlap of other articles citing them.
  - This is called cocitation similarity.
  - Cocitation similarity on the web: Google's "find pages like this" or "Similar" feature.

Citation analysis (2)

- Another application: Citation frequency can be used to measure the impact of an article .
  - Simplest measure: Each article gets one vote – not very accurate.
- On the web: citation frequency = inlink count
  - A high inlink count does not necessarily mean high quality ...
  - ... mainly because of link spam.
- Better measure: weighted citation frequency or citation rank
  - An article's vote is weighted according to its citation impact.
  - Circular? No: can be formalized in a well-defined way

Citation analysis (3)

- Better measure: weighted citation frequency or citation rank, invented in the context of citation analysis by Pinsker and Narin in the 1960s.
- This is basically PageRank.

- We can use the same formal representation for
  - citations in the scientific literature
  - hyperlinks on the web
- Appropriately weighted citation frequency is an excellent measure of quality ...
  - ... both for web pages and for scientific publications.

Definition of PageRank

- The importance of a page is given by the importance of the pages that link to it.

$$x_i = \sum_{j \in B_i} \frac{1}{N_j} x_j$$

importance of page i

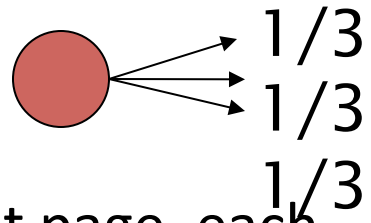
pages j that link to page i

number of outlinks from page j

importance of page j

Pagerank scoring

- Imagine a browser doing a random walk on web pages:
  - Start at a random page
  - At each step, follow one of the *n* links on that page, each with 1/*n* probability

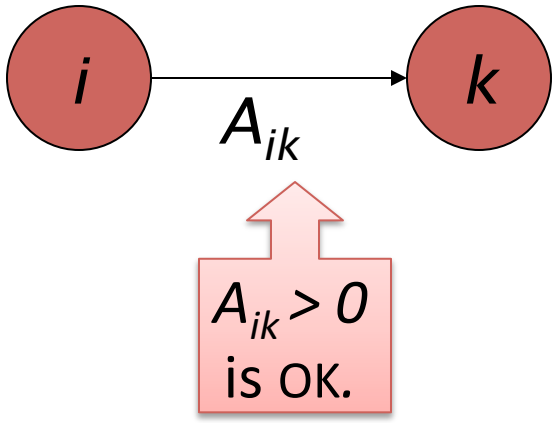


- Do this repeatedly. Use the "long-term visit rate" as the page's score
- This is a global score for the page, based on the topology of the network.
- Think of it as ***g(d)*** from Chapter 7

Markov chains

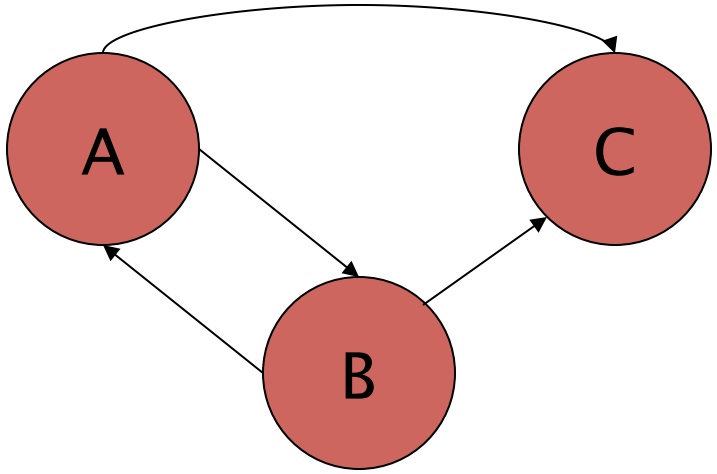
A Markov chain consists of *n* states, plus an *n*×*n* transition probability matrix A.

- At each step, we are in exactly one of the states.
- For $1 \leq i,k \leq n$, the matrix entry A_{ik} tells us the probability of k being the next state, given we are currently in state *i*.
- **Memorylessness property**: The next state depends only at the current state (first order Markov Chain)



Markov chains

- Clearly, for all i, $\sum_{k=1}^{n} A_{ik} = 1.$
- Markov chains are abstractions of random walks



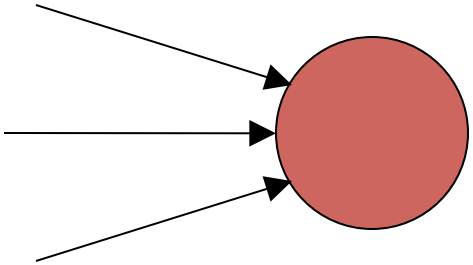
Try this: Calculate the matrix A_{ik} using 1/n possibility

A_{ik}:

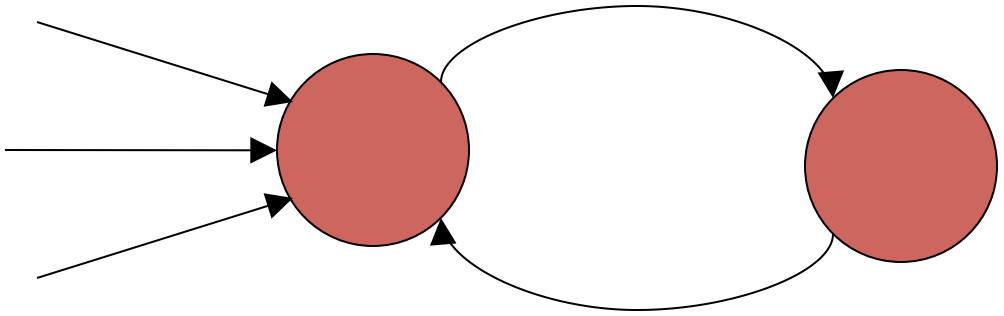
	A	B	C
A			
B			
C			

Not quite enough

- The web is full of dead ends.
  - What sites have dead ends?
  - Our random walk can get stuck.



Dead End



Spider Trap

Teleporting

- At each step, with probability 10%, teleport to a random web page
- With remaining probability (90%), follow a random link on the page
  - If a dead-end, stay put in this case

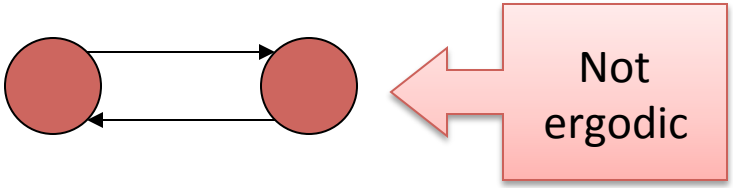
Follow!

Teleport!

$$\vec{rank} = (1-a)A \times \vec{rank} + \alpha \left[\frac{1}{N}\right] N \times 1$$

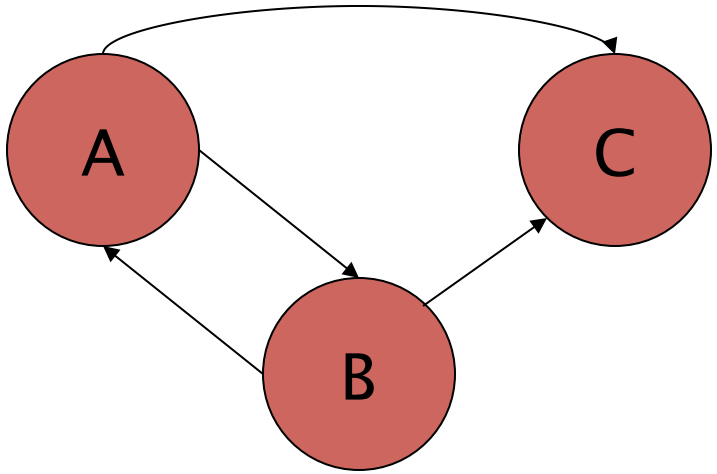
Ergodic Markov chains

- A Markov chain is ergodic if
  - you have a path from any state to any other
  - you can be in any state at every time step, with non-zero probability



  - With teleportation, our Markov chain is ergodic
  - Theorem: With an ergodic Markov chain, there is a stable long term visit rate.

Markov chains (2nd Try)



Try this: Calculate the matrix A_{ik} using a 10% chance of teleportation

A_{ik}:	A	B	C
A			
B			
C			

Probability vectors

- A probability (row) vector **x** = $(x_1, \ldots x_n)$ tells us where the walk is at any point
- E.g., (000…1…000) means we're in state *i*.
  1 *i* *n*

More generally, the vector x = $(x_1, \ldots x_n)$ means The walk is in state *i* with probability x_i.

$$\sum_{i=1}^{n} x_i = 1.$$

Change in probability vector

- If the probability vector is $\mathbf{x} = (x_1, \ldots x_n)$ at this step, what is it at the next step?
- Recall that row *i* of the transition prob. Matrix **A** tells us where we go next from state *i*.
- So from **x**, our next state is distributed as **xA**.

Pagerank algorithm

- Regardless of where we start, we eventually reach the steady state a
  - Start with any distribution (say x=(10…0))
  - After one step, we're at xA
  - After two steps at xA^2 , then xA^3 and so on.
  - "Eventually" means for "large" k, $xA^k = a$

- Algorithm: multiply x by increasing powers of A until the product looks stable

Steady State

- For any ergodic Markov chain, there is a unique long-term visit rate for each state
  - Over a long period, we'll visit each state in proportion to this rate
  - It doesn't matter where we start

Eigenvector formulation

- The flow equations can be written

$$\mathbf{r} = \mathbf{A}\mathbf{r}$$

- So the rank vector is an eigenvector of the adjacency matrix
  - In fact, it's the first or principal eigenvector, with corresponding eigenvalue 1

Pagerank summary

- Pre-processing:
  - Given graph of links, build matrix **A**
  - From it compute **a**
  - The pagerank a_i is a scaled number between 0 and 1
- Query processing:
  - Retrieve pages meeting query
  - Rank them by their pagerank
  - Order is query-*independent*

PageRank issues

- Real surfers are not random surfers.
  - Examples of nonrandom surfing: back button, short vs. long paths, bookmarks, directories – and search!
  - → Markov model is not a good model of surfing.
  - But it's good enough as a model for our purposes.
- Simple PageRank ranking (as described on previous slide) produces bad results for many pages.
  - Consider the query [video service].
  - The Yahoo home page (i) has a very high PageRank and (ii) contains both *video* and *service*.
  - If we rank all Boolean hits according to PageRank, then the Yahoo home page would be top-ranked.
  - Clearly not desirable.

How important is PageRank?

- Frequent claim: PageRank is the most important component of web ranking.
- The reality:
  - There are several components that are at least as important: e.g., anchor text, phrases, proximity, tiered indexes ...
  - Rumor has it that PageRank in his original form (as presented here) now has a negligible impact on ranking!
  - However, variants of a page's PageRank are still an essential part of ranking.
  - Addressing link spam is difficult and crucial.

NUS National University of Singapore

Summary

- Chapters 19, 20 and 21 of IIR
- Resources
  - Paper on Mercator crawler by Heydon et al.
  - Robot exclusion standard
  - American Mathematical Society article on PageRank (popular science style)
  - Google's official description of PageRank: "PageRank reflects our view of the importance of web pages by considering more than 500 million variables and 2 billion terms. Pages that believe are important pages receive a higher PageRank and are more likely to appear at the top of the search results"