Information retrieval

Lecture 8

Special thanks to
Andrei Broder, IBM
Krishna Bharat, Google
for sharing some of the slides to follow.

Top Online Activities (Jupiter Communications, 2000)

(a) Source: Jupiter Communications.

Search on the Web

- Corpus:The publicly accessible Web: static + dynamic
- Goal: Retrieve high quality results relevant to the user's need
  - (not docs!)
- Need
  - Informational – want to learn about something (~40%)

    Low hemoglobin
  - Navigational – want to go to that page (~25%)

    United Airlines
  - Transactional – want to do something (web-mediated) (~35%)
    - Access a service

      Tampere weather
    - Downloads

      Mars surface images
    - Shop

      Nikon CoolPix
  - Gray areas
    - Find a good hub

      Car rental Finland
    - Exploratory search "see what's there"

Results

- Static pages (documents)
  - text, mp3, images, video, ...
- Dynamic pages = generated on request
  - data base access
  - “the invisible web”
  - proprietary content, etc.

Scale

- Immense amount of content
  - 10+B static pages, doubling every 8-12 months
  - Lexicon Size: 10s-100s of millions of words
- Authors galore (1 in 4 hosts run a web server)
- http://news.netcraft.com/archives/web_server_survey.html contains an ongoing survey
- Over 50 million hosts and counting
  - One for every person in Italy

Diversity

- Languages/Encodings
  - Hundreds (thousands ?) of languages, W3C encodings: 55 (Jul01) [W3C01]
  - Home pages (1997): English 82%, Next 15: 13% [Babe97]
  - Google (mid 2001): English: 53%, JGCFSKRIP: 30%
- Document & query topic

Popular Query Topics (from 1 million Google queries, Apr 2000)

Arts	14.6%	Arts: Music	6.1%
Computers	13.8%	Regional: North America	5.3%
Regional	10.3%	Adult: Image Galleries	4.4%
Society	8.7%	Computers: Software	3.4%
Adult	8%	Computers: Internet	3.2%
Recreation	7.3%	Business: Industries	2.3%
Business	7.2%	Regional: Europe	1.8%
...	...	...	...

Rate of change

[Cho00] 720K pages from 270 popular sites sampled daily from Feb 17 – Jun 14, 1999

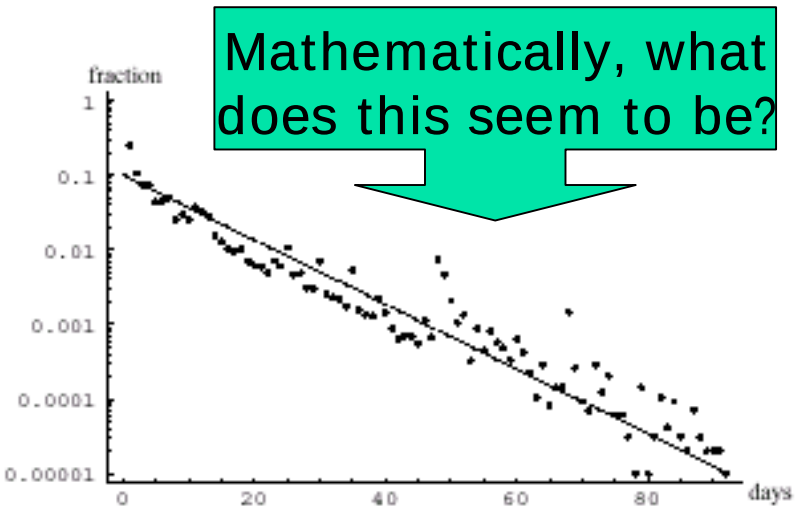


Figure 11: Change intervals for pages with the average change interval of 10 days

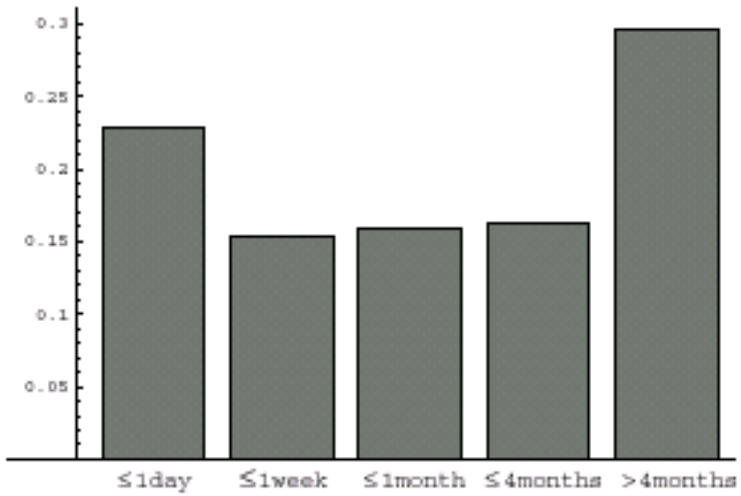


Figure 12: Percentage of pages with given average interval of change

Web idiosyncrasies

- Distributed authorship
  - Millions of people creating pages with their own style, grammar, vocabulary, opinions, facts, falsehoods …
  - Not all have the purest motives in providing high-quality information - commercial motives drive “spamming” - 100s of millions of pages.
  - The open web is largely a marketing tool.
    - IBM’s home page does not contain *computer*.

Other characteristics

- Significant duplication
  - Syntactic - 30%-40% (near) duplicates [Brod97, Shiv99b]
  - Semantic - ???
- High linkage
  - ~ 8 links/page in the average
- Complex graph topology
  - Not a small world; bow-tie structure [Brod00]
- More on these corpus characteristics later
  - how do we measure them?

Web search users

- Ill-defined queries
  - Short
    - AV 2001: 2.54 terms avg, 80% < 3 words)
  - Imprecise terms
  - Sub-optimal syntax (80% queries without operator)
  - Low effort
- Wide variance in
  - Needs
  - Expectations
  - Knowledge
  - Bandwidth
- Specific behavior
  - 85% look over one result screen only (mostly above the fold)
  - 78% of queries are not modified (one query/session)
  - Follow links – "the scent of information" ...

Evolution of search engines

- First generation -- use only “on page”, text da
  - Word frequency, language

1995-1997 AV, Excite, Lycos, etc

- Second generation -- use off-page, web-specific data
  - Link (or connectivity) analysis
  - Click-through data (What results people click
  - Anchor-text (How people refer to this page)

From 1998. Made popular by Google but everyone now

- Third generation -- answer “the need behind the query”
  - Semantic analysis -- what is this about?
  - Focus on user need, rather than on query
  - Context determination
  - Helping the user
  - Integration of search and text analysis

Still experimental

First generation ranking

- Extended Boolean model
  - Matches: exact, prefix, phrase,...
  - Operators: AND, OR, AND NOT, NEAR, ...
  - Fields: TITLE:, URL:, HOST:,...
  - AND is somewhat easier to implement, maybe preferable as default for short queries
- Ranking
  - TF like factors: TF, explicit keywords, words in title, explicit emphasis (headers), etc
  - IDF factors: IDF, total word count in corpus, frequency in query log, frequency in language

Second generation search engine

- Ranking -- use off-page, web-specific data
  - Link (or connectivity) analysis
  - Click-through data (What results people click on)
  - Anchor-text (How people refer to this page)
- Crawling
  - Algorithms to create the best possible corpus

Connectivity analysis

- Idea: mine hyperlink information in the Web
- Assumptions:
  - Links often connect related pages
  - A link between pages is a recommendation

    "people vote with their links"

Third generation search engine: answering “the need behind the query”

- Query language determination
- Different ranking
  - (if query Japanese do not return English)
- Hard & soft matches
  - Personalities (triggered on names)
  - Cities (travel info, maps)
  - Medical info (triggered on names and/or results)
  - Stock quotes, news (triggered on stock symbol)
  - Company info, …
- Integration of Search and Text Analysis

Answering "the need behind the query" Context determination

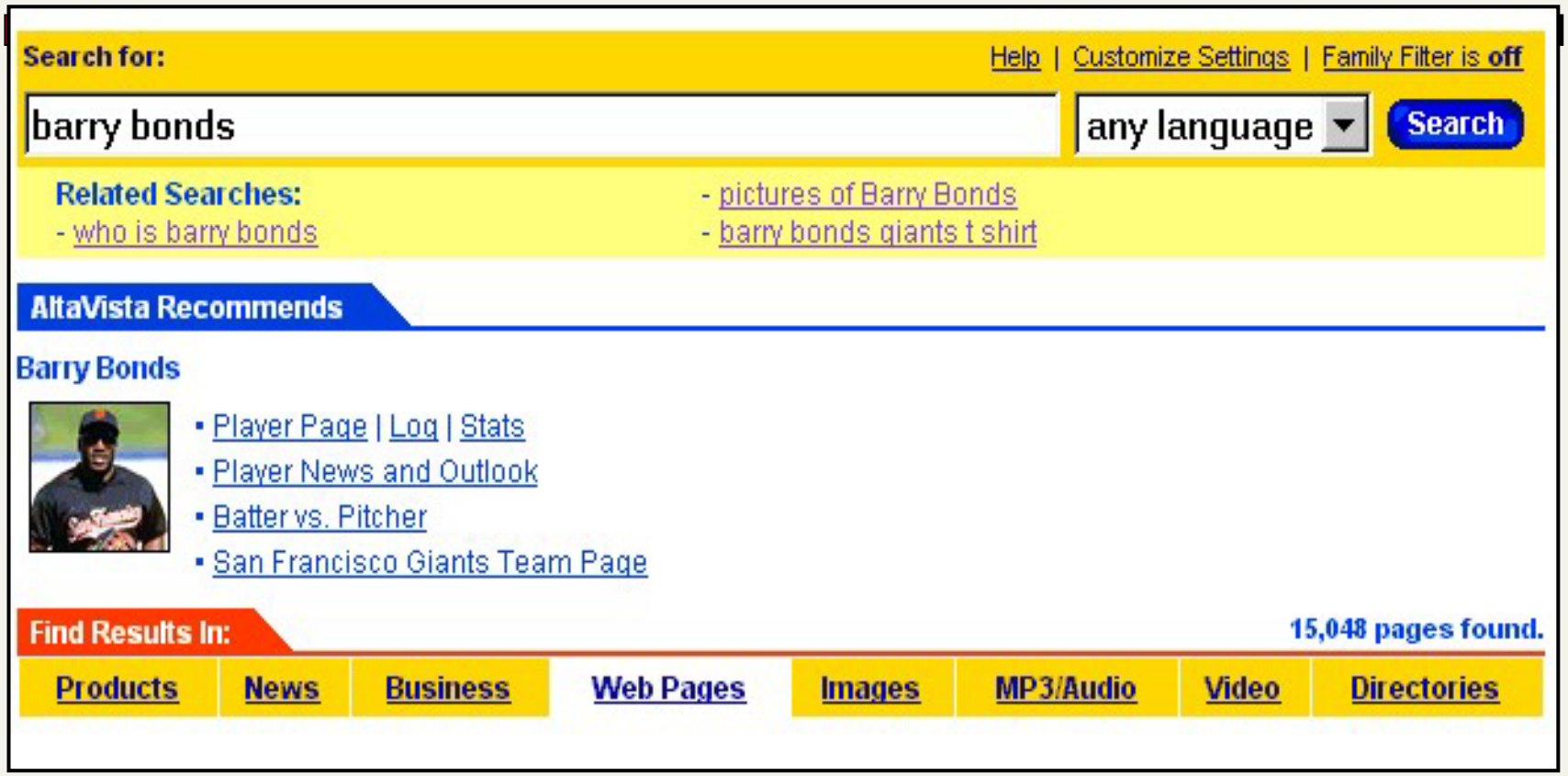
- Context determination
  - spatial (user location/target location)
  - query stream (previous queries)
  - personal (user profile)
  - explicit (vertical search, family friendly)
  - implicit (use AltaVista from AltaVista France)

- Context use
  - Result restriction
  - Ranking modulation

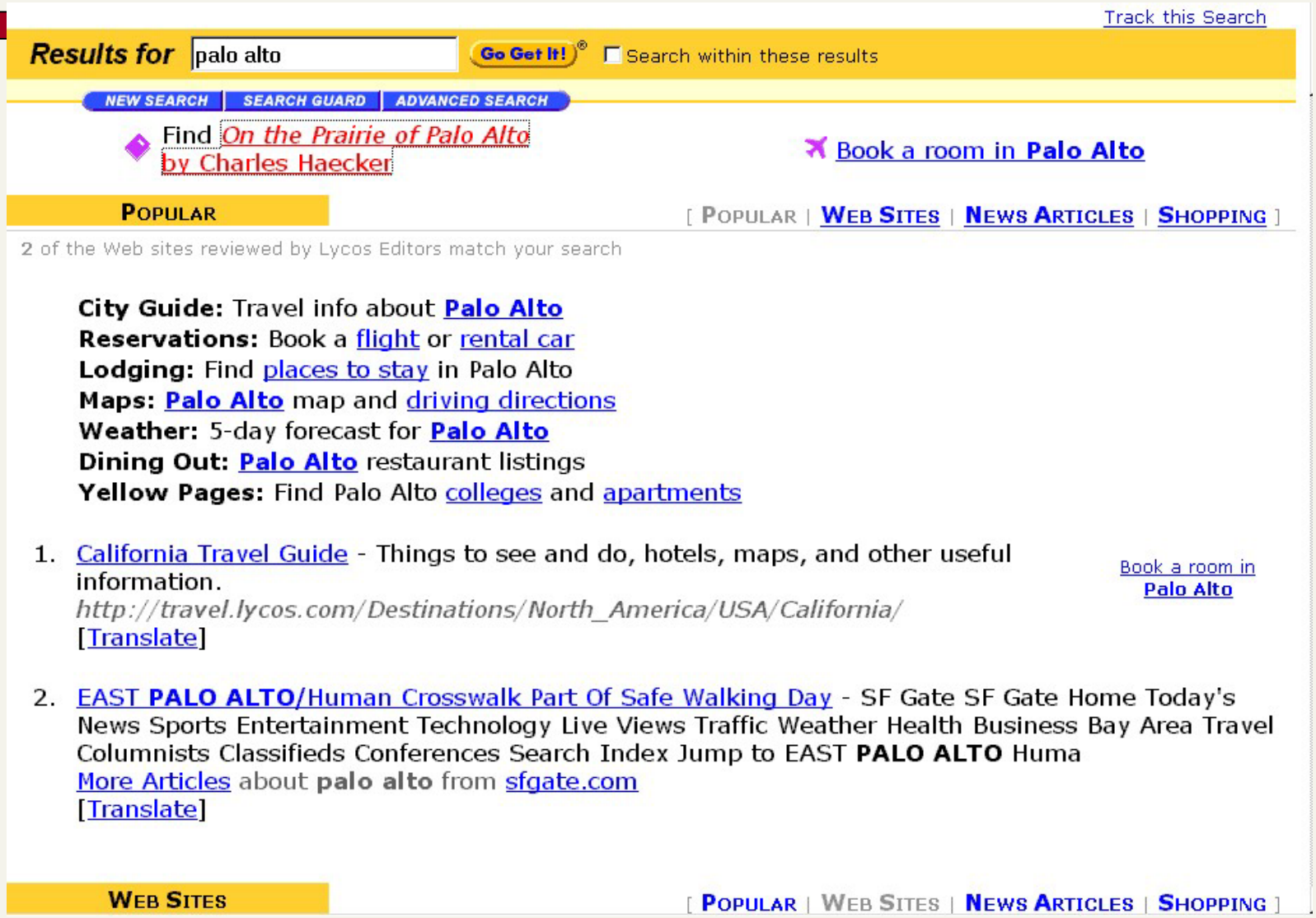
The spatial context - geo-search

- Two aspects
  - Geo-coding
    - encode geographic coordinates to make search effective
  - Geo-parsing
    - the process of identifying geographic context.
- Geo-coding
  - Geometrical hierarchy (squares)
  - Natural hierarchy (country, state, county, city, zip-codes, etc)
- Geo-parsing
  - Pages (infer from phone nos, zip, etc). About 10% feasible.
  - Queries (use dictionary of place names)
  - Users
    - From IP data

AV barry bonds



Lycos **palo alto**



Track this Search

Results for palo alto Go Get It! Search within these results

NEW SEARCH | SEARCH GUARD | ADVANCED SEARCH

Find *On the Prairie of Palo Alto* by Charles Haecker

Book a room in **Palo Alto**

POPULAR

[POPULAR | WEB SITES | NEWS ARTICLES | SHOPPING]

2 of the Web sites reviewed by Lycos Editors match your search

City Guide: Travel info about **Palo Alto**
Reservations: Book a flight or rental car
Lodging: Find places to stay in Palo Alto
Maps: **Palo Alto** map and driving directions
Weather: 5-day forecast for **Palo Alto**
Dining Out: **Palo Alto** restaurant listings
Yellow Pages: Find Palo Alto colleges and apartments

1. California Travel Guide - Things to see and do, hotels, maps, and other useful information.
   http://travel.lycos.com/Destinations/North_America/USA/California/
   [Translate]

Book a room in **Palo Alto**

2. EAST **PALO ALTO**/Human Crosswalk Part Of Safe Walking Day - SF Gate SF Gate Home Today's News Sports Entertainment Technology Live Views Traffic Weather Health Business Bay Area Travel Columnists Classifieds Conferences Search Index Jump to EAST **PALO ALTO** Huma
   More Articles about **palo alto** from sfgate.com
   [Translate]

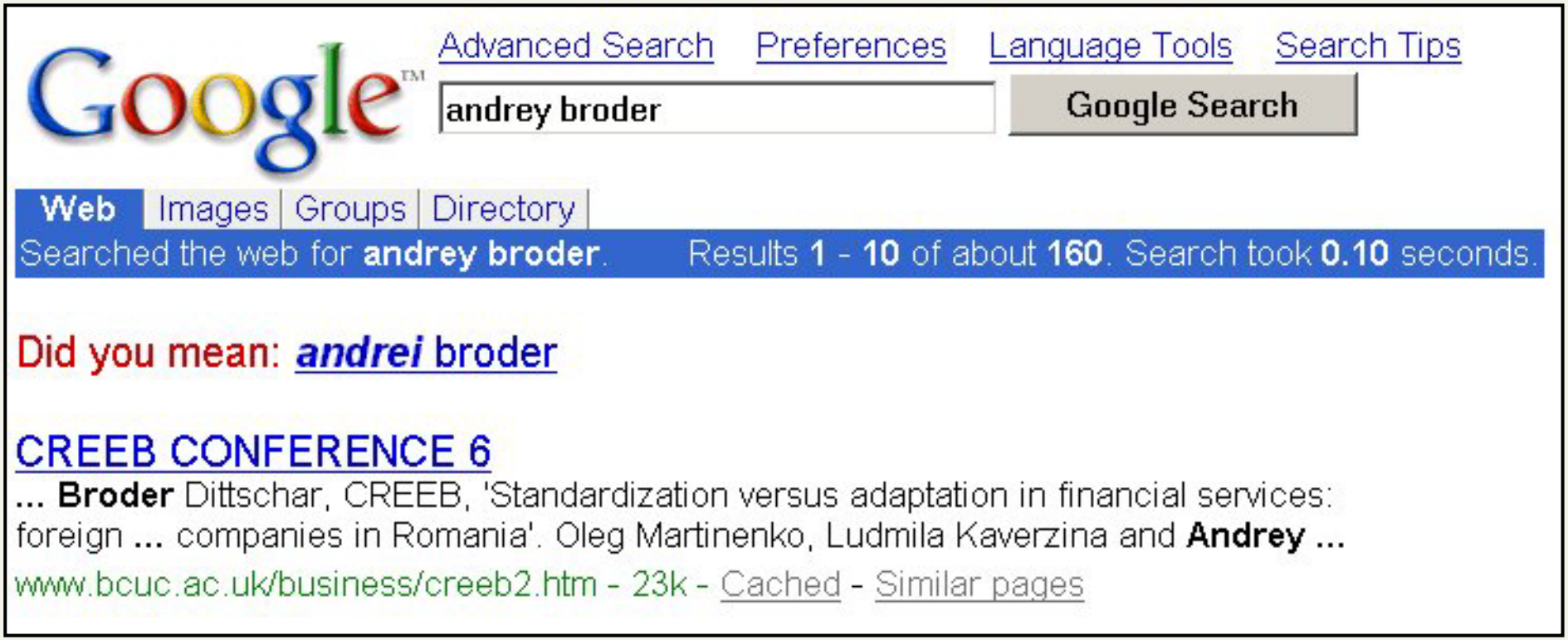
WEB SITES

[POPULAR | WEB SITES | NEWS ARTICLES | SHOPPING]

Helping the user

- UI
- spell checking
- query refinement
- query suggestion
- context transfer ...

Context sensitive spell check

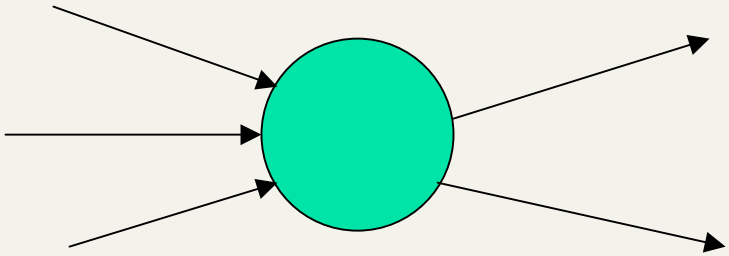


Citation Analysis

- Citation frequency
- Co-citation coupling frequency
  - Cocitations with a given author measures "impact"
  - Cocitation analysis [Mcca90]
- Bibliographic coupling frequency
  - Articles that co-cite the same articles are related
- Citation indexing
  - Who is a given author cited by? (Garfield [Garf72])
- Pinski and Narin
  - Precursor of Google's PageRank

Query-independent ordering

- First generation: using link counts as simple measures of popularity.
- Two basic suggestions:
  - Undirected popularity:
    - Each page gets a score = the number of in-links plus the number of out-links (3+2=5).
  - Directed popularity:
    - Score of a page = number of its in-links (3).



Query processing

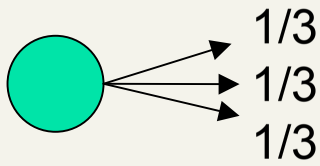
- First retrieve all pages meeting the text query (say *venture capital*).
- Order these by their link popularity (either variant on the previous page).

Spamming simple popularity

- *Exercise*: How do you spam each of the following heuristics so your page gets a high score?
- Each page gets a score = the number of in-links plus the number of out-links.
- Score of a page = number of its in-links.

Pagerank scoring

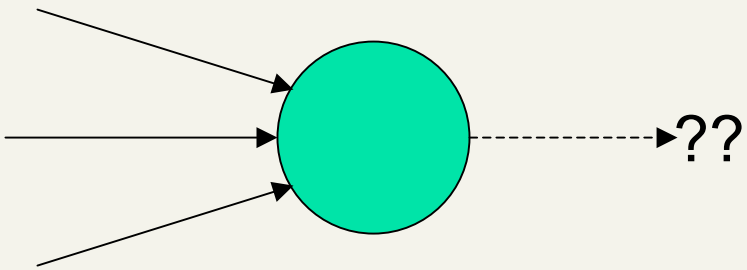
- Imagine a browser doing a random walk on web pages:
  - Start at a random page
  - At each step, go out of the current page along one of the links on that page, equiprobably



- "In the steady state" each page has a long-term visit rate - use this as the page's score.

Not quite enough

- The web is full of dead-ends.
  - Random walk can get stuck in dead-ends.
  - Makes no sense to talk about long-term visit rates.



Teleporting

- At each step, with probability 10%, jump to a random web page.
- With remaining probability (90%), go out on a random link.
  - If no out-link, stay put in this case.

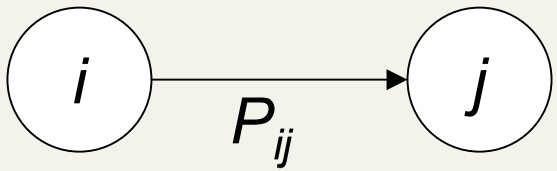
Result of teleporting

- Now cannot get stuck locally.
- There is a long-term rate at which any page is visited (not obvious, will show this).
- How do we compute this visit rate?

Markov chains

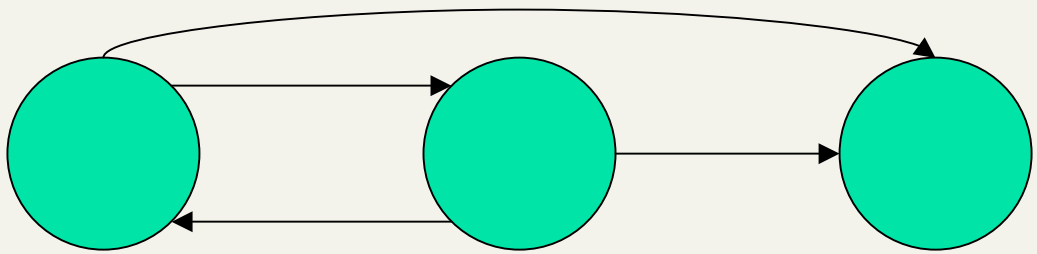
- A Markov chain consists of n states, plus an $n \times n$ transition probability matrix P.
- At each step, we are in exactly one of the states.
- For $1 \leq i,j \leq n$, the matrix entry P_{ij} tells us the probability of j being the next state, given we are currently in state i.

$P_{ii}>0$ is OK.



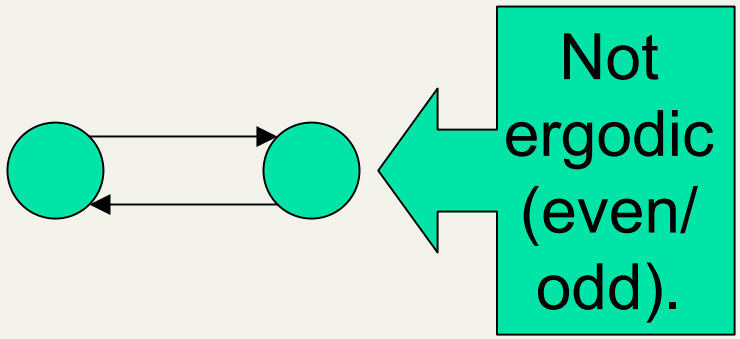
Markov chains

- Clearly, for all i, $\sum_{j=1}^{n} P_{ij} = 1.$
- Markov chains are abstractions of random walks.
- *Exercise*: represent the teleporting random walk from 3 slides ago as a Markov chain, for this case:



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.



Ergodic Markov chains

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

Probability vectors

- A probability (row) vector $\mathbf{x} = (x_1, \ldots x_n)$ 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 $\mathbf{x} = (x_1, \ldots x_n)$ means the walk is in state *i* with probability x_i.

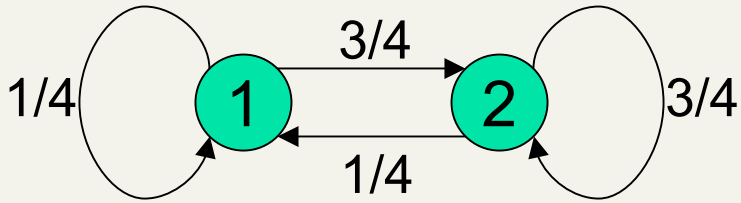
$$\sum_{i=1}^{n} x_i = 1.$$

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 P tells us where we go next from state *i*.
- So from x, our next state is distributed as xP.

Computing the visit rate

- The steady state looks like a vector of probabilities **a** $= (a_1, \ldots a_n)$:
  - a_i is the probability that we are in state i.



For this example, $a_1 = 1/4$ and $a_2 = 3/4$.

How do we compute this vector?

- Let $\mathbf{a} = (a_1, \ldots a_n)$ denote the row vector of steady-state probabilities.
- If we our current position is described by **a**, then the next step is distributed as **a**P.
- But **a** is the steady state, so **a**=**a**P.
- Solving this matrix equation gives us **a**.
  - So **a** is the (left) eigenvector for P.
  - (Corresponds to the “principal” eigenvector of P with the largest eigenvalue.)

One way of computing **a**

- Recall, regardless of where we start, we eventually reach the steady state **a**.
- Start with any distribution (say $\mathbf{x}=(10\ldots0)$).
- After one step, we're at $\mathbf{x}\mathbf{P}$;
- after two steps at $\mathbf{x}\mathbf{P}^2$, then $\mathbf{x}\mathbf{P}^3$ and so on.
- "Eventually" means for "large" k, $\mathbf{x}\mathbf{P}^k = \mathbf{a}$.
- Algorithm: multiply **x** by increasing powers of P until the product looks stable.

Pagerank summary

- Preprocessing:
  - Given graph of links, build matrix **P**.
  - From it compute **a**.
  - The entry a_i is a number between 0 and 1: the pagerank of page *i*.
- Query processing:
  - Retrieve pages meeting query.
  - Rank them by their pagerank.
  - Order is query-*independent*.

The reality

- Pagerank is used in google, but so are many other clever heuristics
  - more on these heuristics later.

Special notes

- Bib entries for this (and following) web search lectures
  http://www.stanford.edu/class/archive/cs/cs276a/cs276a.1032/handouts/tutbib_v4.html