CS3245

# Information Retrieval

12

## Lecture 12: Crawling and Link Analysis

# Last Time

Chapter 11

1. Probabilistic Approach to Retrieval / Basic Probability Theory
2. Probability Ranking Principle
3. Binary Independence Model, BestMatch25 (Okapi)

Chapter 12

1. Language Models for IR

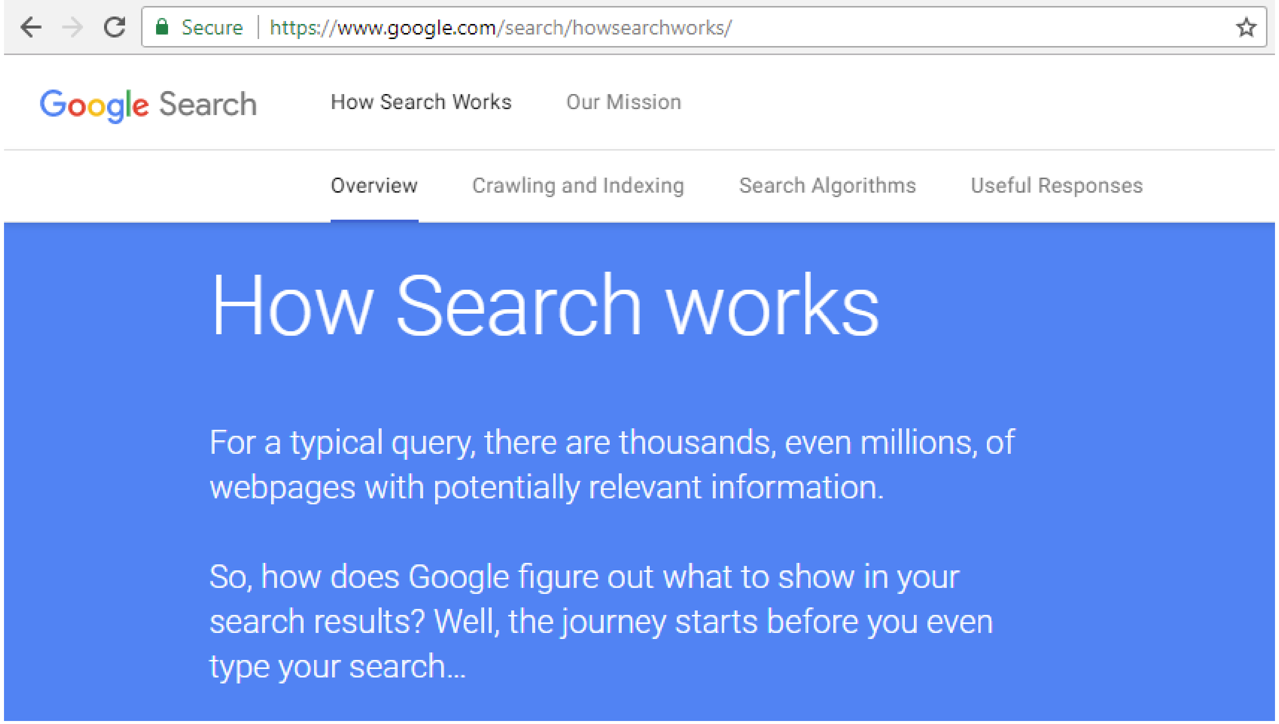
# Today – The Web

## Chapter 20

- Crawling

## Chapter 21

- Link Analysis



### How Search works

For a typical query, there are thousands, even millions, of webpages with potentially relevant information.

So, how does Google figure out what to show in your search results? Well, the journey starts before you even type your search...

# CRAWLING

# 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
- Bandwidth, latency…
  - Not just on crawler side, but also on the web server side.

# Magnitude of the crawling problem

## How Google searches 30 trillion web pages, 100 billion times a month

JOHN KOETSIER, TUNE MARCH 1, 2013 12:43 PM

TAGS: FEATURED, GOOGLE, INDEX, SEARCH ENGINE, WEB SEARCH

To fetch 30T pages in one month, we need to fetch almost 11M pages per second!

Actually many more since many of the pages we attempt to crawl will be duplicates, unfetchable, spam etc.

http://venturebeat.com/2013/03/01/how-google-searches-30-trillion-web-pages-100-billion-times-a-month/

# 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
  - Call the indexer to index the page

- 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)

  indexer.index(mypage)
```

# What’s wrong with the simple crawler?

- Politeness: we need to be “nice” and space out all requests for a site over a longer period (hours, days)
- We can’t index everything: we need to subselect. How?
- 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
- Duplicates: need to integrate duplicate detection
- Scale: we need to distribute.
- Spam and spider traps: need to integrate spam detection

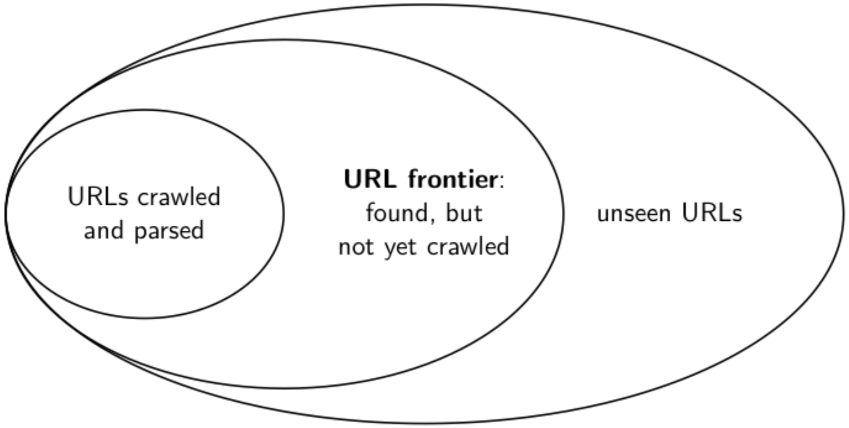
# Robots.txt

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

```
 User-agent: *
     Disallow: /yoursite/temp/
 User-agent: searchengine
     Disallow: /
```

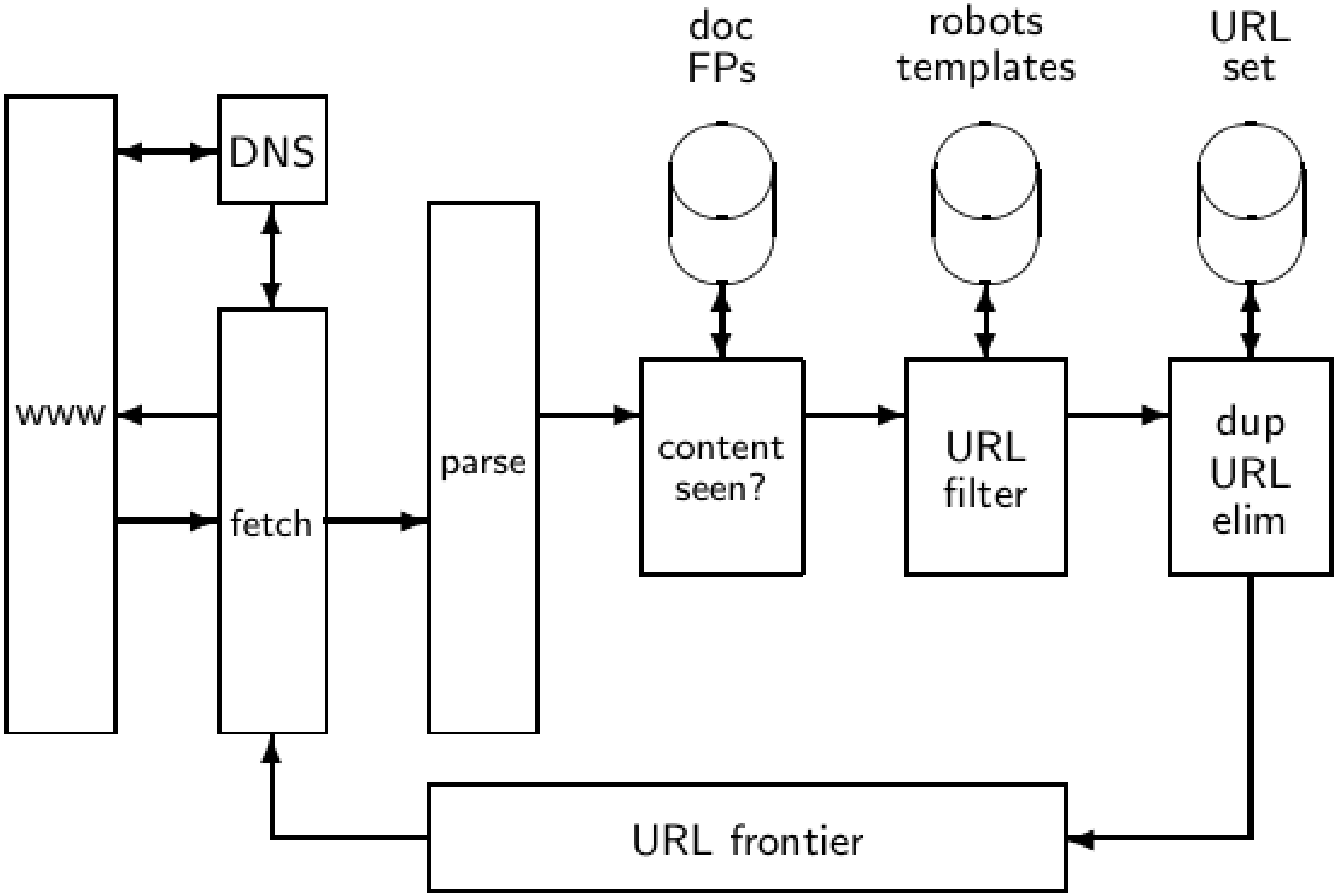
- **Important**: cache the robots.txt file of each site we are crawling

# 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



# 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

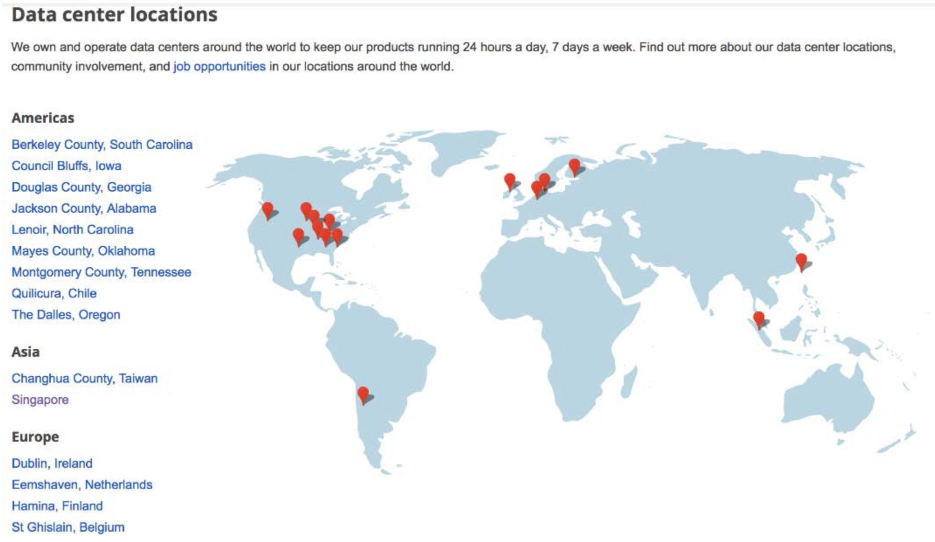
# URL seen

- 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.

- Duplicate elimination
  - 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
- Map target hosts to crawl to nodes



**Data center locations**

We own and operate data centers around the world to keep our products running 24 hours a day, 7 days a week. Find out more about our data center locations, community involvement, and job opportunities in our locations around the world.

**Americas**

Berkeley County, South Carolina
Council Bluffs, Iowa
Douglas County, Georgia
Jackson County, Alabama
Lenoir, North Carolina
Mayes County, Oklahoma
Montgomery County, Tennessee
Quilicura, Chile
The Dalles, Oregon

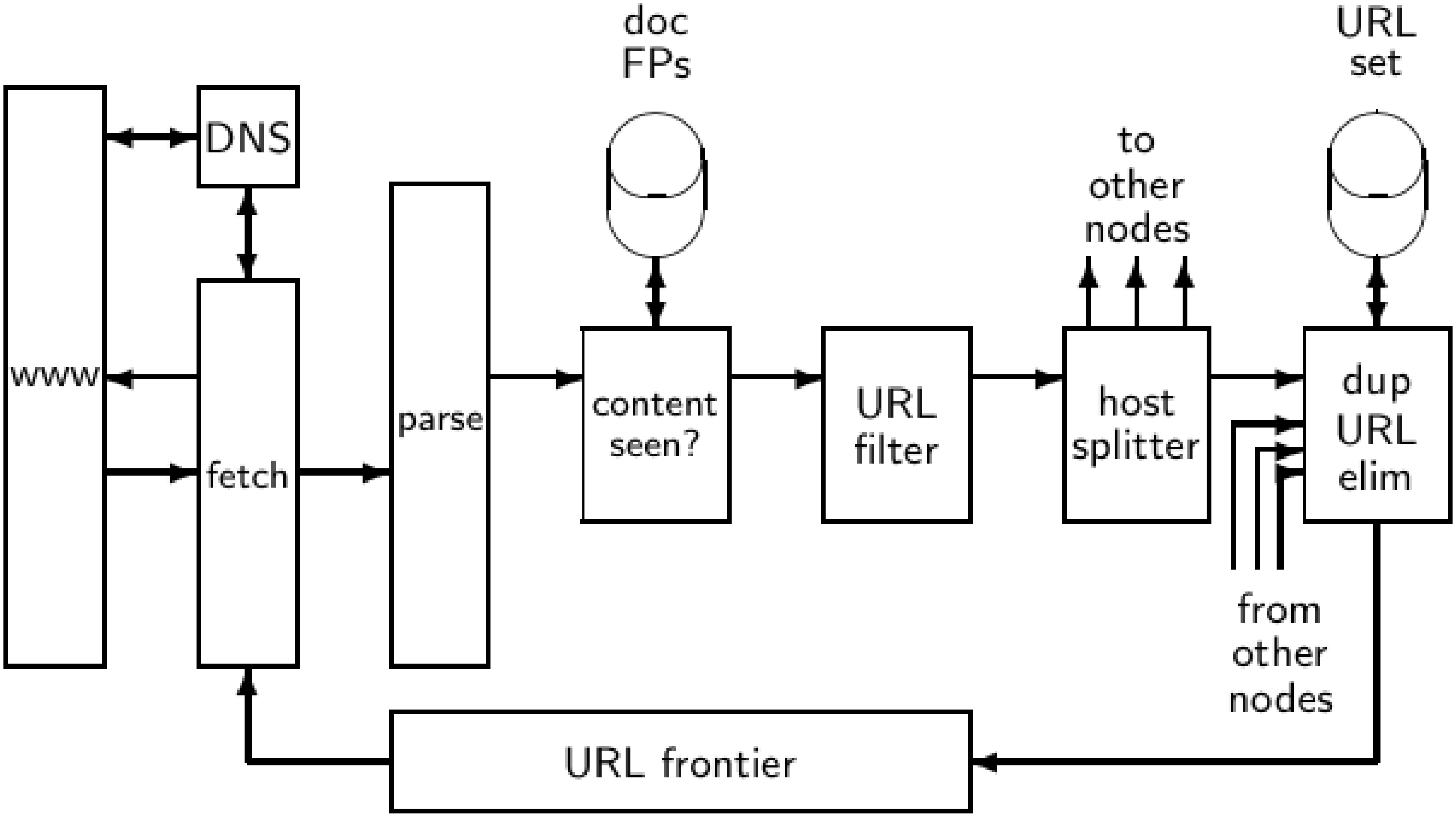
**Asia**

Changhua County, Taiwan
Singapore

**Europe**

Dublin, Ireland
Eemshaven, Netherlands
Hamina, Finland
St Ghislain, Belgium

# Distributed crawling architecture

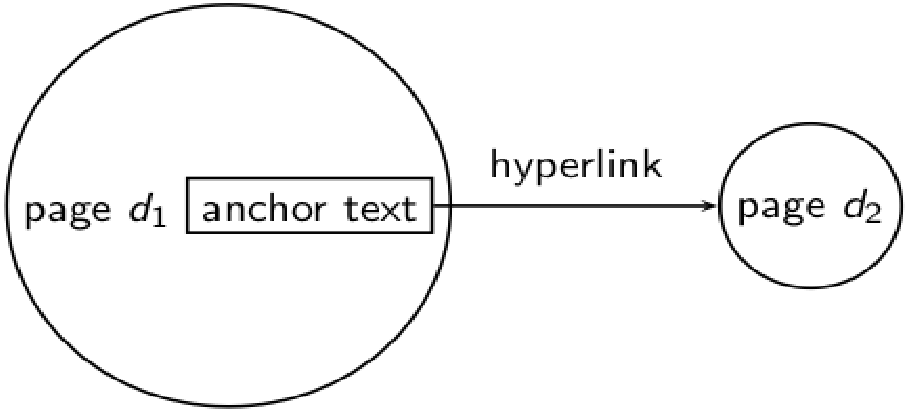


# A Crawler Issue: Spider traps

- A set of webpages that cause the clawer to go into an infinite loop or crash
  - May be created intentionally or unintentionally
- Sophisticated spider traps are not easily identified as dynamic.

# LINK ANALYSIS

# The web as a directed graph

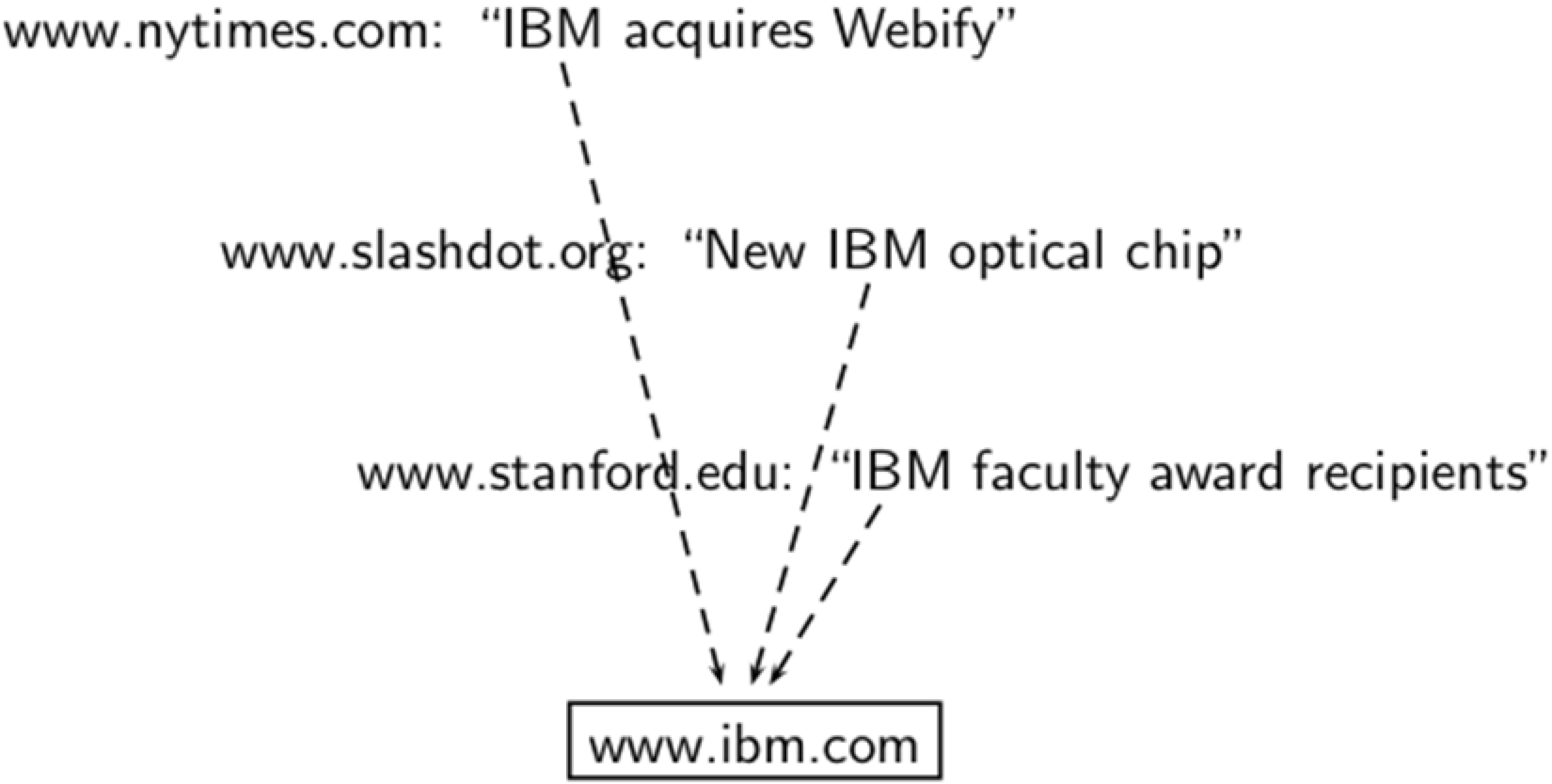


- Assumption 1: A hyperlink is a quality signal.
  - The hyperlink $d_1 \rightarrow d_2$ indicates that $d_1$'s author deems $d_2$ high-quality and relevant.
- Assumption 2: The anchor text describes the content of $d_2$.
  - We use anchor text somewhat loosely here for: the text surrounding the hyperlink .
  - Example: "You can find cheap cars <a href =http://…>here</a>. "
  - 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 occurrences of *IBM* is www.ibm.com

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



www.nytimes.com: “IBM acquires Webify”

www.slashdot.org: “New IBM optical chip”

www.stanford.edu: “IBM faculty award recipients”

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)

Aside: we use this in academic networks too. Citances (“sentences that cite another work”) and their host document sections are more useful in categorizing a target work. (*cf* Assumption 1)

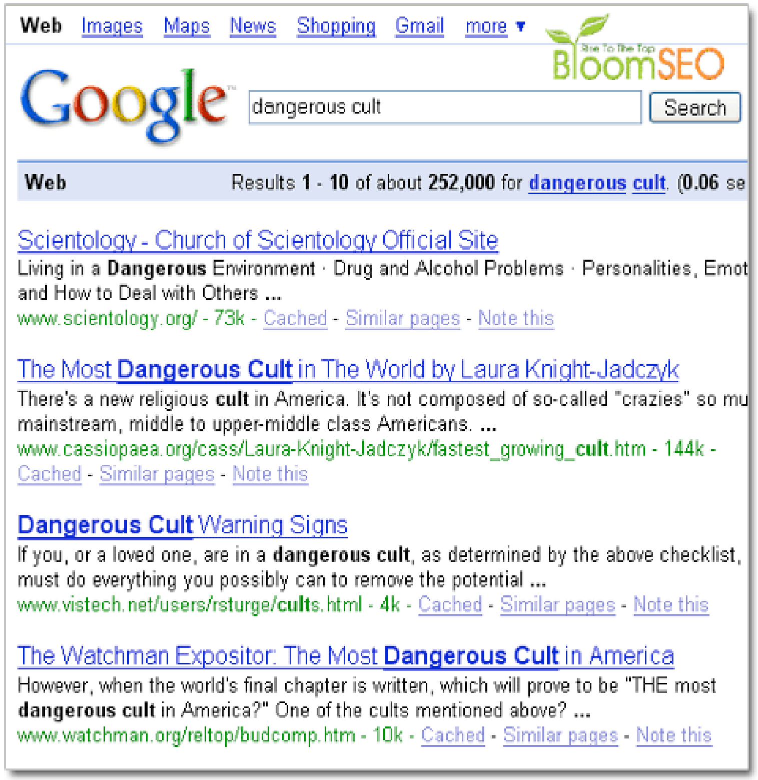
# Assumptions underlying PageRank

- 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.
- E.g., [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]



Web Results **1** - **10** of about **252,000** for **dangerous cult**. (**0.06** se

Scientology - Church of Scientology Official Site
Living in a **Dangerous** Environment · Drug and Alcohol Problems · Personalities, Emot
and How to Deal with Others **...**
www.scientology.org/ - 73k - Cached - Similar pages - Note this

The Most **Dangerous Cult** in The World by Laura Knight-Jadczyk
There's a new religious **cult** in America. It's not composed of so-called "crazies" so mu
mainstream, middle to upper-middle class Americans. **...**
www.cassiopaea.org/cass/Laura-Knight-Jadczyk/fastest_growing_**cult**.htm - 144k -
Cached - Similar pages - Note this

**Dangerous Cult** Warning Signs
If you, or a loved one, are in a **dangerous cult**, as determined by the above checklist,
must do everything you possibly can to remove the potential **...**
www.vistech.net/users/rsturge/**cults**.html - 4k - Cached - Similar pages - Note this

The Watchman Expositor: The Most **Dangerous Cult** in America
However, when the world's final chapter is written, which will prove to be "THE most
**dangerous cult** in America?" One of the cults mentioned above? **...**
www.watchman.org/reltop/budcomp.htm - 10k - Cached - Similar pages - Note this

# 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 = in-link count
  - A high inlink count does not necessarily mean high quality ... ... mainly because of link spam.

# Citation analysis – 3

- Better measure: weighted citation frequency or citation rank, by Pinsker and Narin in the 1960s.
  - An article's vote is weighted according to its citation impact.
  - This is basically PageRank.
- We can use the same formal representation for
  - citations in the scientific literature
  - hyperlinks on the web

# Definition of PageRank

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

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

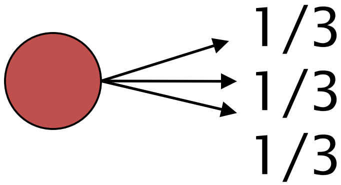
importance of page $i$

importance of page $j$

pages $j$ that link to page $i$

number of outlinks from page $j$

# Pagerank scoring

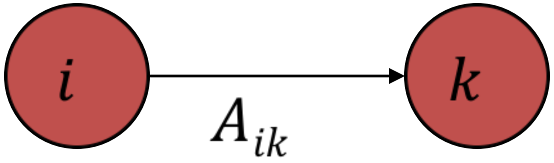


- Imagine a browser doing a random walk on web pages:
  - Start at a random page
  - At each step, follow one of the $n$ links on that page, each with $1/n$ probability
- Do this repeatedly. Use the "long-term visit rate" as the page's score
- This is a global score for the page, based on the topology of the network.
- Think of it as $g(d)$ from Chapter 7

# Markov chains

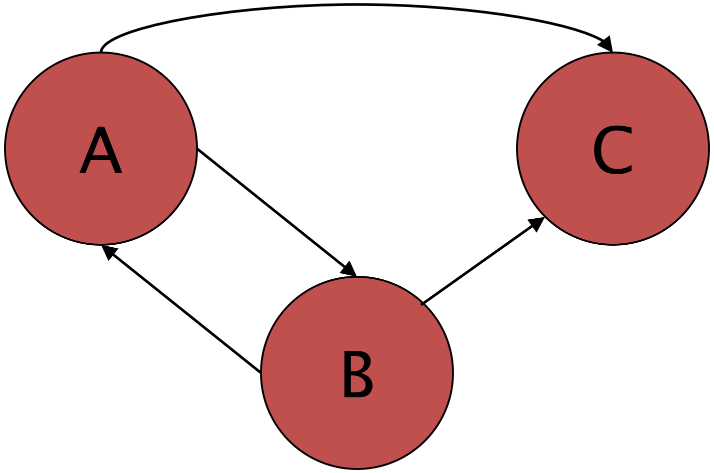
A Markov chain consists of $n$ states, plus an $n \times n$ transition probability matrix $A$.

- 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

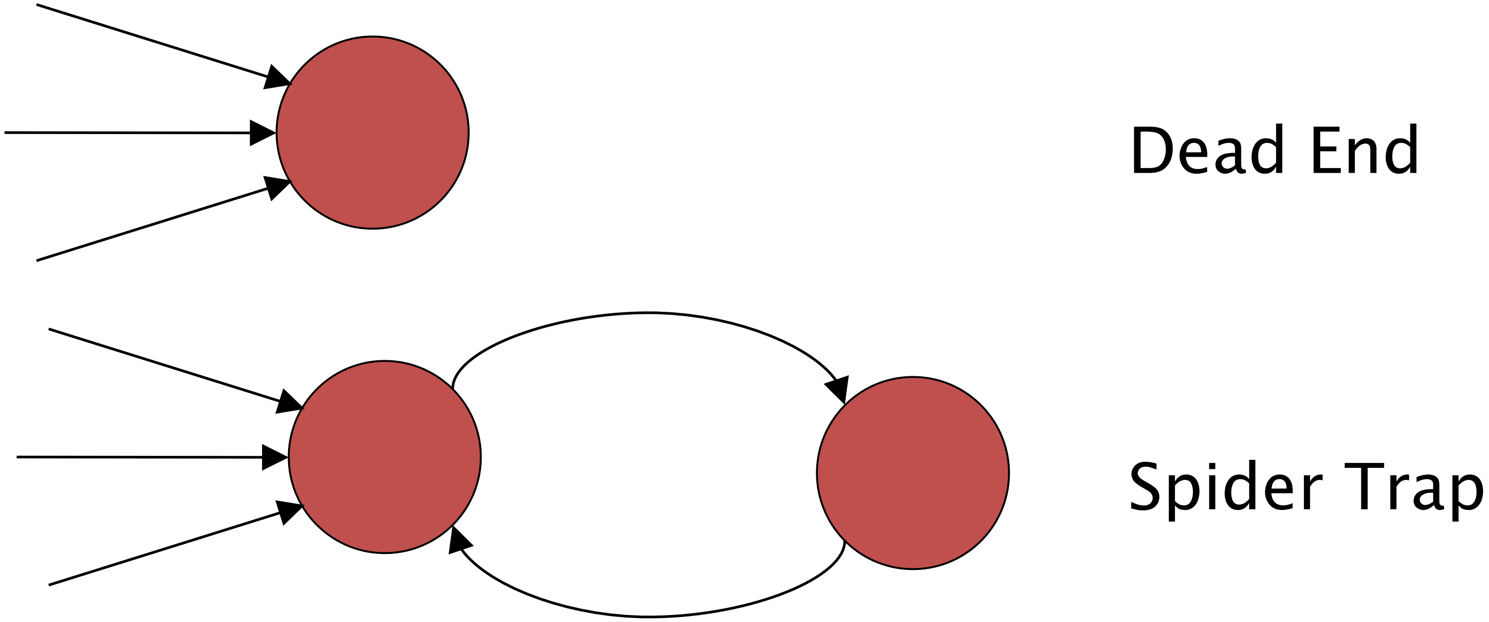


Try this: Calculate the matrix $A_{ik}$ using $1/n$ probability

| $A_{ik}$: | 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.



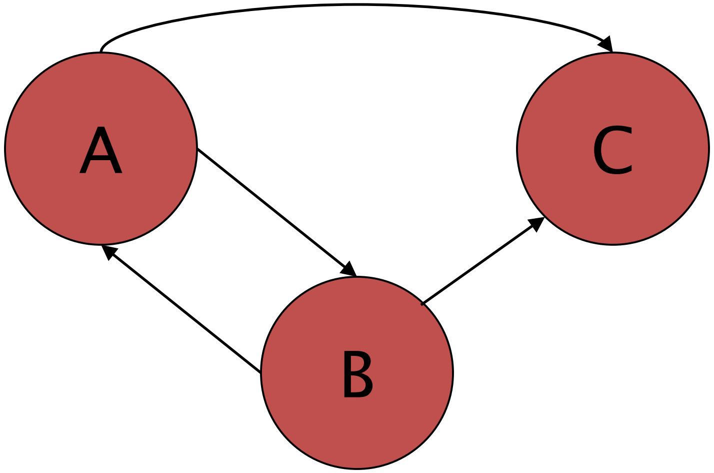
Dead End

Spider Trap

# Teleporting

- At each step, with probability α (e.g., 10%), teleport to a random web page
- With remaining probability (90%), follow a random link on the page
  - If the page is a dead-end, just stay put.

# Markov chains (2nd Try)

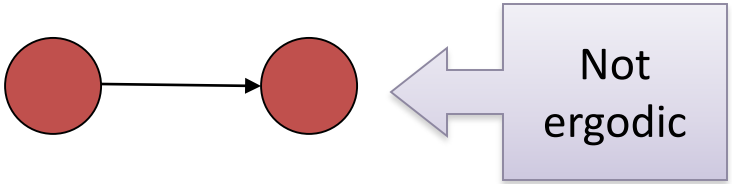


Try this: Calculate the matrix $A_{ik}$ using a 10% chance of teleportation

| $A_{ik}$: | A | B | C |
|---|---|---|---|
| A | | | |
| B | | | |
| C | | | |

# 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



Not ergodic

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

# Probability vectors

- A probability (row) vector $\boldsymbol{x} = (x_1, \ldots\ x_n)$ tells us where the walk is at any point
- E.g., $(0_1 00 \ldots 1_i \ldots 000_n)$ means we're in state $i$.

More generally, the vector $\boldsymbol{x} = (x_1, \ldots\ x_n)$ 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 $\boldsymbol{x} = (x_1, \dots x_n)$ at this step, what is it at the next step?
- Recall that the transition prob. Matrix $\boldsymbol{A}$ tells us where we go next.
- So from $\boldsymbol{x}$, our next state is distributed as $\boldsymbol{xA}$.

$$\vec{x_0} = (1\ 0\ 0)$$

$$\vec{x_0}\boldsymbol{A} = \left(\ 1/6 \quad 2/3 \quad 1/6\ \right) = \vec{x_1}$$

$$\vec{x_1}\boldsymbol{A} = \left(\ 1/3 \quad 1/3 \quad 1/3\ \right) = \vec{x_2}$$

$$\boldsymbol{A} = \begin{array}{|c|c|c|c|} \hline & S_1 & S_2 & S_3 \\ \hline S_1 & 1/6 & 2/3 & 1/6 \\ \hline S_2 & 5/12 & 1/6 & 5/12 \\ \hline S_3 & 1/6 & 2/3 & 1/6 \\ \hline \end{array}$$

# 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

# Pagerank algorithm

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

- Algorithm: multiply $\boldsymbol{x}$ by increasing powers of $\boldsymbol{A}$ until the product looks stable.

# At the steady state a…

$$a = aA$$

- 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 $\boldsymbol{A}$
  - From it compute $\boldsymbol{a}$
  - 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 ...
  - PageRank in its original form (as presented here) 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.

# Summary

- Crawling – Obtaining documents for indexing
  - Need to be polite
- PageRank – A $G(d)$ for asymmetrically linked documents

- Chapters 20 and 21 of IIR
- Resources
  - Paper on Mercator crawler by Heydon et al.
  - Robot exclusion standard

"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"