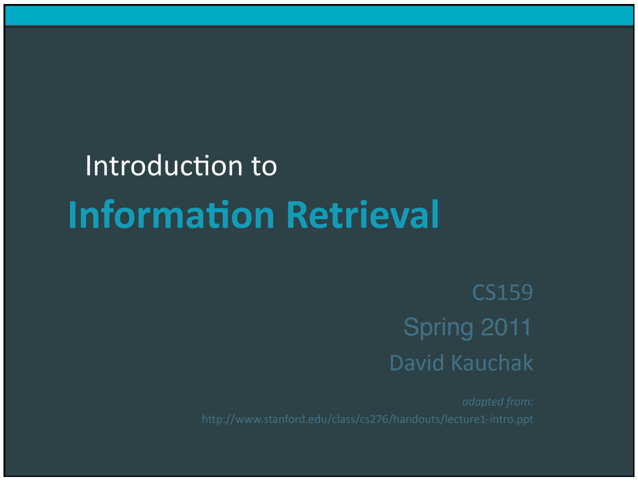


Introduction to

Information Retrieval

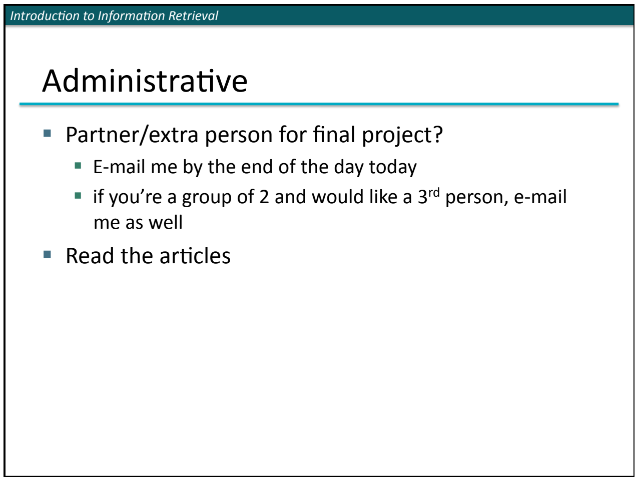
CS159

Spring 2011

David Kauchak

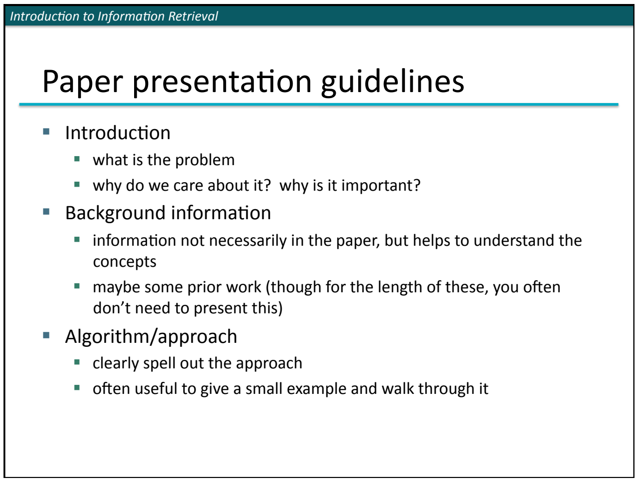
adapted from:

http://www.stanford.edu/class/cs276/handouts/lecture1-intro.ppt



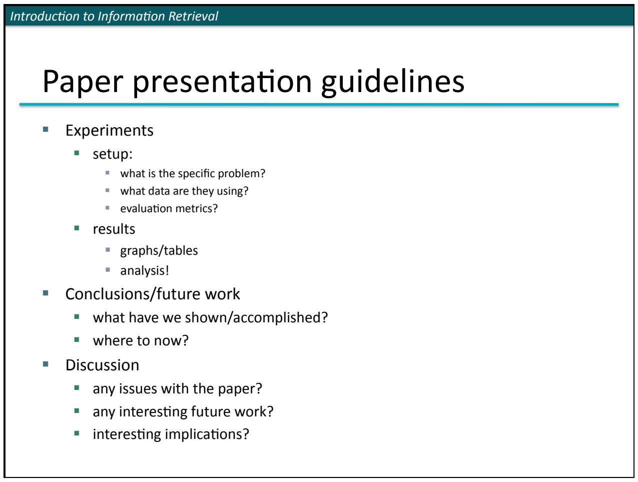
Administrative

- Partner/extra person for final project?
  - E-mail me by the end of the day today
  - if you're a group of 2 and would like a 3rd person, e-mail me as well
- Read the articles



Paper presentation guidelines

- Introduction
  - what is the problem
  - why do we care about it? why is it important?
- Background information
  - information not necessarily in the paper, but helps to understand the concepts
  - maybe some prior work (though for the length of these, you often don't need to present this)
- Algorithm/approach
  - clearly spell out the approach
  - often useful to give a small example and walk through it



Paper presentation guidelines

- Experiments
  - setup:
    - what is the specific problem?
    - what data are they using?
    - evaluation metrics?
  - results
    - graphs/tables
    - analysis!
- Conclusions/future work
  - what have we shown/accomplished?
  - where to now?
- Discussion
  - any issues with the paper?
  - any interesting future work?
  - interesting implications?

Paper presentation guidelines

- Misc
  - Presenting the material
    - be energetic/enthusiastic
    - make sure you know the material!
    - don't read directly from your slides (or note cards if you bring them)
    - use some visual presentation software (e.g. powerpoint)
    - audience interaction is good (though not necessary for this type of presentation)
  - Avoid lots of text (i.e. this is a bad slide ☺)
    - powerpoint has a notes feature that you can use to remind yourself what you want to say, but not show to the audience (you can also print it out and use this instead)
  - use lots of images/figures/diagrams
  - show examples to illustrate algorithms/points
  - go beyond the paper – papers and presentations have difference goals

Paper presentation guidelines

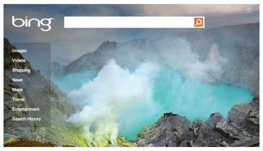
- more misc
  - presentation should add value to the paper
  - equations: make it clear what each part of the equation is
  - graphs: if you show a graph:
    - explain what the axes are
    - explain what we're looking at
    - explain why we care about this/what the result is
  - ~1 slide per minute (give or take with introductory material, animations, etc)
  - consider an outline during presentation to help the audience know where you're at

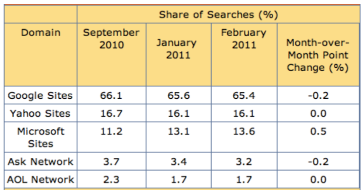
Information retrieval (IR)

- What comes to mind when I say "information retrieval"?

- Where have you seen IR? What are some real-world examples/uses?
  - Search engines
  - File search (e.g. OS X Spotlight, Windows Instant Search, Google Desktop)
  - Databases?
  - Catalog search (e.g. library)
  - Intranet search (i.e. corporate networks)

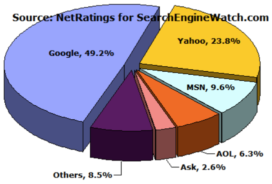
Web search



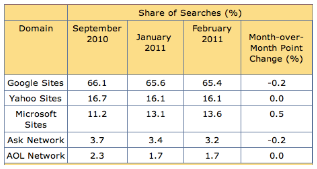


Domain	Share of Searches (%)			
	September 2010	January 2011	February 2011	Month-over-Month Point Change (%)
Google Sites	66.1	65.6	65.4	-0.2
Yahoo Sites	16.7	16.1	16.1	0.0
Microsoft Sites	11.2	13.1	13.6	0.5
Ask Network	3.7	3.4	3.2	-0.2
AOL Network	2.3	1.7	1.7	0.0

Web search



July 2006



Domain	Share of Searches (%)			
	September 2010	January 2011	February 2011	Month-over-Month Point Change (%)
Google Sites	66.1	65.6	65.4	-0.2
Yahoo Sites	16.7	16.1	16.1	0.0
Microsoft Sites	11.2	13.1	13.6	0.5
Ask Network	3.7	3.4	3.2	-0.2
AOL Network	2.3	1.7	1.7	0.0

Feb 2011

Challenges

- Why is information retrieval hard?
  - Lots and lots of data
    - efficiency
    - storage
    - discovery (web)
  - Data is unstructured
  - Querying/Understanding user intent
  - SPAM
  - Data quality

web search engine traffic

About 92,200,000 results (0.13 seconds)

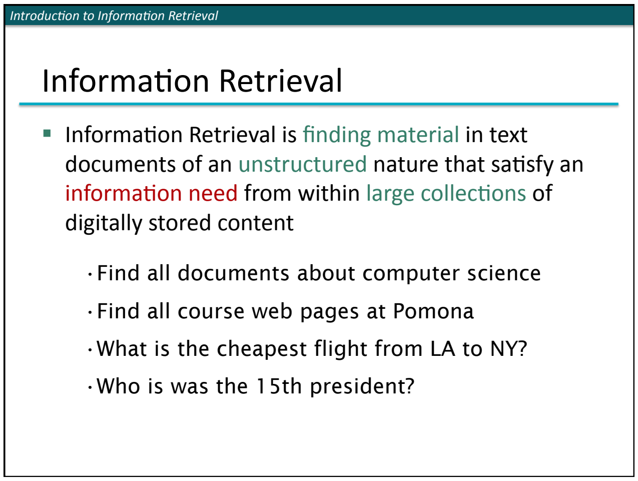
Information Retrieval

- Information Retrieval is finding material in documents of an unstructured nature that satisfy an information need from within large collections of digitally stored content

Information Retrieval

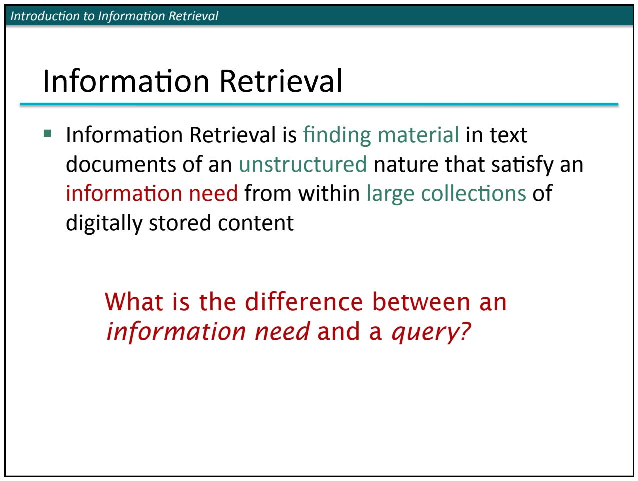
- Information Retrieval is finding material in documents of an unstructured nature that satisfy an information need from within large collections of digitally stored content

?



Information Retrieval

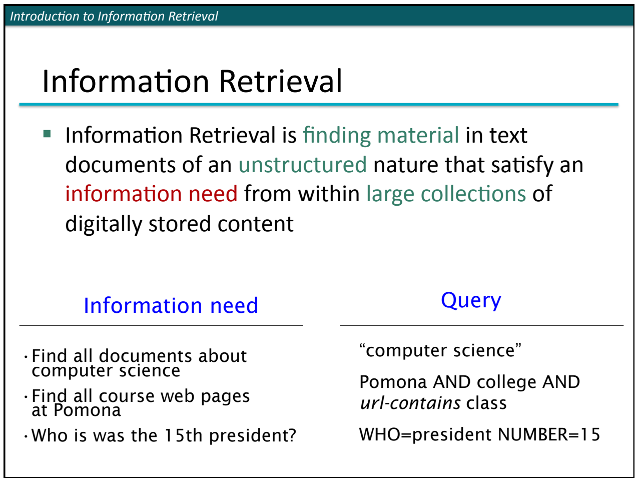
- Information Retrieval is finding material in text documents of an unstructured nature that satisfy an information need from within large collections of digitally stored content
  - Find all documents about computer science
  - Find all course web pages at Pomona
  - What is the cheapest flight from LA to NY?
  - Who is was the 15th president?



Information Retrieval

- Information Retrieval is finding material in text documents of an unstructured nature that satisfy an information need from within large collections of digitally stored content

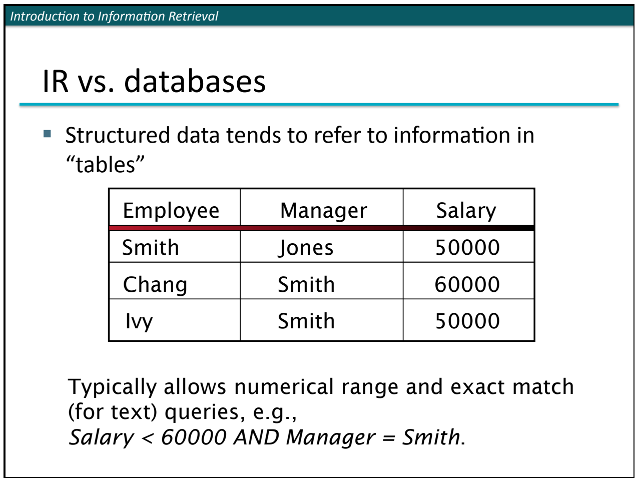
What is the difference between an *information need* and a *query?*



Information Retrieval

- Information Retrieval is finding material in text documents of an unstructured nature that satisfy an information need from within large collections of digitally stored content

Information need	Query
Find all documents about computer science	"computer science"
Find all course web pages at Pomona	Pomona AND college AND *url-contains* class
Who is was the 15th president?	WHO=president NUMBER=15



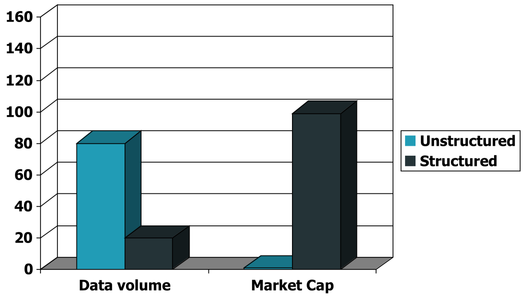
IR vs. databases

- Structured data tends to refer to information in "tables"

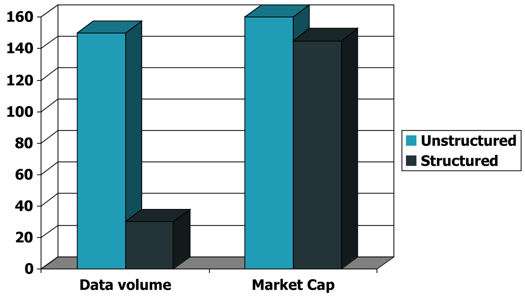
Employee	Manager	Salary
Smith	Jones	50000
Chang	Smith	60000
Ivy	Smith	50000

Typically allows numerical range and exact match (for text) queries, e.g.,
Salary < 60000 AND Manager = Smith.

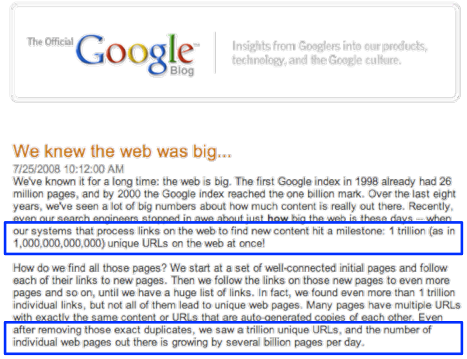
Unstructured (text) vs. structured (database) data in 1996



Unstructured (text) vs. structured (database) data in 2006



The web


The Official Google Blog
Insights from Googlers into our products, technology, and the Google culture.

We knew the web was big...

7/25/2008 10:12:00 AM

We've known it for a long time: the web is big. The first Google index in 1998 already had 26 million pages, and by 2000 the Google index reached the one billion mark. Over the last eight years, we've seen a lot of big numbers about how much content is really out there. Recently, even our search engineers stopped in awe about just **how** big the web is these days -- when our systems that process links on the web to find new content hit a milestone: 1 trillion (as in 1,000,000,000,000) unique URLs on the web at once!

How do we find all those pages? We start at a set of well-connected initial pages and follow each of their links to new pages. Then we follow the links on those new pages to even more pages and so on, until we have a huge list of links. In fact, we found even more than 1 trillion individual links, but not all of them lead to unique web pages. Many pages have multiple URLs with exactly the same content or URLs that are auto-generated copies of each other. Even after removing those exact duplicates, we saw a trillion unique URLs, and the number of individual web pages out there is growing by several billion pages per day.

Web is just the start…

e-mail

247 billion e-mails a day

twitter

27 million tweets a day

corporate databases

Blogs:**126 million** different blogs

http://royal.pingdom.com/2010/01/22/internet-2009-in-numbers/

Challenges

- Why is information retrieval hard?
  - Lots and lots of data
    - efficiency
    - storage
    - discovery (web)
  - Data is unstructured
  - Understanding user intent
  - SPAM
  - Data quality

Efficiency

- 27 million tweets over 4 years = ~40 billion tweets
- How much data is this?
  - ~4 TB of data uncompressed for the text itself
  - ~40 TB of data including additional meta-data
- 40 billion web pages?
  - assume web pages are 100 times longer than tweets
    - 400 TB of data
    - 100 4 TB disks
  - assume web pages are 1000 times long than tweets
    - 4 PB of data
    - 1000 4 TB disks
  - assume web pages are 10,000 times longer than tweets
    - 40 PB of data
    - 10,000 4TB disks

Efficiency

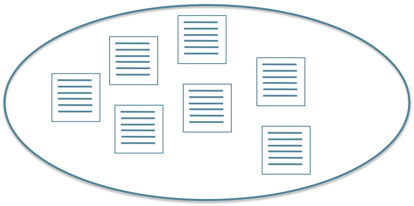
- Can we store all of the documents in memory?
- How long will it take to do a naïve search of the data?

- To search over a small data collection, almost any approach will work (e.g. grep)
- At web scale, there are many challenges:
  - queries need to be really fast!
  - massive parallelization
  - redundancy (hard-drives fail, networks fail, …)

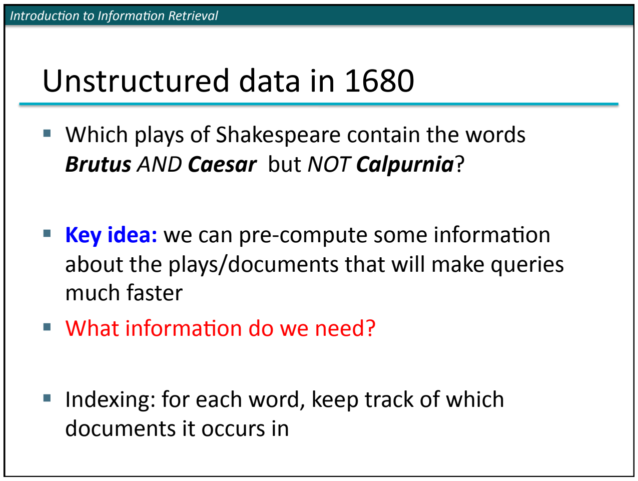
Unstructured data in 1680

- Which plays of Shakespeare contain the words ***Brutus*** *AND* ***Caesar*** but *NOT* ***Calpurnia***?

All of Shakespeare's plays

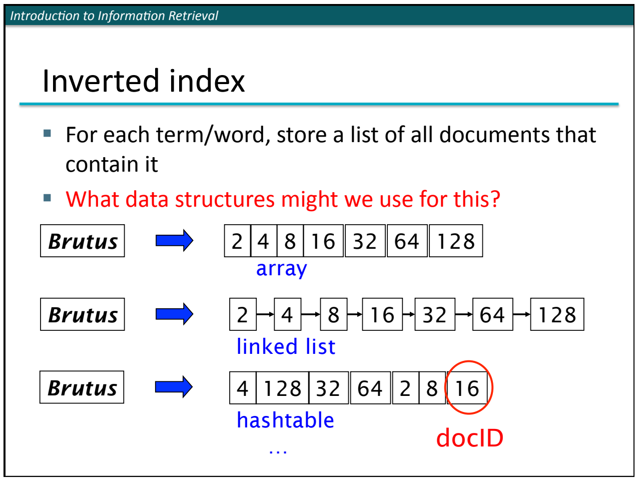


How can we answer this query quickly?



Unstructured data in 1680

- Which plays of Shakespeare contain the words ***Brutus*** *AND* ***Caesar*** but *NOT* ***Calpurnia***?
- **Key idea:** we can pre-compute some information about the plays/documents that will make queries much faster
- What information do we need?
- Indexing: for each word, keep track of which documents it occurs in



Inverted index

- For each term/word, store a list of all documents that contain it
- What data structures might we use for this?

Brutus → 2 | 4 | 8 | 16 | 32 | 64 | 128

array

Brutus → 2 → 4 → 8 → 16 → 32 → 64 → 128

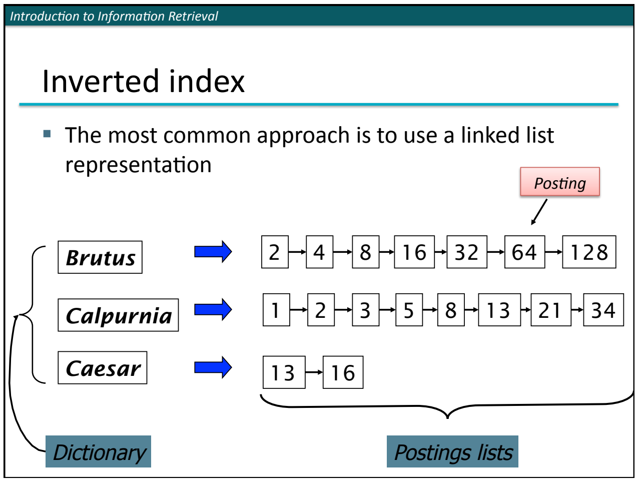
linked list

Brutus → 4 | 128 | 32 | 64 | 2 | 8 | 16

hashtable

…

docID



Inverted index

- The most common approach is to use a linked list representation

Posting

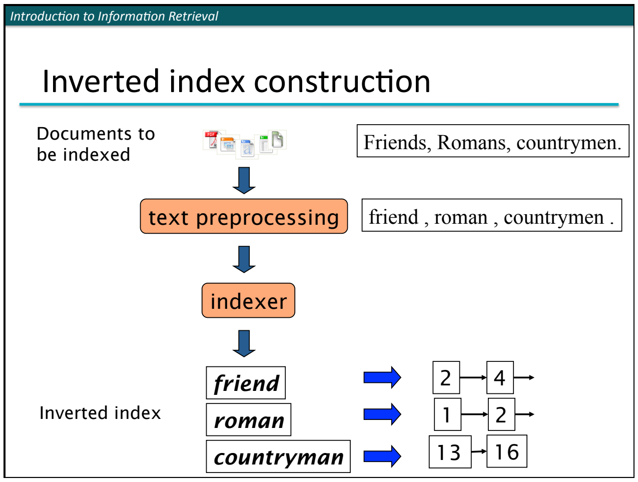
Brutus → 2 → 4 → 8 → 16 → 32 → 64 → 128

Calpurnia → 1 → 2 → 3 → 5 → 8 → 13 → 21 → 34

Caesar → 13 → 16

Dictionary

Postings lists



Inverted index construction

Documents to be indexed

Friends, Romans, countrymen.

text preprocessing

friend , roman , countrymen .

indexer

Inverted index

friend → 2 → 4 →

roman → 1 → 2 →

countryman → 13 → 16

Boolean retrieval

- Support queries that are boolean expressions:
  - A boolean query uses *AND, OR* and *NOT* to join query terms
    - Caesar *AND* Brutus *AND NOT* Calpurnia
    - Pomona *AND* College
    - (Mike *OR* Michael) *AND* Jordan *AND NOT*(Nike *OR* Gatorade)
- Given only these operations, what types of questions can't we answer?
  - Phrases, e.g. "Pomona College"
  - Proximity, "Michael" within 2 words of "Jordan"
  - Regular expression-like

Boolean retrieval

- Primary commercial retrieval tool for 3 decades
- Professional searchers (e.g., lawyers) still like boolean queries
- Why?
  - You know exactly what you're getting, a query either matches or it doesn't
  - Through trial and error, can frequently fine tune the query appropriately
  - Don't have to worry about underlying heuristics (e.g. PageRank, term weightings, synonym, etc…)

Example: WestLaw http://www.westlaw.com/

- Largest commercial (paying subscribers) legal search service (started 1975; ranking added 1992)
- Tens of terabytes of data; 700,000 users
- Majority of users *still* use boolean queries
- Example query:
  - What is the statute of limitations in cases involving the federal tort claims act?
  - LIMIT! /3 STATUTE ACTION /S FEDERAL /2 TORT /3 CLAIM
  - All words starting with "LIMIT"

Example: WestLaw http://www.westlaw.com/

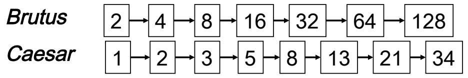
- Largest commercial (paying subscribers) legal search service (started 1975; ranking added 1992)
- Tens of terabytes of data; 700,000 users
- Majority of users *still* use boolean queries
- Example query:
  - What is the statute of limitations in cases involving the federal tort claims act?
  - LIMIT! /3 STATUTE ACTION /S FEDERAL /2 TORT /3 CLAIM

Example: WestLaw http://www.westlaw.com/

- Largest commercial (paying subscribers) legal search service (started 1975; ranking added 1992)
- Tens of terabytes of data; 700,000 users
- Majority of users *still* use boolean queries
- Example query:
  - What is the statute of limitations in cases involving the federal tort claims act?
  - LIMIT! /3 STATUTE ACTION /S FEDERAL /2 TORT /3 CLAIM
  - /3 = within 3 words, /S = in same sentence

Query processing: AND

- What needs to happen to process: ***Brutus** AND **Caesar***
- Locate ***Brutus*** and ***Caesar*** in the Dictionary;
  - Retrieve postings lists

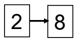


Brutus 2 → 4 → 8 → 16 → 32 → 64 → 128

Caesar 1 → 2 → 3 → 5 → 8 → 13 → 21 → 34

- "Merge" the two postings:

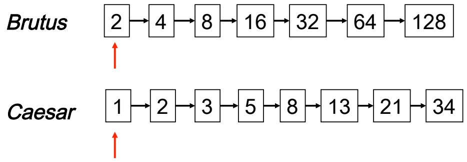
Brutus AND Caesar



2 → 8

The merge

- Walk through the two postings simultaneously



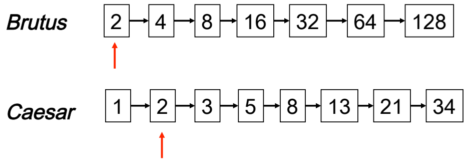
Brutus 2 → 4 → 8 → 16 → 32 → 64 → 128

Caesar 1 → 2 → 3 → 5 → 8 → 13 → 21 → 34

Brutus AND Caesar

The merge

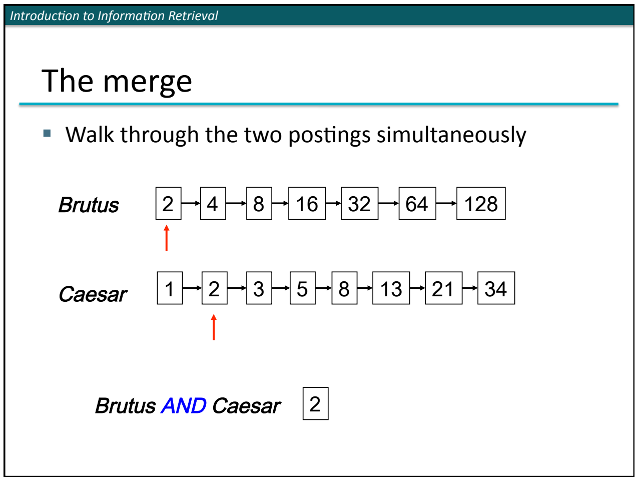
- Walk through the two postings simultaneously



Brutus 2 → 4 → 8 → 16 → 32 → 64 → 128

Caesar 1 → 2 → 3 → 5 → 8 → 13 → 21 → 34

Brutus AND Caesar



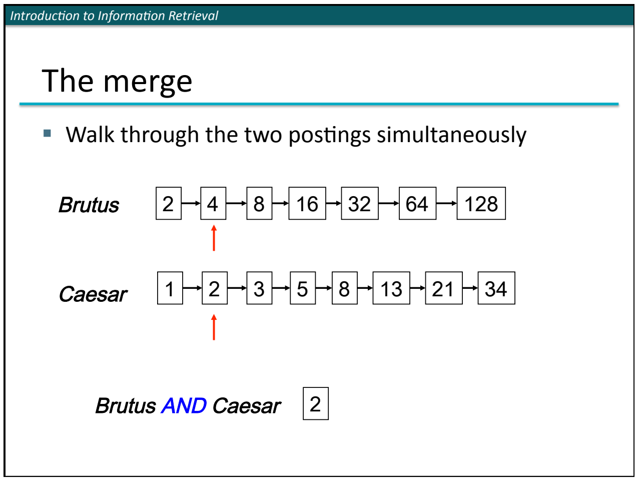
The merge

- Walk through the two postings simultaneously

Brutus 2 → 4 → 8 → 16 → 32 → 64 → 128

Caesar 1 → 2 → 3 → 5 → 8 → 13 → 21 → 34

Brutus AND Caesar 2



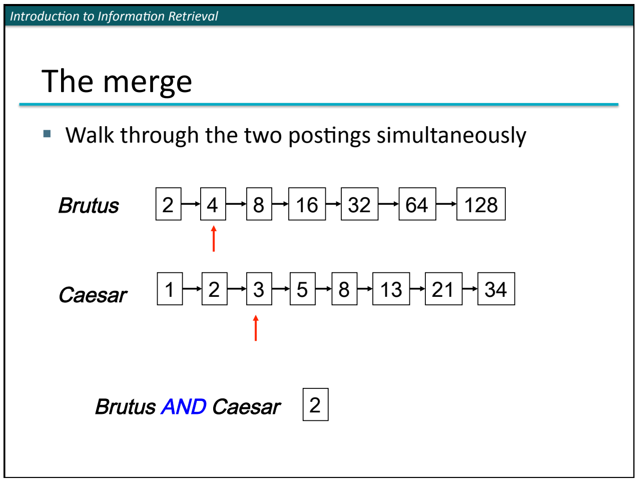
The merge

- Walk through the two postings simultaneously

Brutus 2 → 4 → 8 → 16 → 32 → 64 → 128

Caesar 1 → 2 → 3 → 5 → 8 → 13 → 21 → 34

Brutus AND Caesar 2



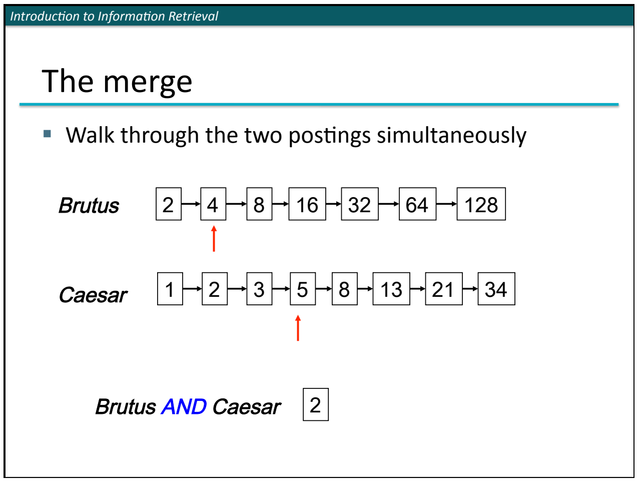
The merge

- Walk through the two postings simultaneously

Brutus 2 → 4 → 8 → 16 → 32 → 64 → 128

Caesar 1 → 2 → 3 → 5 → 8 → 13 → 21 → 34

Brutus AND Caesar 2



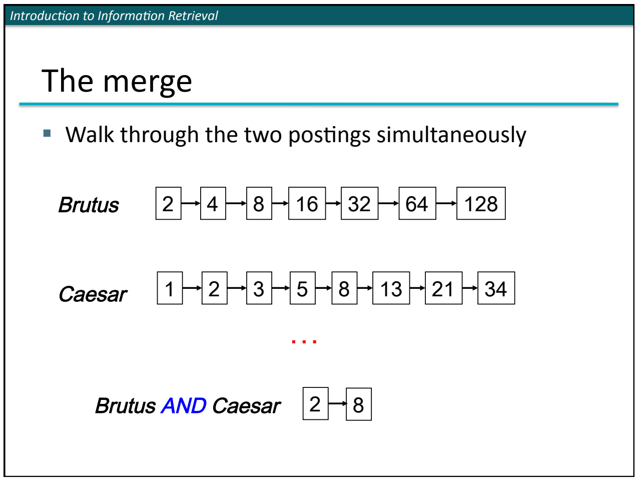
The merge

- Walk through the two postings simultaneously

Brutus 2 → 4 → 8 → 16 → 32 → 64 → 128

Caesar 1 → 2 → 3 → 5 → 8 → 13 → 21 → 34

Brutus AND Caesar 2



Introduction to Information Retrieval

The merge

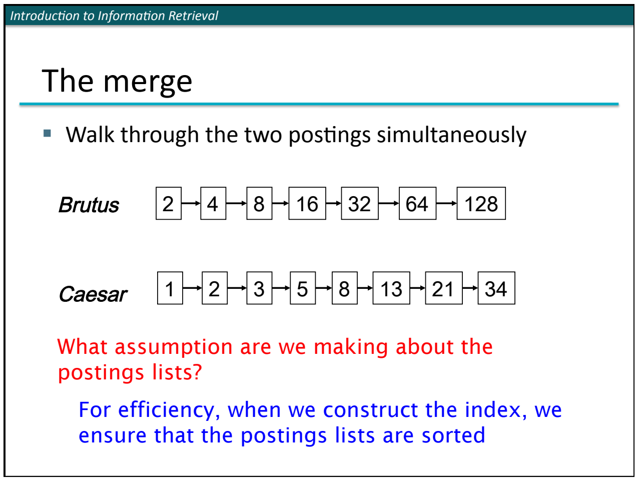
- Walk through the two postings simultaneously

Brutus 2 → 4 → 8 → 16 → 32 → 64 → 128

Caesar 1 → 2 → 3 → 5 → 8 → 13 → 21 → 34

. . .

Brutus AND Caesar 2 → 8



Introduction to Information Retrieval

The merge

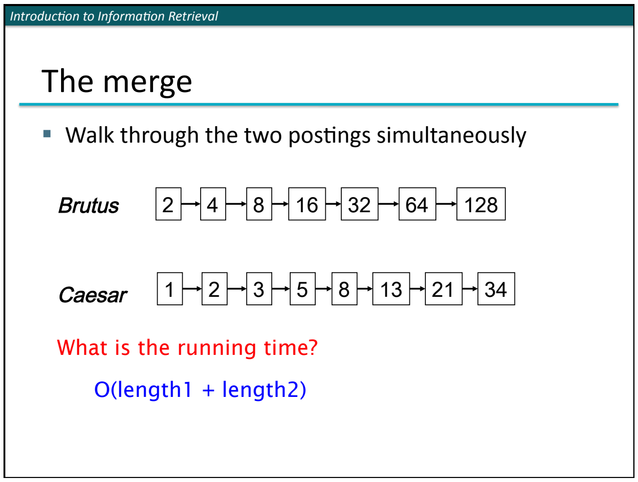
- Walk through the two postings simultaneously

Brutus 2 → 4 → 8 → 16 → 32 → 64 → 128

Caesar 1 → 2 → 3 → 5 → 8 → 13 → 21 → 34

What assumption are we making about the postings lists?

For efficiency, when we construct the index, we ensure that the postings lists are sorted



Introduction to Information Retrieval

The merge

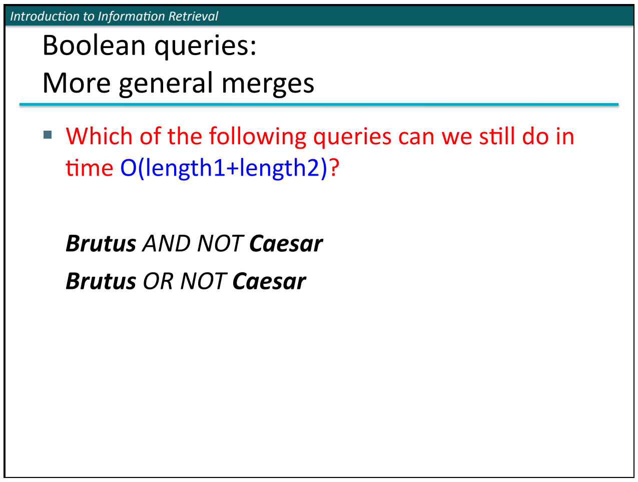
- Walk through the two postings simultaneously

Brutus 2 → 4 → 8 → 16 → 32 → 64 → 128

Caesar 1 → 2 → 3 → 5 → 8 → 13 → 21 → 34

What is the running time?

O(length1 + length2)



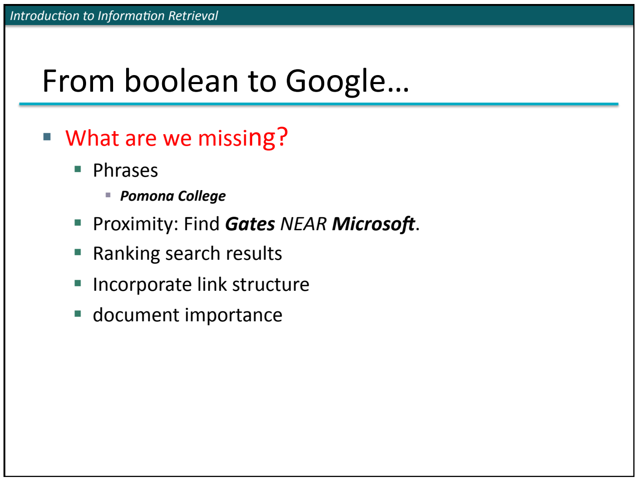
Introduction to Information Retrieval

Boolean queries: More general merges

- Which of the following queries can we still do in time O(length1+length2)?

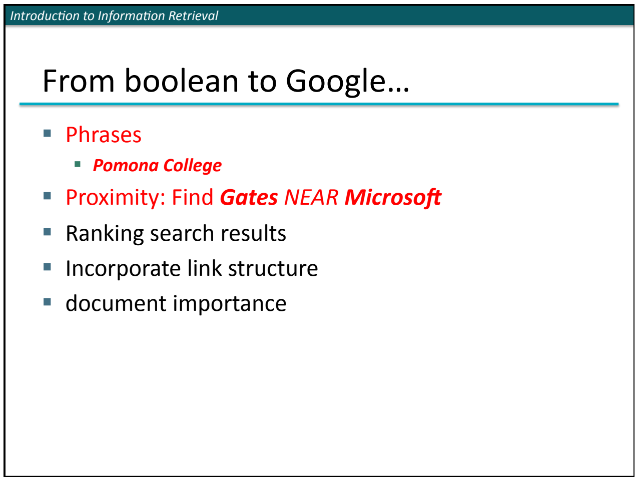
Brutus *AND NOT* ***Caesar***

Brutus *OR NOT* ***Caesar***



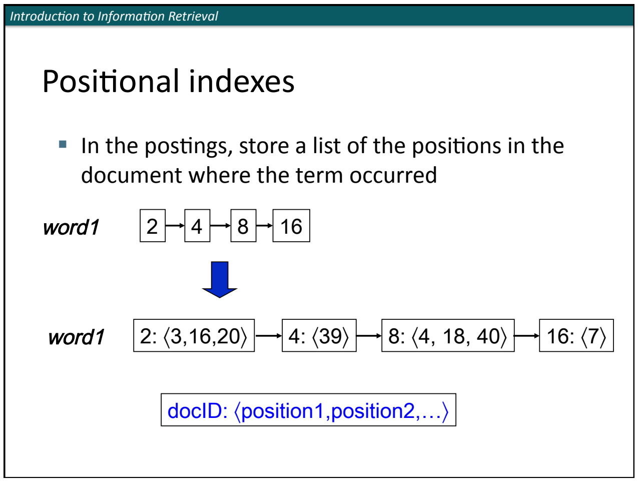
From boolean to Google…

- What are we missing?
  - Phrases
    - ***Pomona College***
  - Proximity: Find ***Gates*** *NEAR* ***Microsoft***.
  - Ranking search results
  - Incorporate link structure
  - document importance



From boolean to Google…

- Phrases
  - ***Pomona College***
- Proximity: Find ***Gates*** *NEAR* ***Microsoft***
- Ranking search results
- Incorporate link structure
- document importance



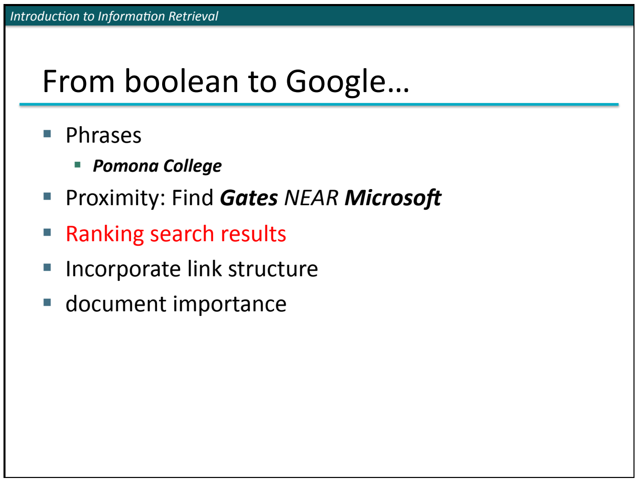
Positional indexes

- In the postings, store a list of the positions in the document where the term occurred

word1 2 → 4 → 8 → 16

word1 2: ⟨3,16,20⟩ → 4: ⟨39⟩ → 8: ⟨4, 18, 40⟩ → 16: ⟨7⟩

docID: ⟨position1,position2,…⟩



From boolean to Google…

- Phrases
  - ***Pomona College***
- Proximity: Find ***Gates*** *NEAR* ***Microsoft***
- Ranking search results
- Incorporate link structure
- document importance

Rank documents by text similarity

- Ranked information retrieval!
- Simple version: Vector space ranking (e.g. TF-IDF)
  - include occurrence frequency
  - weighting (e.g. IDF)
  - rank results by similarity between query and document
- Realistic version: many more things in the pot…
  - treat different occurrences differently (e.g. title, header, link text, …)
  - many other weightings
  - document importance
  - spam
  - hand-crafted/policy rules

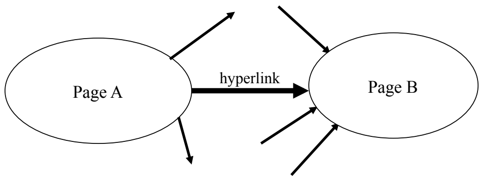
IR with TF-IDF

- How can we change our inverted index to make ranked queries (e.g. TF-IDF) fast?
- Store the TF initially in the index
- In addition, store the number of documents the term occurs in in the index
- IDFs
  - We can either compute these on the fly using the number of documents in each term
  - We can make another pass through the index and update the weights for each entry

From boolean to Google…

- Phrases
  - ***Pomona College***
- Proximity: Find ***Gates*** *NEAR* ***Microsoft***
- Ranking search results
  - include occurrence frequency
  - weighting
  - treat different occurrences differently (e.g. title, header, link text, …)
- Incorporate link structure
- document importance

The Web as a Directed Graph

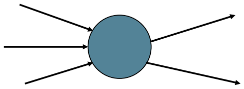


A hyperlink between pages denotes author perceived relevance AND importance

How can we use this information?

Query-independent ordering

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



problems?

What is pagerank?

- The random surfer model
- Imagine a user surfing the web randomly using a web browser
- The pagerank score of a page is the probability that that user will visit a given page

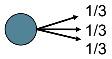
http://images.clipartof.com/small/7872-Clipart-Picture-Of-A-World-Earth-Globe-Mascot-Cartoon-Character-Surfing-On-A-Blue-And-Yellow-Surfboard.jpg

Random surfer model

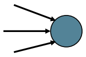
- We want to model the behavior of a "random" user interfacing the web through a browser
- Model is independent of content (i.e. just graph structure)
- What types of behavior should we model and how?
  - Where to start
  - Following links on a page
  - Typing in a url (bookmarks)
  - What happens if we get a page with no outlinks
  - Back button on browser

Random surfer model

- Start at a random page
- Go out of the current page along one of the links on that page, equiprobably

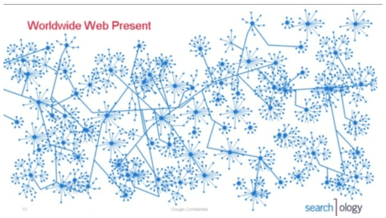


- "Teleporting"
  - If a page has no outlinks always jump to random page
  - With some fixed probability, randomly jump to any other page, otherwise follow links



The questions…

- Given a graph and a teleporting probability, we have some probability of visiting every page
- What is that probability for each page in the graph?



http://3.bp.blogspot.com/_ZaGO7GjCqAI/Rkyo5uCmBdI/AAAAAAAACLo/zsHdSlKc-q4/s640/searchology-web-graph.png

Pagerank summary

- Preprocessing:
  - Given a graph of links, build matrix **P**
  - From it compute **steady state** of each state
  - An entry is a number between 0 and 1: the pagerank of a page
- Query processing:
  - Retrieve pages meeting query
  - Integrate pagerank score with other scoring (e.g. tf-idf)
  - Rank pages by this combined score

Pagerank problems?

- Can still fool pagerank
  - link farms
    - Create a bunch of pages that are tightly linked and on topic, then link a few pages to off-topic pages
  - link exchanges
    - I'll pay you to link to me
    - I'll link to you if you'll link to me
  - buy old URLs
  - post on blogs, etc. with URLs
  - Create crappy content (but still may seem relevant)

IR Evaluation

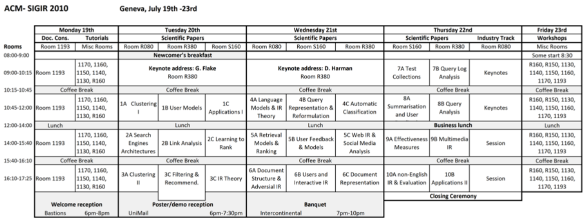
- Like any research area, an important component is how to evaluate a system
- What are important features for an IR system?
- How might we automatically evaluate the performance of a system? Compare two systems?
- What data might be useful?

Measures for a search engine

- How fast does it index (how frequently can we update the index)
- How fast does it search
- How big is the index
- Expressiveness of query language
- UI
- Is it free?

- Quality of the search results

IR Research



ACM- SIGIR 2010 **Geneva, July 19th -23rd**

	Monday 19th		Tuesday 20th			Wednesday 21st			Thursday 22nd			Friday 23rd
	Doc. Cons.	Tutorials	Scientific Papers			Scientific Papers			Scientific Papers		Industry Track	Workshops
Rooms	Room 1193	Misc Rooms	Room R080	Room R380	Room S160	Room R080	Room S160	Room R380	Room S160	Room R380	Room R080	Misc Rooms
08:00-9:00			Newcomer's breakfast									Some start 8:30
09:00-10:15	Room 1193	1170, 1160, 1150, 1140, 1130, R160	Keynote address: G. Flake Room R380			Keynote address: D. Harman Room R380			7A Test Collections	7B Query Log Analysis	Keynotes	R160, R150, 1130, 1140, 1150, 1160, 1170, 1193
10:15-10:45	Coffee Break		Coffee Break			Coffee Break			Coffee Break			Coffee Break
10:45-12:00	Room 1193	1170, 1160, 1150, 1140, 1130, R160	1A Clustering I	1B User Models	1C Applications I	4A Language Models & IR Theory	4B Query Representation & Reformulation	4C Automatic Classification	8A Summarisation and User	8B Query Analysis	Keynotes	R160, R150, 1130, 1140, 1150, 1160, 1170, 1193
12:00-14:00	Lunch		Lunch			Lunch			Business lunch			Lunch
14:00-15:40	Room 1193	1170, 1160, 1150, 1140, 1130, R160	2A Search Engines Architectures	2B Link Analysis	2C Learning to Rank	5A Retrieval Models & Ranking	5B User Feedback & Models	5C Web IR & Social Media Analysis	9A Effectiveness Measures	9B Multimedia IR	Session	R160, R150, 1130, 1140, 1150, 1160, 1170, 1193
15:40-16:10	Coffee Break		Coffee Break			Coffee Break			Coffee Break			Coffee Break
16:10-17:25	Room 1193	1170, 1160, 1150, 1140, 1130, R160	3A Clustering II	3C Filtering & Recommend.	3C IR Theory	6A Document Structure & Adversial IR	6B Users and Interactive IR	6C Document Representation	10A non-English IR & Evaluation	10B Applications II	Session	R160, R150, 1130, 1140, 1150, 1160, 1170, 1193
	Welcome reception Bastions 6pm-8pm		Poster/demo reception UniMail 6pm-7:30pm			Banquet Intercontinental 7pm-10pm			Closing Ceremony			

$$$$

- How do search engines make money?