ΕΠΛ660

Ανάκτηση Πληροφοριών και Μηχανές Αναζήτησης

- Introduction and Boolean Retrieval

Διαδικαστικά

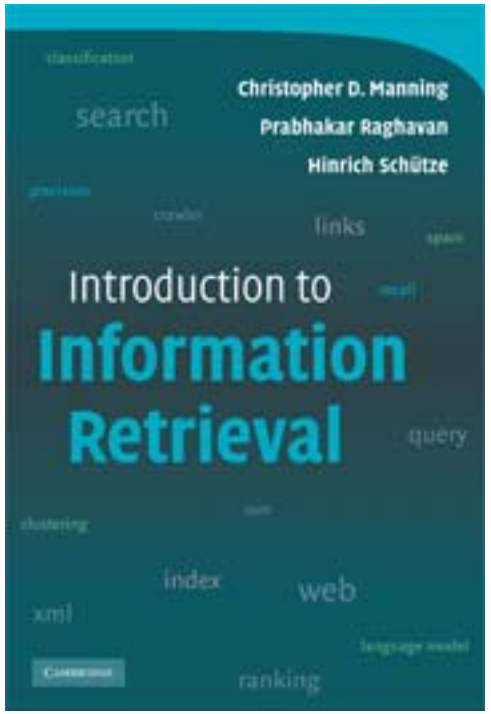
- Μεταπτυχιακό μάθημα Πληροφορικής
- Το μάθημα απευθύνεται επίσης σε:
- προπτυχιακούς φοιτητές (τελειόφοιτους) Πληροφορικής
- Ό,τι πληροφορία χρειαστείτε για το μάθημα (συμβόλαιο, πρόγραμμα μαθημάτων, παραπομπές βιβλιογραφίας, επικοινωνία και ανακοινώσεις) θα την βρείτε στην
- Ιστοσελίδα: http://www.cs.ucy.ac.cy/courses/EPL660

Διαλέξεις - Εργαστήριο

- Μία 3-ωρη διάλεξη την εβδομάδα (Τρίτη – 15.00-18.00)
- Εργαστήριο:
  - Θα γίνονται παρουσιάσεις σχετικά με εργαλεία Ανάκτησης Πληροφοριών (Lucene, Hadoop), συζητήσεις για θέματα του μαθήματος (ασκήσεις, εργασίες κλπ).

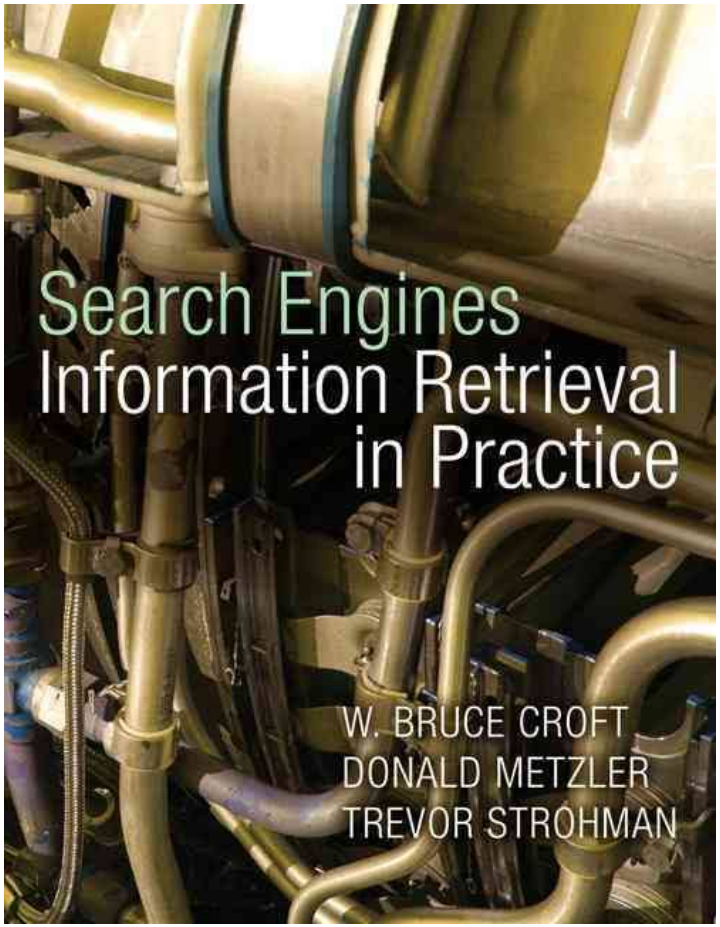
Βιβλιογραφία

- Βιβλίο μαθήματος:
- **An Introduction to Information Retrieval,** Manning, Raghavan, Schütze, Cambridge University Press (υπάρχει στο Διαδίκτυο σε pdf και html μορφότυπο).
- Άρθρα από την επιστημονική βιβλιογραφία
- Για περισσότερες πληροφορίες, ανατρέξετε στην ιστοσελίδα *"Resources"* του ιστιακού τόπου του μαθήματος.



Βιβλιογραφία II

- Search Engines – Information Retrieval in Practice by Donald Metzler, Trevor Strohman, and W. Bruce Croft
- Υπάρχει στο Διαδίκτυο σε pdf: http://ciir.cs.umass.edu/irbook/



Αξιολόγηση

- 2 γραπτές εξετάσεις:
- Ενδιάμεση (20%) – 30 Οκτ. 2018
- Τελική (45%)
- 2 σειρές ασκήσεων: (10%)
- Εργασία 6-μήνου (25%)

Περιγραφή Μαθήματος

Σκεπτικό

- Τα *Συστήματα Ανάκτησης Πληροφοριών* (Information Retrieval systems) επιτρέπουν την πρόσβαση σε **μεγάλους** όγκους πληροφοριών αποθηκευμένων με τη μορφή *κειμένου*, *φωνής*, video, ή σε σύνθετη μορφή όπως *Ιστοσελίδες*.
- **Σκοπός** των συστημάτων αυτών είναι η **ανάκτηση** μόνο εκείνων των εγγράφων που είναι ***συναφή*** με αυτό που αναζητεί ο χρήστης. Για να το επιτύχουν πρέπει να αντιμετωπίσουν την ***αβεβαιότητα*** ως προς το τι πραγματικά αναζητεί ο χρήστης και ποιο το θέμα ενός εγγράφου.

Σκοπός του Μαθήματος

- Εισαγωγή στην περιοχή των συστημάτων ανάκτησης πληροφοριών και των Μηχανών Αναζήτησης. Εξέταση των *θεωρητικών* και *πρακτικών* ζητημάτων που σχετίζονται με τη σχεδίαση, υλοποίηση και αξιολόγηση τέτοιων συστημάτων.

Διάρθρωση Μαθήματος

- Μπούλειος Ανάκτηση Πληροφοριών
- Κωδικοποίηση κειμένου, λημματοποίηση, στελέχωση κειμένων
- Λεξικά και ανάκτηση ανεκτική σε σφάλματα
- Κατασκευή και συμπίεση ευρετηρίων
- Διαβάθμιση όρων
- Ανάκτηση διανυσματικού χώρου
- Αξιολόγηση ανάκτησης πληροφοριών
- Μηχανισμοί ανάδρασης και διαστολή επερωτήσεων
- Ταξινόμηση κειμένου και απλοϊκές τεχνικές Bayes
- Ταξινόμηση διανυσματικού χώρου
- Επίπεδη ομαδοποίηση/Ιεραρχική ομαδοποίηση
- Βασικές έννοιες αναζήτησης στον Ιστό
- Αναζήτηση λογισμικού σε υπολογιστικές νεφέλες
- Αναζήτηση σε ημι-δομημένα δεδομένα
- Ιχνηλασία και ευρετηριασμός Ιστού
- Ανάλυση υπερσυνδέσμων

Why information retrieval

- An essential tool to deal with information overload

Search Computing Project

A class of queries search engines are not good at

- "Where can I attend an interesting scientific conference in my field and at the same time relax on a beautiful beach nearby?"
- "Retrieve jobs as Java developer in the Silicon Valley, nearby affordable fully-furnished flats, and close to good schools
- "Find a theater close to Union Square, San Francisco, showing a recent thriller movie, close to a steak house?"

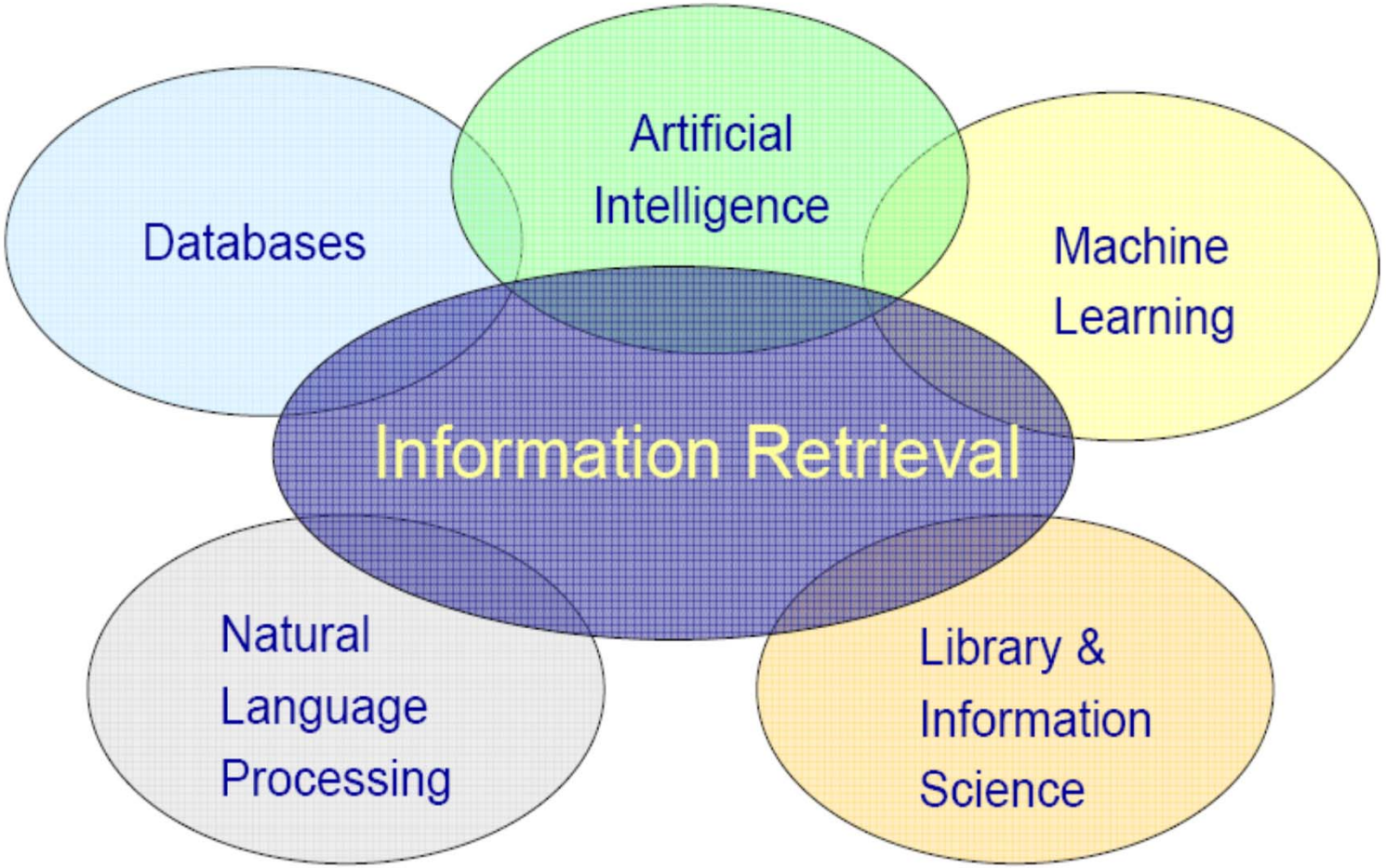
Due to query complexity, not data heterogeneity or unavailability

- With a complex notion of "best"
  - with many factors contributing to optimality
- Involving several different data sources
  - possibly hidden in the deep Web
  - typically returning ranked results (search services)
- With possibly articulated "join" conditions
  - capturing search sessions rather than one-shot queries

Τι είναι η ΑΠ;

- Πολλές μηχανές αναζήτησης είναι
  - Αρκετά αποτελεσματικές
  - Αναγνωρίσιμες και γνωστές
  - Εμπορικά επιτυχημένες (τουλάχιστον μερικές)
- Τι συμβαίνει όμως στο **παρασκήνιο** ;
  - **Πως** δουλεύουν?
  - Πως μπορούμε να κρίνουμε αν **δουλεύουν καλά**;
  - Πως μπορούμε να τις κάνουμε **πιο αποτελεσματικές**;
  - Πως μπορούμε να τις κάνουμε να λειτουργούν **πιο γρήγορα**;
  - Υπάρχει τίποτα παραπάνω από αυτό που βλέπουμε στον Παγκόσμιο Ιστό;

Relevant Areas



Market Salary

The median advertised salary for professionals with big data expertise is $124,000 a year. Sample jobs in this category include Software Engineer, Big Data Platform Engineer, Information Systems Developer, Platform Software Engineer, Data Quality Director, and many others. The distribution of median salaries across all industries shown below:

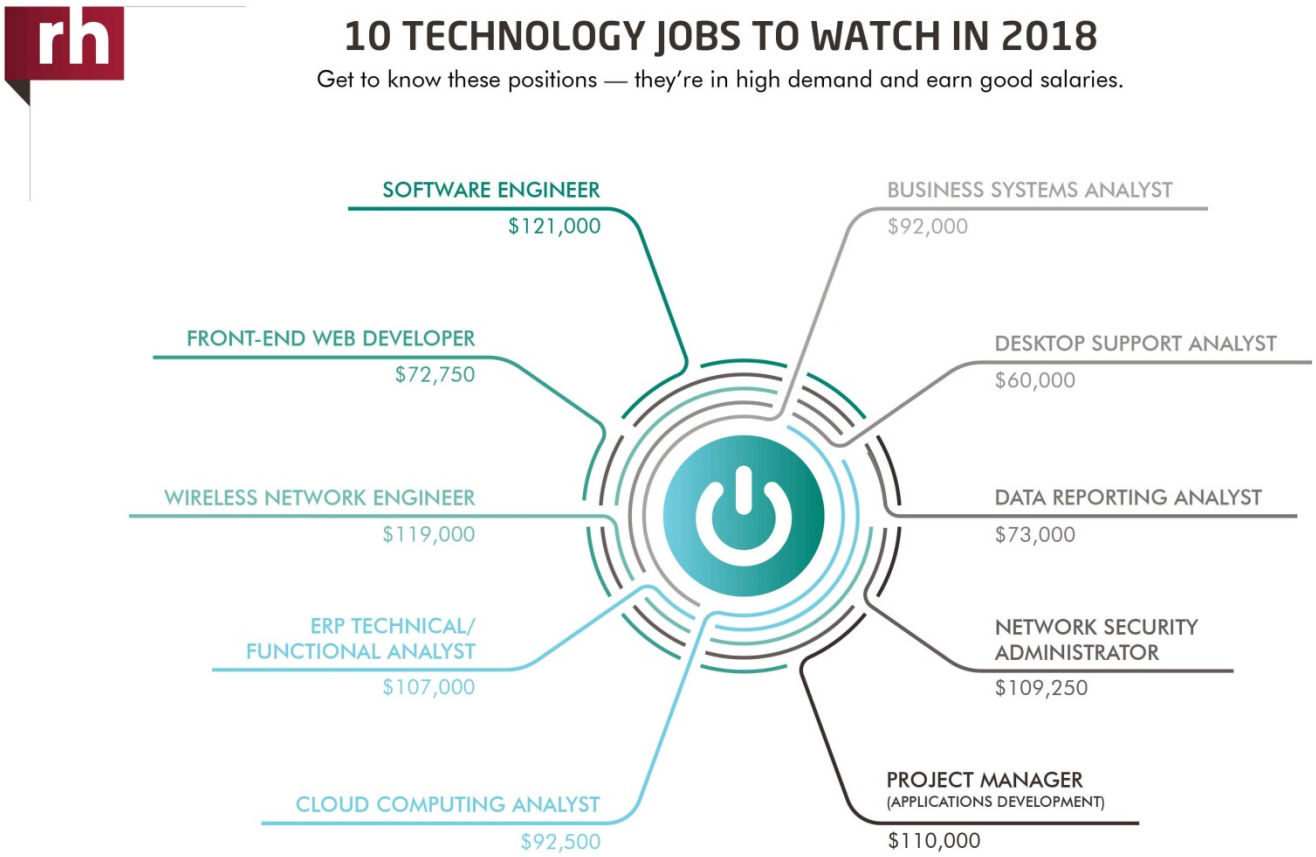
2017 Global Market Capitalization Leaderboard = Tech = 40% of Top 20 Companies…100% of Top 5…

Rank	Company	Region	Industry Segment	Current Market Value ($B)	2016 Revenue ($B)
1	Apple	USA	Tech – Hardware	$801	$218
2	Google / Alphabet	USA	Tech – Internet	680	90
3	Microsoft	USA	Tech – Software	540	86
4	Amazon	USA	Tech – Internet	476	136
5	Facebook	USA	Tech – Internet	441	28
6	Berkshire Hathaway	USA	Financial Services	409	215
7	Exxon Mobil	USA	Energy	346	198
8	Johnson & Johnson	USA	Healthcare	342	72
9	Tencent	China	Tech – Internet	335	22
10	Alibaba	China	Tech – Internet	314	21
11	JP Morgan Chase	USA	Financial Services	303	90
12	ICBC	China	Financial Services	264	85
13	Nestlé	Switzerland	Food / Beverages	263	88
14	Wells Fargo	USA	Financial Services	262	85
15	Samsung Electronics	Korea	Tech – Hardware	259	168
16	General Electric	USA	Industrial	238	120
17	Wal-Mart	USA	Retail	237	486
18	AT&T	USA	Telecom	234	164
19	Roche	Switzerland	Healthcare	233	51
20	Bank of America	USA	Financial Services	231	80
Total				$7,207	$2,497

KLEINER PERKINS

Source: CapIQ. Market value data as of 5/26/17
Note: For public companies, colors denote current market value relative to Y/Y market value. Green = higher, red = lower.

Top 10 Technology Jobs to Watch for in 2018



USC ISI to Develop Translation and Information-Retrieval System for Uncommon Languages

Caitlin Dawson | January 8, 2018

Researchers receive $16.7 million grant to automatically translate and summarize "low-resource" language documents into English

History of information retrieval

- Idea popularized in the pioneer article "***As We May Think***" by Vannevar Bush, 1945
  - *"Wholly new forms of encyclopedias will appear, ready-made with a mesh of associative trails running through them, ready to be dropped into the memex and there amplified."* **- > WWW**
  - *"A memex is a device in which an individual stores all his books, records, and communications, and which is mechanized so that it may be consulted with exceeding speed and flexibility."* **- > Search engine**

Major research milestones

- Early days (late 1950s to 1960s): foundation of the field
  - Luhn's work on automatic indexing
  - Cleverdon's Cranfield evaluation methodology and index experiments
  - Salton's early work on SMART system and experiments
- 1970s-1980s: a large number of retrieval models
  - Vector space model
  - Probabilistic models
- 1990s: further development of retrieval models and new tasks
  - Language models
  - TREC evaluation
  - Web search
- 2000s-present: more applications, especially Web search and interactions with other fields
  - Learning to rank
  - Scalability (e.g., MapReduce)
  - Real-time search

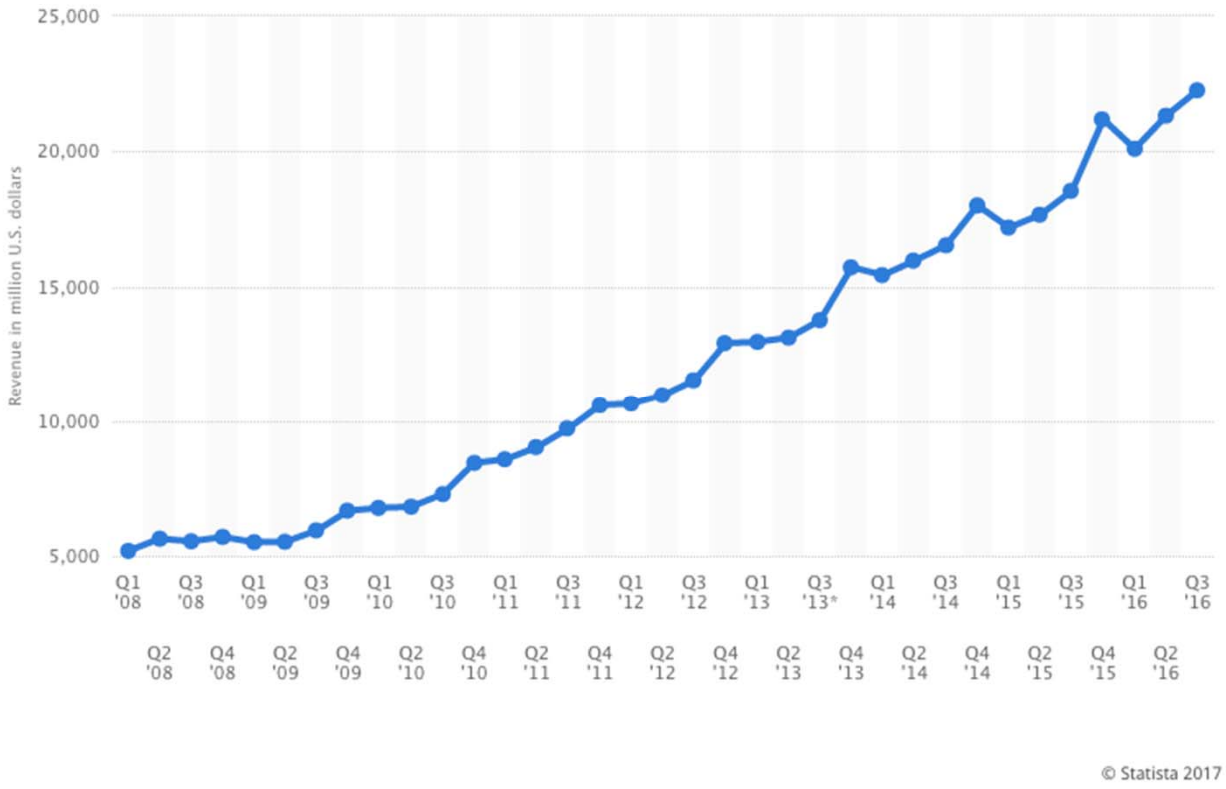
History of information retrieval

- Catalyst
  - Academia: Text Retrieval Conference (TREC) in 1992
    - "*Its purpose was to support research within the information retrieval community by providing the infrastructure necessary for large-scale evaluation of text retrieval methodologies.*"
    - "*… about one-third of the improvement in web search engines from 1999 to 2009 is attributable to TREC. Those enhancements likely saved up to 3 billion hours of time using web search engines*."
    - Till today, it is still a major test-bed for academic research in IR

History of information retrieval

- Catalyst
  - Industry: web search engines
    - WWW unleashed explosion of published information and drove the innovation of IR techniques
    - First web search engine: "*Oscar Nierstrasz at the University of Geneva wrote a series of Perl scripts that periodically mirrored these pages and rewrote them into a standard format*." Sept 2, 1993
    - Lycos (started at CMU) was launched and became a major commercial endeavor in 1994
    - Booming of search engine industry: *Magellan*, *Excite*, *Infoseek*, *Inktomi*, *Northern Light*, *AltaVista*, *Yahoo!*, *Google*, and *Bing*

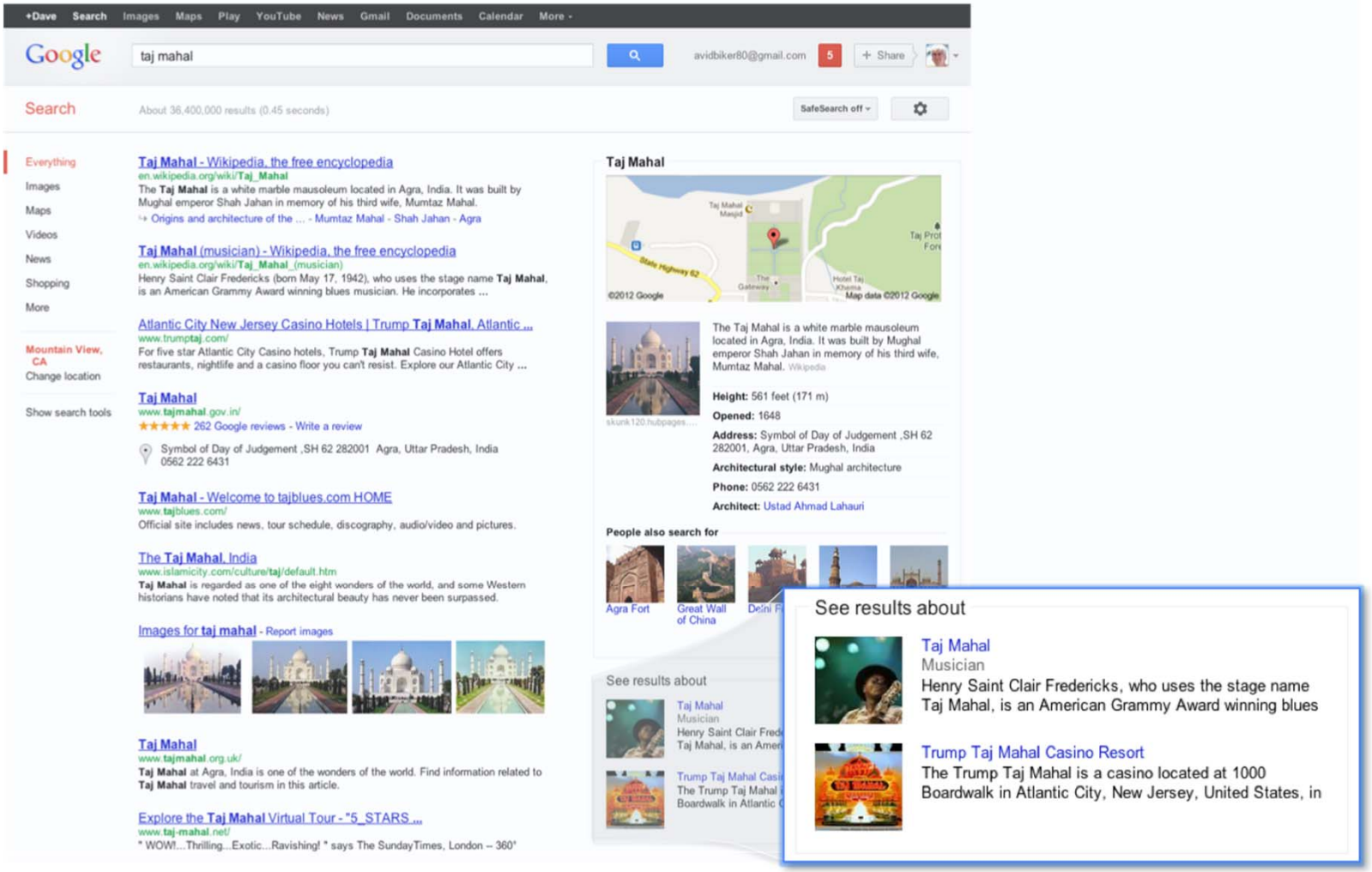
Google Revenue



Google Knowledge Graph

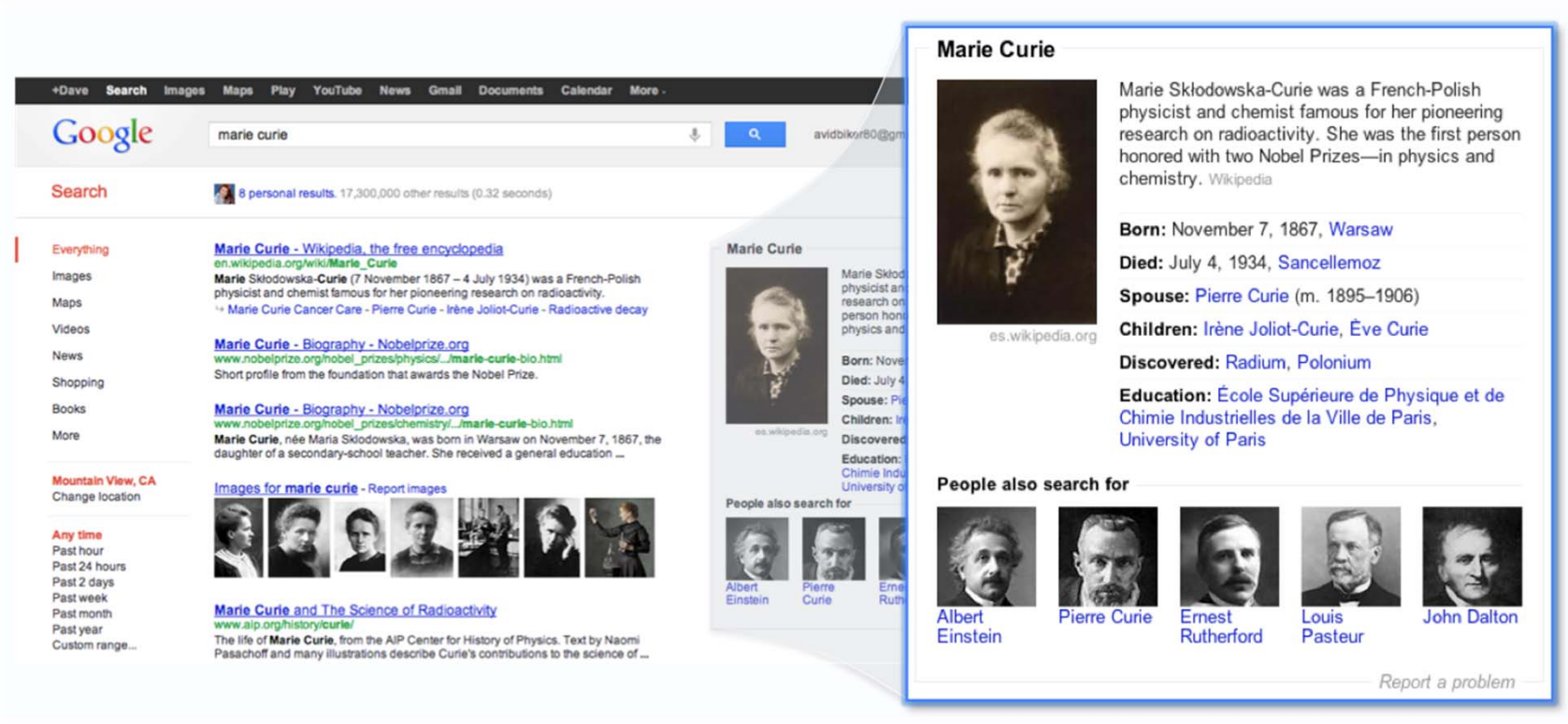
- The **Knowledge Graph** is a **knowledge base** used by Google to enhance its search engine's search results with semantic-search information gathered from a wide variety of sources. Knowledge Graph display was added to Google's search engine in 2012, starting in the United States, having been announced on May 16, 2012.
- http://www.google.com/insidesearch/features/search/knowledge.html

Google

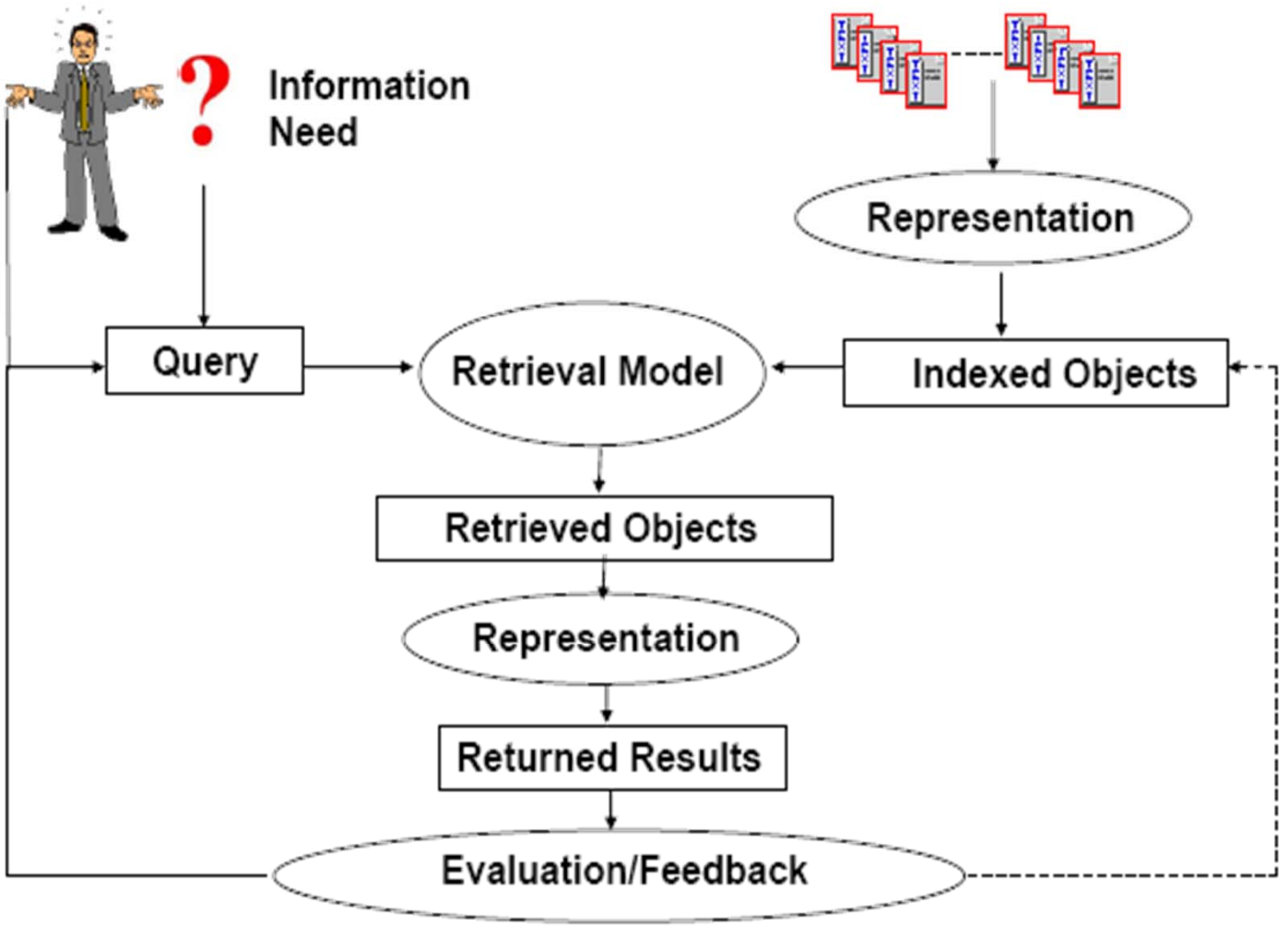


Slides by Manning, Raghavan, Schutze

Google


+Dave Search Images Maps Play YouTube News Gmail Documents Calendar More
Google
marie curie
Search
8 personal results. 17,300,000 other results (0.32 seconds)
Everything
Images
Maps
Videos
News
Shopping
Books
More
Mountain View, CA
Change location
Any time
Past hour
Past 24 hours
Past 2 days
Past week
Past month
Past year
Custom range...
Marie Curie - Wikipedia, the free encyclopedia
en.wikipedia.org/wiki/Marie_Curie
Marie Skłodowska-Curie (7 November 1867 – 4 July 1934) was a French-Polish physicist and chemist famous for her pioneering research on radioactivity.
Marie Curie Cancer Care - Pierre Curie - Irène Joliot-Curie - Radioactive decay
Marie Curie - Biography - Nobelprize.org
www.nobelprize.org/nobel_prizes/physics/.../marie-curie-bio.html
Short profile from the foundation that awards the Nobel Prize.
Marie Curie - Biography - Nobelprize.org
www.nobelprize.org/nobel_prizes/chemistry/.../marie-curie-bio.html
Marie Curie, née Maria Skłodowska, was born in Warsaw on November 7, 1867, the daughter of a secondary-school teacher. She received a general education ...
Images for marie curie - Report images
Marie Curie and The Science of Radioactivity
www.aip.org/history/curie/
The life of Marie Curie, from the AIP Center for History of Physics. Text by Naomi Pasachoff and many illustrations describe Curie's contributions to the science of ...
Marie Curie
es.wikipedia.org
People also search for
Albert Einstein
Pierre Curie
Marie Curie
Marie Skłodowska-Curie was a French-Polish physicist and chemist famous for her pioneering research on radioactivity. She was the first person honored with two Nobel Prizes—in physics and chemistry. Wikipedia
Born: November 7, 1867, Warsaw
Died: July 4, 1934, Sancellemoz
Spouse: Pierre Curie (m. 1895–1906)
Children: Irène Joliot-Curie, Ève Curie
Discovered: Radium, Polonium
Education: École Supérieure de Physique et de Chimie Industrielles de la Ville de Paris, University of Paris
People also search for
Albert Einstein
Pierre Curie
Ernest Rutherford
Louis Pasteur
John Dalton
Report a problem

Some core concepts of IR



Search and Information Retrieval

- Search on the Web is a daily activity for many people throughout the world
- Search and communication are most popular uses of the computer
- Applications involving search are everywhere
- The field of computer science that is most involved with R&D for search is *information retrieval (IR)*

Information Retrieval

- *"Information retrieval is a field concerned with the structure, analysis, organization, storage, searching, and retrieval of information."* (Salton, 1968)
- Information Retrieval (IR) is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from within large collections (usually stored on computers).
- General definition that can be applied to many types of information and search applications
- Primary focus of IR since the 50s has been on *text* and *documents*

What is a Document?

- Examples:
  - web pages, email, books, news stories, scholarly papers, text messages, Word™, Powerpoint™, PDF, forum postings, patents, IM sessions, etc.
- Common properties
  - Significant text content
  - Some structure (e.g., title, author, date for papers; subject, sender, destination for email)

Documents vs. Database Records

- Database records (or *tuples* in relational databases) are typically made up of well-defined fields (or *attributes*)
  - e.g., bank records with account numbers, balances, names, addresses, social security numbers, dates of birth, etc.
- Easy to compare fields with well-defined semantics to queries in order to find matches
- Text is more difficult

Documents vs. Records

- Example bank database query
  - *Find records with balance > $50,000 in branches located in Amherst, MA.*
  - Matches easily found by comparison with field values of records
- Example search engine query
  - *bank scandals in western mass*
  - This text must be compared to the text of entire news stories

Comparing Text

- Comparing the query text to the document text and determining what is a good match is the core issue of information retrieval
- Exact matching of words is not enough
  - Many different ways to write the same thing in a "natural language" like English
  - e.g., does a news story containing the text *"bank director in Amherst steals funds"* match the query?
  - Some stories will be better matches than others

Dimensions of IR

- IR is more than just text, and more than just web search
  - although these are central
- People doing IR work with different media, different types of search applications, and different tasks

Other Media

- New applications increasingly involve new media
  - e.g., video, photos, music, speech
- Like text, content is difficult to describe and compare
  - text may be used to represent them (e.g. tags)
- IR approaches to search and evaluation are appropriate

Dimensions of IR

Content	Applications	Tasks
Text	Web search	Ad hoc search
Images	Vertical search	Filtering
Video	Enterprise search	Classification
Scanned docs	Desktop search	Question answering
Audio	Forum search	
Music	P2P search	
	Literature search	

IR Tasks

- Ad-hoc search
  - Find relevant documents for an arbitrary text query
- Filtering
  - Identify relevant user profiles for a new document
- Classification
  - Identify relevant labels for documents
- Question answering
  - Give a specific answer to a question

Big Issues in IR

- Relevance
  - What is it?
  - Simple (and simplistic) definition: A relevant document contains the information that a person was looking for when they submitted a query to the search engine
  - Many factors influence a person's decision about what is relevant: e.g., task, context, novelty, style
  - *Topical relevance* (same topic) vs. *user relevance* (everything else)

Big Issues in IR

- Relevance
  - *Retrieval models* define a view of relevance
  - *Ranking algorithms* used in search engines are based on retrieval models
  - Most models describe statistical properties of text rather than linguistic
    - i.e. counting simple text features such as words instead of parsing and analyzing the sentences
    - Statistical approach to text processing started with Luhn in the 50s
    - Linguistic features can be part of a statistical model

Big Issues in IR

- Evaluation
  - Experimental procedures and measures for comparing system output with user expectations
    - Originated in Cranfield experiments in the 60s
  - IR evaluation methods now used in many fields
  - Typically use *test collection* of documents, queries, and relevance judgments
    - Most commonly used are TREC collections
  - *Recall* and *precision* are two examples of effectiveness measures

Big Issues in IR

- Users and Information Needs
  - Search evaluation is user-centered
  - Keyword queries are often poor descriptions of actual information needs
  - Interaction and context are important for understanding user intent
  - Query refinement techniques such as *query expansion*, *query suggestion*, *relevance feedback* improve ranking

IR and Search Engines

- A search engine is the practical application of information retrieval techniques to large scale text collections
- Web search engines are best-known examples, but many others
  - *Open source* search engines are important for research and development
    - e.g., Lucene, Lemur/Indri, *Galago*
- Big issues include main IR issues but also some others

IR and Search Engines

Information Retrieval

Relevance

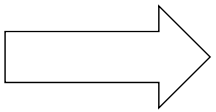
-Effective ranking

Evaluation

-Testing and measuring

Information needs

-User interaction



Search Engines

Performance

-Efficient search and indexing

Incorporating new data

-Coverage and freshness

Scalability

-Growing with data and users

Adaptability

-Tuning for applications

Specific problems

-e.g. Spam

Search Engine Issues

- Performance
  - Measuring and improving the efficiency of search
    - e.g., reducing *response time*, increasing *query throughput*, increasing *indexing speed*
  - *Indexes* are data structures designed to improve search efficiency
    - designing and implementing them are major issues for search engines

Search Engine Issues

- Dynamic data
  - The "collection" for most real applications is constantly changing in terms of updates, additions, deletions
    - e.g., web pages
  - Acquiring or "crawling" the documents is a major task
    - Typical measures are *coverage* (how much has been indexed) and *freshness* (how recently was it indexed)
  - Updating the indexes while processing queries is also a design issue

Search Engine Issues

- Scalability
  - Making everything work with millions of users every day, and many terabytes of documents
  - Distributed processing is essential
- Adaptability
  - Changing and tuning search engine components such as ranking algorithm, indexing strategy, interface for different applications

Spam

- For Web search, spam in all its forms is one of the major issues
- Affects the efficiency of search engines and, more seriously, the effectiveness of the results
- Many types of spam
  - e.g. spamdexing or term spam, link spam, "optimization"
- New subfield called *adversarial IR*, since spammers are "adversaries" with different goals

Course Goals

- To help you to understand search engines, evaluate and compare them, and modify them for specific applications
- Provide broad coverage of the important issues in information retrieval and search engines
  - includes underlying models and current research directions

Unstructured data in 1680

- Which plays of Shakespeare contain the words ***Brutus*** *AND* ***Caesar*** but *NOT* ***Calpurnia***?
- One could *grep* all of Shakespeare's plays for ***Brutus*** and ***Caesar,*** then strip out lines containing ***Calpurnia***?
- Why is that not the answer?
  - Slow (for large corpora)
  - *NOT* ***Calpurnia*** is non-trivial
  - Other operations (e.g., find the word ***Romans*** near ***countrymen***) not feasible
  - Ranked retrieval (best documents to return)
    - Later lectures

Term-document incidence

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	1	1	0	0	0	1
Brutus	1	1	0	1	0	0
Caesar	1	1	0	1	1	1
Calpurnia	0	1	0	0	0	0
Cleopatra	1	0	0	0	0	0
mercy	1	0	1	1	1	1
worser	1	0	1	1	1	0

Brutus *AND* ***Caesar*** *BUT NOT* ***Calpurnia***

1 if play contains word, 0 otherwise

Incidence vectors

- So we have a 0/1 vector for each term.
- To answer query: take the vectors for ***Brutus, Caesar*** and ***Calpurnia*** (complemented) ➔ bitwise *AND*.
- 110100 *AND* 110111 *AND* 101111 = 100100.

Answers to query

- Antony and Cleopatra, Act III, Scene ii

Agrippa [Aside to DOMITIUS ENOBARBUS]: Why, Enobarbus,
When Antony found Julius ***Caesar*** dead,
He cried almost to roaring; and he wept
When at Philippi he found ***Brutus*** slain.

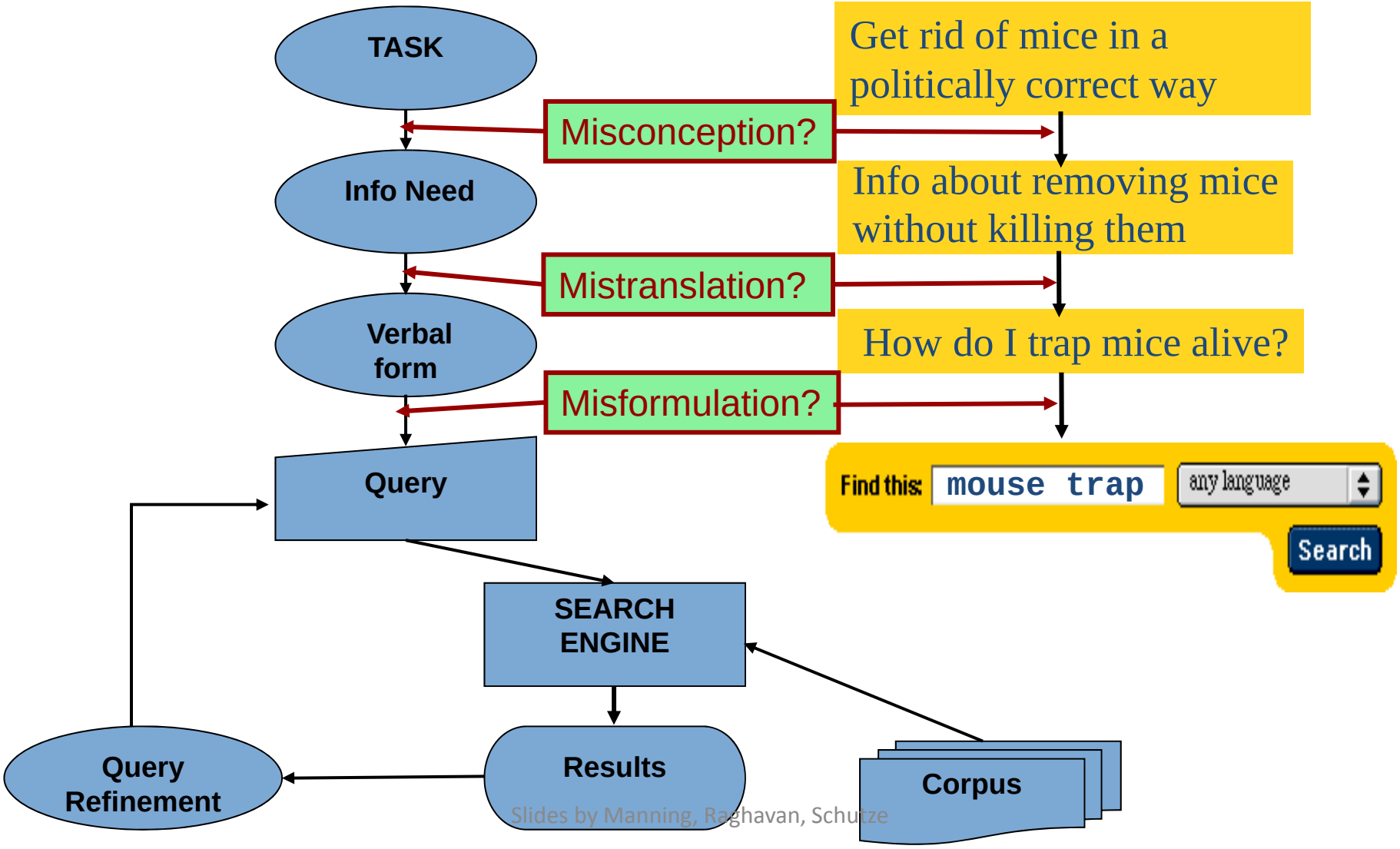
- Hamlet, Act III, Scene ii

Lord Polonius: I did enact Julius ***Caesar*** I was killed i' the Capitol; ***Brutus*** killed me.

Basic assumptions of Information Retrieval

- Collection: Fixed set of documents
- Goal: Retrieve documents with information that is relevant to the user's information need and helps the user complete a task

The classic search model



How good are the retrieved docs?

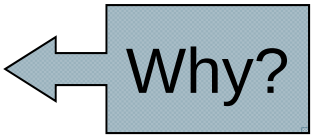
- *Precision* : Fraction of retrieved docs that are relevant to user's information need
- *Recall* : Fraction of relevant docs in collection that are retrieved
- More precise definitions and measurements to follow in later lectures

Bigger collections

- Consider *N* = 1 million documents, each with about 1000 words.
- Avg 6 bytes/word including spaces/punctuation
  - 6GB of data in the documents.
- Say there are *M* = 500K *distinct* terms among these.

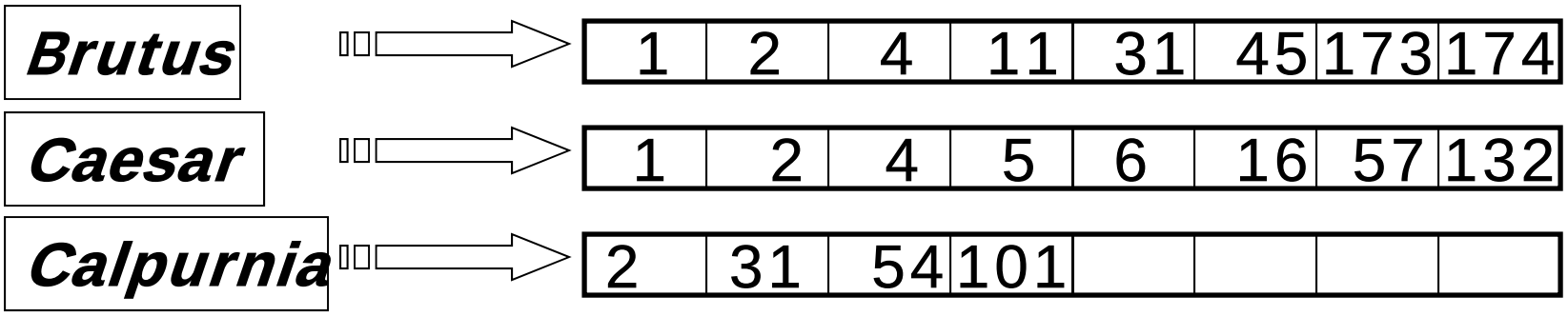
Can't build the matrix

- 500K x 1M matrix has half-a-trillion 0's and 1's.
- But it has no more than one billion 1's.
  - matrix is extremely sparse.
- What's a better representation?
  - We only record the 1 positions.



Inverted index

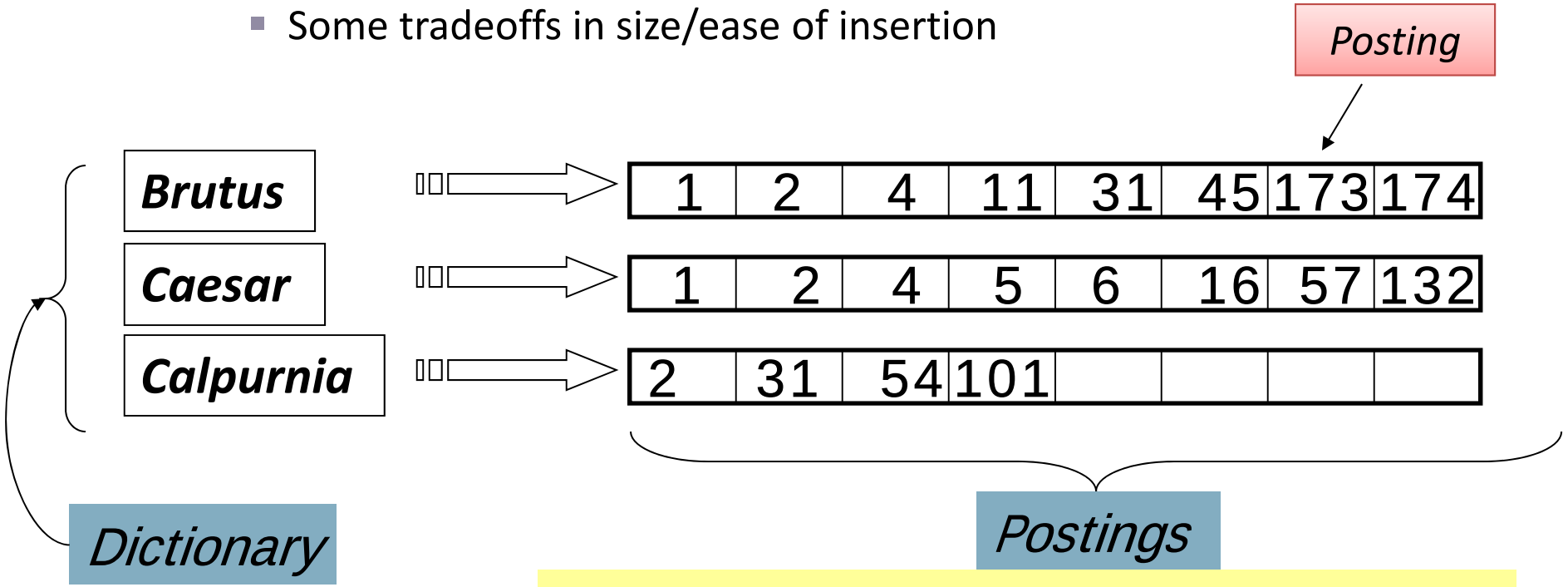
- For each term *t*, we must store a list of all documents that contain *t*.
  - Identify each by a **docID**, a document serial number
- Can we use fixed-size arrays for this?



What happens if the word ***Caesar*** is added to document 14?

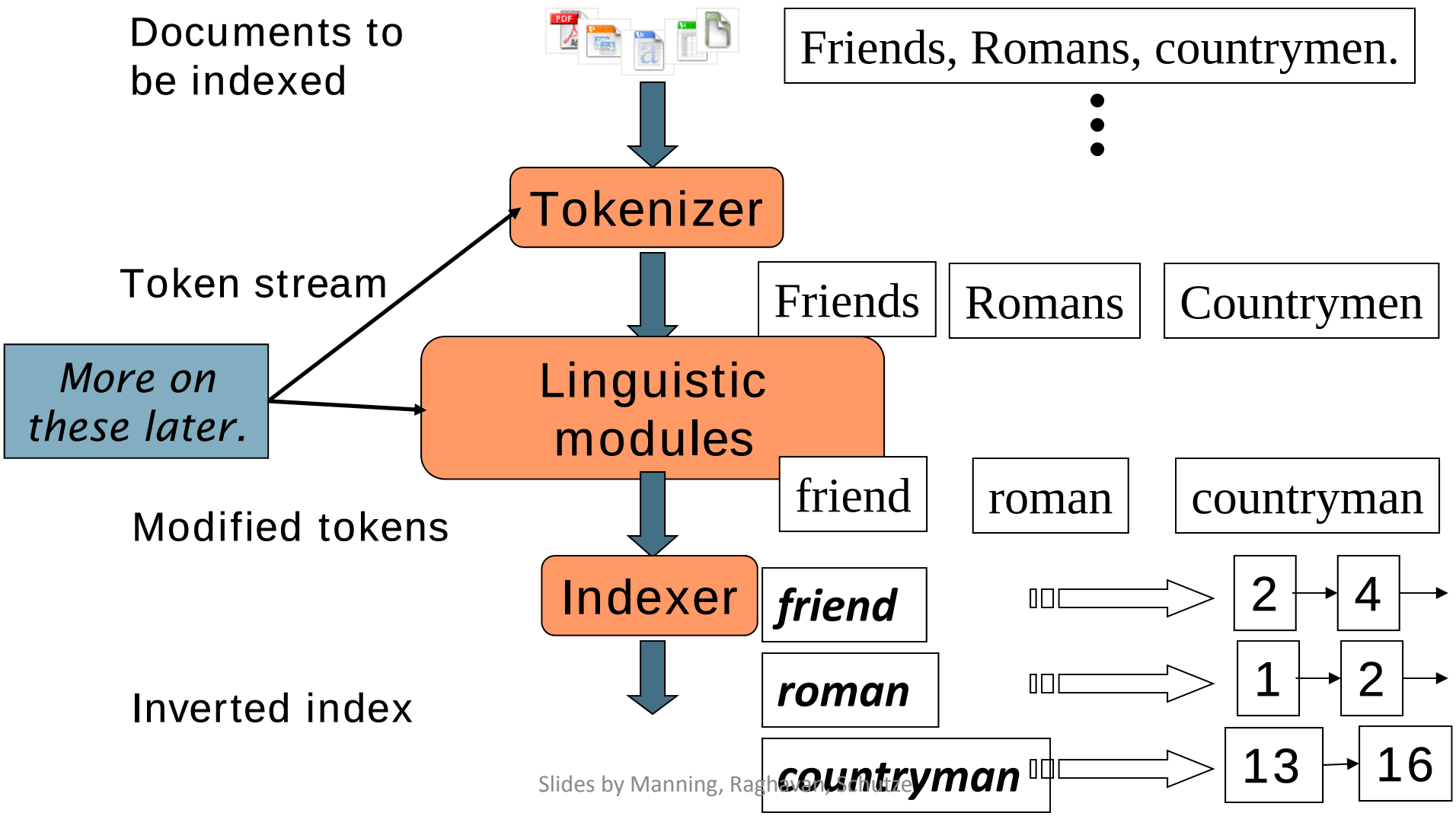
Inverted index

- We need variable-size postings lists
  - On disk, a continuous run of postings is normal and best
  - In memory, can use linked lists or variable length arrays
    - Some tradeoffs in size/ease of insertion



Sorted by docID (more later on why).

Inverted index construction



Indexer steps: Token sequence

- Sequence of (Modified token, Document ID) pairs.

Doc 1

I did enact Julius Caesar I was killed i' the Capitol; Brutus killed me.

Doc 2

So let it be with Caesar. The noble Brutus hath told you Caesar was ambitious

Term	docID
I	1
did	1
enact	1
julius	1
caesar	1
I	1
was	1
killed	1
i'	1
the	1
capitol	1
brutus	1
killed	1
me	1
so	2
let	2
it	2
be	2
with	2
caesar	2
the	2
noble	2
brutus	2
hath	2
told	2
you	2
caesar	2
was	2
ambitious	2

Indexer steps: Sort

- Sort by terms
  - And then docID

Core indexing step

Term	docID
I	1
did	1
enact	1
julius	1
caesar	1
I	1
was	1
killed	1
i'	1
the	1
capitol	1
brutus	1
killed	1
me	1
so	2
let	2
it	2
be	2
with	2
caesar	2
the	2
noble	2
brutus	2
hath	2
told	2
you	2
caesar	2
was	2
ambitious	2

Term	docID
ambitious	2
be	2
brutus	1
brutus	2
capitol	1
caesar	1
caesar	2
caesar	2
did	1
enact	1
hath	1
I	1
I	1
i'	1
it	2
julius	1
killed	1
killed	1
let	2
me	1
noble	2
so	2
the	1
the	2
told	2
you	2
was	1
was	2
with	2

Indexer steps: Dictionary & Postings

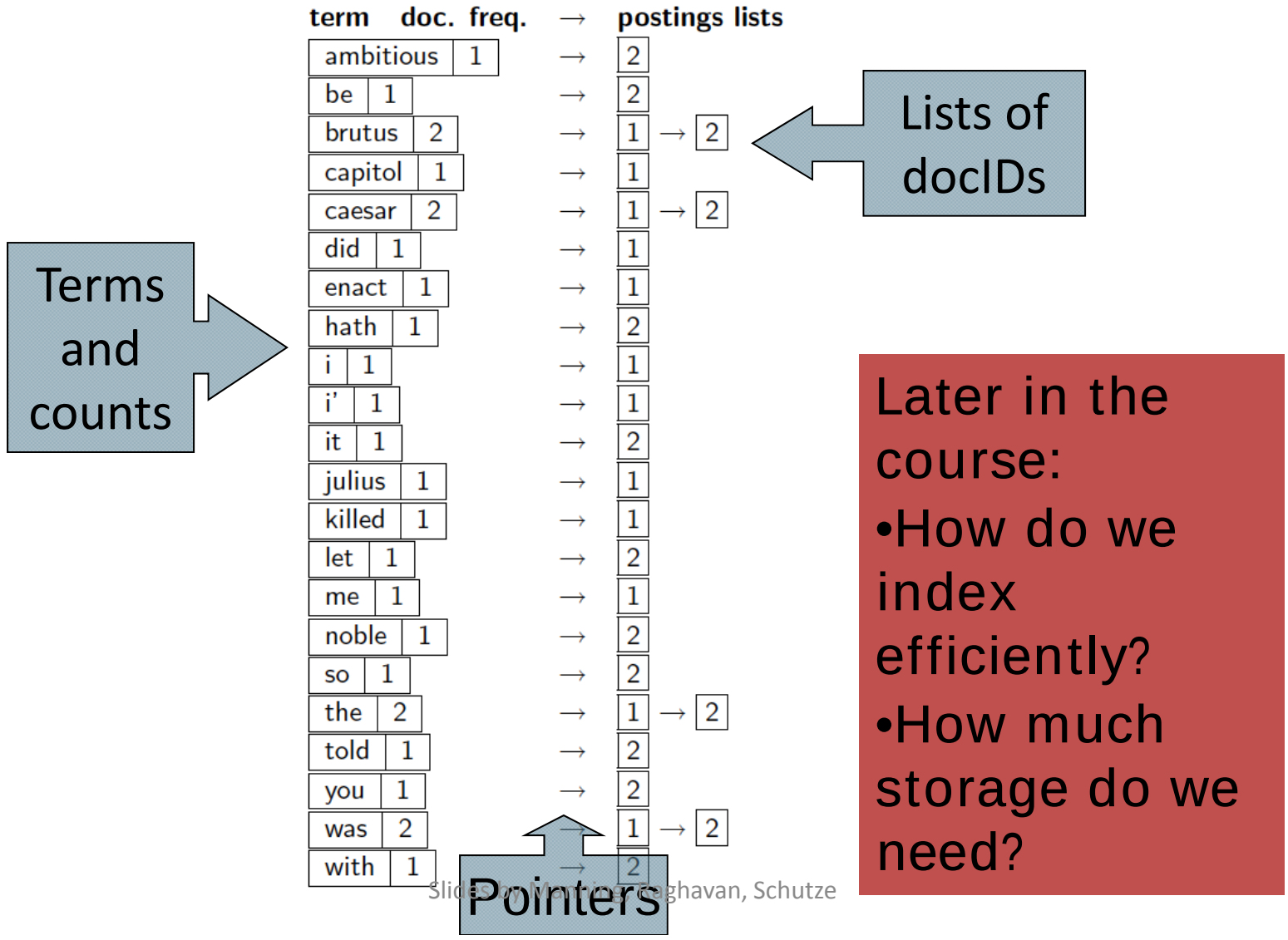
- Multiple term entries in a single document are merged.
- Split into Dictionary and Postings
- Doc. frequency information is added.

Why frequency? Will discuss later.

Term	docID
ambitious	2
be	2
brutus	1
brutus	2
capitol	1
caesar	1
caesar	2
caesar	2
did	1
enact	1
hath	1
I	1
I	1
i'	1
it	2
julius	1
killed	1
killed	1
let	2
me	1
noble	2
so	2
the	1
the	2
told	2
you	2
was	1
was	2
with	2

term	doc. freq.	→	postings lists
ambitious	1	→	2
be	1	→	2
brutus	2	→	1 → 2
capitol	1	→	1
caesar	2	→	1 → 2
did	1	→	1
enact	1	→	1
hath	1	→	2
i	1	→	1
i'	1	→	1
it	1	→	2
julius	1	→	1
killed	1	→	1
let	1	→	2
me	1	→	1
noble	1	→	2
so	1	→	2
the	2	→	1 → 2
told	1	→	2
you	1	→	2
was	2	→	1 → 2
with	1	→	2

Where do we pay in storage?



Later in the course:

- How do we index efficiently?
- How much storage do we need?

The index we just built

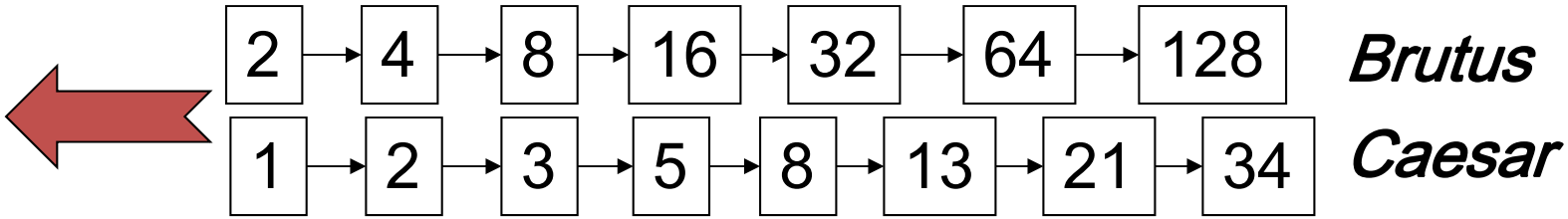
- How do we process a query?
  - Later - what kinds of queries can we process?

Query processing: AND

- Consider processing the query:

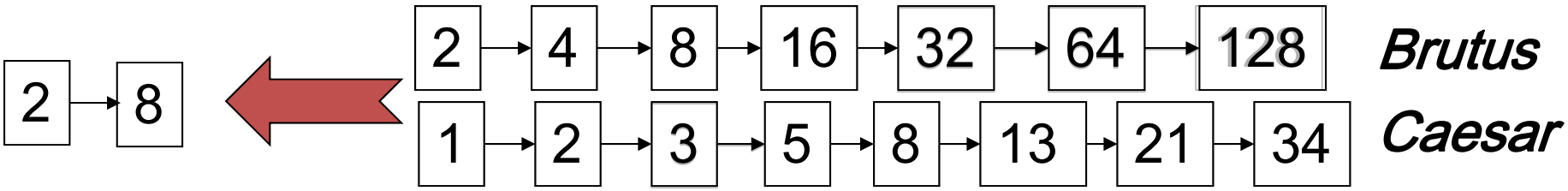
  Brutus *AND* ***Caesar***

  - Locate ***Brutus*** in the Dictionary;
    - Retrieve its postings.
  - Locate *Caesar* in the Dictionary;
    - Retrieve its postings.
  - "Merge" the two postings:



The merge

- Walk through the two postings simultaneously, in time linear in the total number of postings entries



If list lengths are *x* and *y*, merge takes O(*x*+*y*) operations.
Crucial: postings sorted by docID.

Intersecting two postings lists (a "merge" algorithm)

```
INTERSECT(p1, p2)
 1  answer ← ⟨ ⟩
 2  while p1 ≠ NIL and p2 ≠ NIL
 3  do if docID(p1) = docID(p2)
 4        then ADD(answer, docID(p1))
 5             p1 ← next(p1)
 6             p2 ← next(p2)
 7        else if docID(p1) < docID(p2)
 8                then p1 ← next(p1)
 9                else p2 ← next(p2)
10  return answer
```

Boolean queries: Exact match

- The Boolean retrieval model is being able to ask a query that is a Boolean expression:
  - Boolean Queries use *AND, OR* and *NOT* to join query terms
    - Views each document as a set of words
    - Is precise: document matches condition or not.
  - Perhaps the simplest model to build an IR system on
- Primary commercial retrieval tool for 3 decades.
- Many search systems you still use are Boolean:
  - Email, library catalog, Mac OS X Spotlight

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

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

- Another example query:
  - Requirements for disabled people to be able to access a workplace
  - disabl! /p access! /s work-site work-place (employment /3 place)
- Note that SPACE is disjunction, not conjunction!
- Long, precise queries; proximity operators; incrementally developed; not like web search
- Many professional searchers still like Boolean search
  - You know exactly what you are getting
- But that doesn't mean it actually works better….

Boolean queries: More general merges

- Exercise: Adapt the merge for the queries:

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

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

Can we still run through the merge in time O(*x+y*)?

What can we achieve?

Merging

What about an arbitrary Boolean formula?

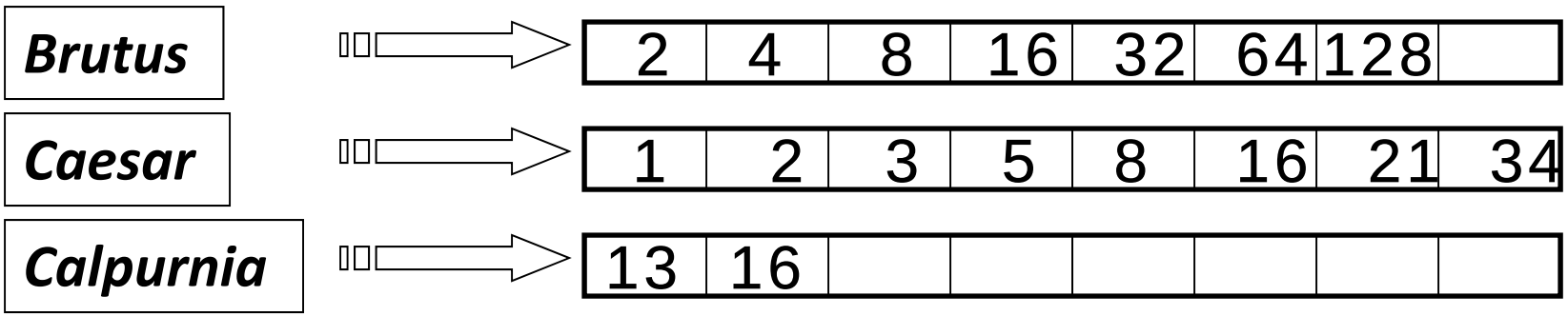
***(Brutus** OR **Caesar)** AND NOT*

(Antony** OR **Cleopatra)

- Can we always merge in "linear" time?
  - Linear in what?
- Can we do better?

Query optimization

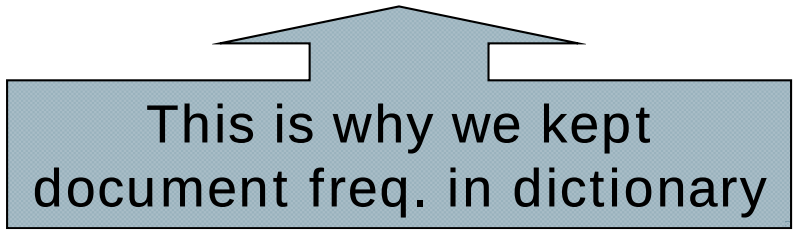
- What is the best order for query processing?
- Consider a query that is an *AND* of *n* terms.
- For each of the *n* terms, get its postings, then *AND* them together.

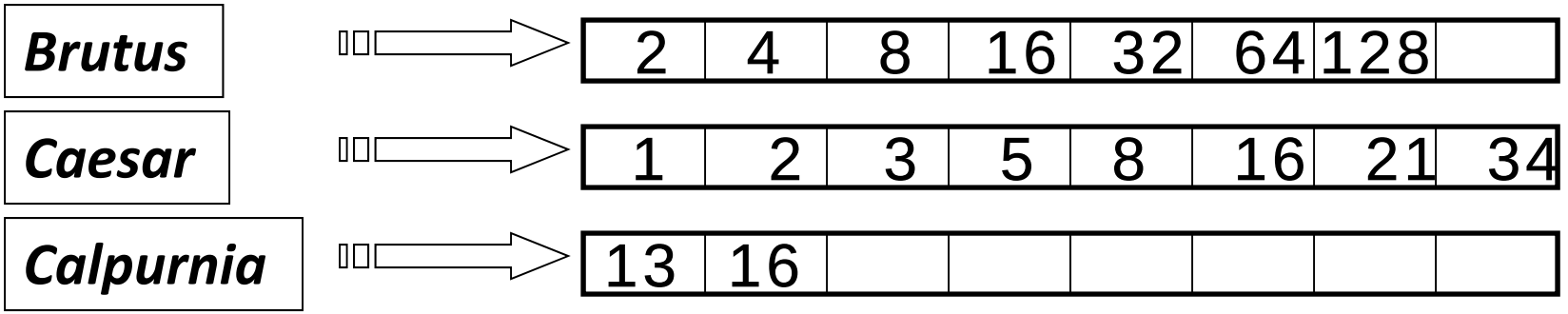


Query: ***Brutus*** *AND* ***Calpurnia*** *AND* ***Caesar***

Query optimization example

- Process in order of increasing freq:
  - *start with smallest set, then keep cutting further*.





Execute the query as (***Calpurnia*** *AND* ***Brutus)*** *AND* ***Caesar***.

More general optimization

- e.g., *(**madding** OR **crowd**) AND (**ignoble** OR **strife**)*
- Get doc. freq.'s for all terms.
- Estimate the size of each *OR* by the sum of its doc. freq.'s (conservative).
- Process in increasing order of *OR* sizes.

Exercise

- Recommend a query processing order for

***(tangerine** OR **trees)** AND*
***(marmalade** OR **skies)** AND*
(kaleidoscope** OR **eyes)

Term	Freq
eyes	213312
kaleidoscope	87009
marmalade	107913
skies	271658
tangerine	46653
trees	316812

Query processing exercises

- Exercise: If the query is ***friends*** *AND* ***romans*** *AND (NOT* ***countrymen****),* how could we use the freq of ***countrymen***?
- Exercise: Extend the merge to an arbitrary Boolean query. Can we always guarantee execution in time linear in the total postings size?
- Hint: Begin with the case of a Boolean *formula* query where each term appears only once in the query.

Exercise

- Try the search feature at http://www.rhymezone.com/shakespeare/
- Write down five search features you think it could do better

What's ahead in IR? Beyond term search

- What about phrases?
  - ***Stanford University***
- Proximity: Find ***Gates*** *NEAR* ***Microsoft***.
  - Need index to capture position information in docs.
- Zones in documents: Find documents with (*author =* ***Ullman***) *AND* (text contains ***automata***).

Evidence accumulation

- 1 vs. 0 occurrence of a search term
  - 2 vs. 1 occurrence
  - 3 vs. 2 occurrences, etc.
  - Usually more seems better
- Need term frequency information in docs

Ranking search results

- Boolean queries give inclusion or exclusion of docs.
- Often we want to rank/group results
  - Need to measure proximity from query to each doc.
  - Need to decide whether docs presented to user are singletons, or a group of docs covering various aspects of the query.

IR vs. databases: Structured vs unstructured data

- 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 data

- Typically refers to free-form text
- Allows
  - Keyword queries including operators
  - More sophisticated "concept" queries, e.g.,
    - find all web pages dealing with *drug abuse*
- Classic model for searching text documents

Semi-structured data

- In fact almost no data is "unstructured"
- E.g., this slide has distinctly identified zones such as the *Title* and *Bullets*
- Facilitates "semi-structured" search such as
  - *Title* contains data AND *Bullets* contain search

… to say nothing of linguistic structure

More sophisticated semi-structured search

- *Title* is about Object Oriented Programming AND *Author* something like stro*rup
- where * is the wild-card operator
- Issues:
  - how do you process "about"?
  - how do you rank results?
- The focus of XML search (*IIR* chapter 10)

Clustering, classification and ranking

- Clustering: Given a set of docs, group them into clusters based on their contents.
- Classification: Given a set of topics, plus a new doc *D*, decide which topic(s) *D* belongs to.

- Ranking: Can we learn how to best order a set of documents, e.g., a set of search results

The web and its challenges

- Unusual and diverse documents
- Unusual and diverse users, queries, information needs
- Beyond terms, exploit ideas from social networks
  - link analysis, clickstreams ...

- How do search engines work?
  And how can we make them better?

More sophisticated *information* retrieval

- Cross-language information retrieval
- Question answering
- Summarization
- Text mining
- …

Resources for today's lecture

- *Introduction to Information Retrieval*, chapter 1
- Shakespeare:
  - http://www.rhymezone.com/shakespeare/
  - Try the neat browse by keyword sequence feature!

Any questions?