

# Introduction to Information Retrieval

## IIR 1: Boolean Retrieval

Mihai Surdeanu

(Based on slides by Hinrich Schütze at [informationretrieval.org](http://informationretrieval.org))

Fall 2015

# Take-away

- Why you should take this course
- Admin issues
- Boolean Retrieval: Design and data structures of a simple information retrieval system
- What topics will be covered in this class?

# Why you should take this course (take 1)



# Outline

- 1 Administration
- 2 Introduction
- 3 Inverted index
- 4 Processing Boolean queries
- 5 Query optimization
- 6 Course overview

# Instructor information

Instructor: Mihai Surdeanu

Email: [msurdeanu@email.arizona.edu](mailto:msurdeanu@email.arizona.edu)

Office: Gould-Simpson 746

Office Hours: by request

TA: Enrique Noriega

Email: [enoriega@email.arizona.edu](mailto:enoriega@email.arizona.edu)

Office: Gould-Simpson 934

Office Hours: Wednesday, 2 – 3PM

# Website and syllabus

- Website:

http:

//www.surdeanu.info/mihai/teaching/cs483-fall15/

- Syllabus:

http://www.surdeanu.info/mihai/teaching/  
cs483-fall15/IR-syllabus.pdf

- Piazza:

https://piazza.com/class/idhoajqezmu1qi

- See website and syllabus for: code of conduct, classroom electronics, DRC accommodations, non-discrimination policy, confidentiality of student records, etc.

- But most materials will actually be in D2L

# Prerequisites

- Know how to program and have a decent understanding of data structures such as hash maps and trees: CSC 345
- Ideally, Math 129 (Calc 2). However, we will cover the necessary math in class.

# Prerequisites: does this look scary?

INTERSECTWITHSKIPS( $p_1, p_2$ )

```
1  answer ← ⟨ ⟩
2  while  $p_1 \neq \text{NIL}$  and  $p_2 \neq \text{NIL}$ 
3  do if  $\text{docID}(p_1) = \text{docID}(p_2)$ 
4      then ADD(answer,  $\text{docID}(p_1)$ )
5       $p_1 \leftarrow \text{next}(p_1)$ 
6       $p_2 \leftarrow \text{next}(p_2)$ 
7  else if  $\text{docID}(p_1) < \text{docID}(p_2)$ 
8      then if  $\text{hasSkip}(p_1)$  and ( $\text{docID}(\text{skip}(p_1)) \leq \text{docID}(p_2)$ )
9          then while  $\text{hasSkip}(p_1)$  and ( $\text{docID}(\text{skip}(p_1)) \leq \text{docID}(p_2)$ )
10         do  $p_1 \leftarrow \text{skip}(p_1)$ 
11         else  $p_1 \leftarrow \text{next}(p_1)$ 
12     else if  $\text{hasSkip}(p_2)$  and ( $\text{docID}(\text{skip}(p_2)) \leq \text{docID}(p_1)$ )
13         then while  $\text{hasSkip}(p_2)$  and ( $\text{docID}(\text{skip}(p_2)) \leq \text{docID}(p_1)$ )
14         do  $p_2 \leftarrow \text{skip}(p_2)$ 
15         else  $p_2 \leftarrow \text{next}(p_2)$ 
16 return answer
```

# Prerequisites: does this look scary?

$$\|x\|_2 = \sqrt{\sum_i x_i^2}$$

$$\cos(\vec{q}, \vec{d}) = \text{SIM}(\vec{q}, \vec{d}) = \frac{\vec{q} \cdot \vec{d}}{|\vec{q}| |\vec{d}|} = \frac{\sum_{i=1}^{|V|} q_i d_i}{\sqrt{\sum_{i=1}^{|V|} q_i^2} \sqrt{\sum_{i=1}^{|V|} d_i^2}}$$

Dot product, matrix multiplication, Bayes rule

# Choosing a programming language

## My recommendations

- Scala
- Java
- Python
- C/C++

# Java

- Pros

- Pretty fast
- Probably the most common language for IR and NLP
- Clean exception handling
- Statically typed
- Garbage collection
- Several great IDEs

- Cons

- Syntax too verbose
- Inconsistent semantics due to enforced backwards compatibility  
(primitive types vs. objects, equality, etc.)

# Scala

- Pros

- Pretty fast
- “Hot” language for IR, NLP, ML, distributed computing, web development
- Clean, transparent exception handling
- Clean, minimalist syntax
- Consistent semantics
- Statically typed
- Garbage collection
- At least one great IDE (IntelliJ)
- Fully compatible with Java (use all Java libraries)

- Cons

- It has some “dark corners”
- Backwards compatibility not guaranteed

# Python

- Pros

- Clean syntax
- Popular: many NLP/ML libraries exist
- Clean exception handling

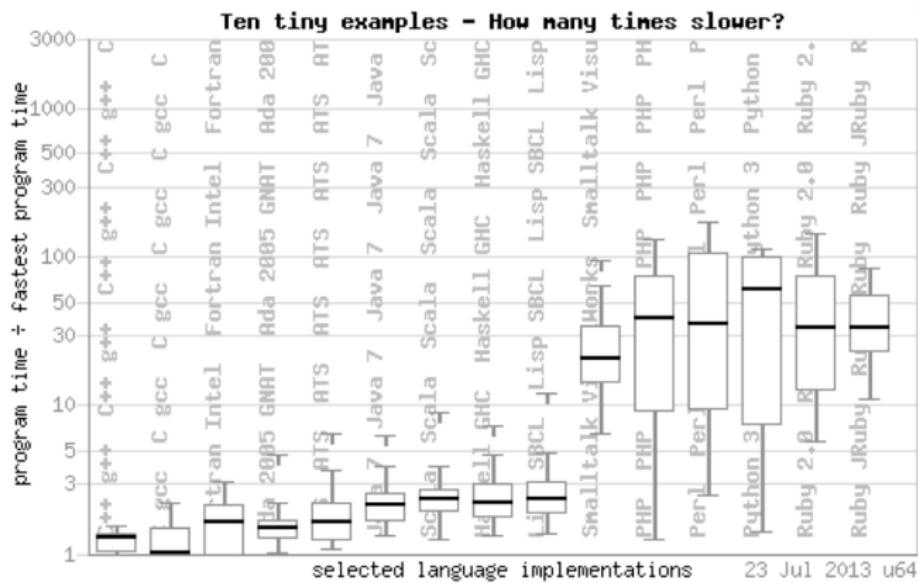
- Cons

- Slow
- Dynamically typed
- No great IDE

# C/C++

- Pros
  - As fast as it gets
- Cons
  - Too many to list

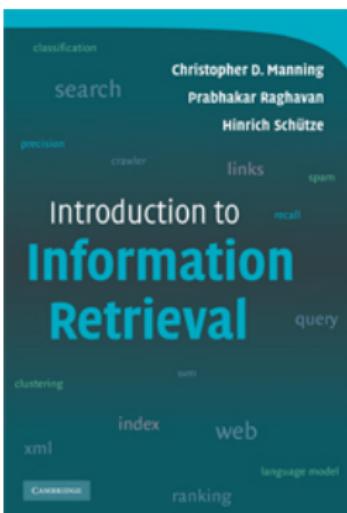
## Comparison



More benchmarks:

<http://benchmarksgame.alioth.debian.org/u64/benchmark.php?test=all&lang=all&data=u64>

# Textbook



- Introduction to Information Retrieval, by Manning et al.
- <http://nlp.stanford.edu/IR-book/>

# Grading

- Final grade = 4 assignments + 2 exams + 1 project + 1 presentation + in-class participation
- First assignment has been posted. Due September 13! [Let's take a quick look at the assignment.](#)
- Undergraduate vs. graduate requirements

# Grading scheme

Component	Weight	Grade	Point Range
Written assignments	300 pts	A	900 – 1000
Midterm Exam	200 pts	B	800 – 899
Final Exam	250 pts	C	700 – 799
Programming Project	200 pts	D	600 – 699
Final Presentation	25 pts	E	0 – 599
In-class Participation	25 pts		
Total	1000 pts		

# Final Project



- Option 1: building (parts of) Watson
- Option 2: your own idea. Use an existing IR system to implement a real-world application

# Late work + attendance policy

- Late work is not accepted, except in case of documented emergency approved by the instructor
- Attendance is required
- Students who miss class due to illness or emergency are required to bring documentation

# Dates

What	When
Midterm	October 14
Project due	December 8 before 11:59PM
Project presentations	December 9
Final	Between December 11 – 17 (TBA)

Written assignments will be due approximately every three weeks, as announced by the instructor. All assignments are due in the D2L dropbox by 11:59 P.M. on the indicated day.

# Cooperation & Cheating

- Students may not discuss individual homework with anybody other than the instructors and teaching assistants.
- Students may not share individual homework solutions with anybody.
- Students may post questions to Piazza, but should refrain from posting solutions or partial solutions.
- Students may not share test cases with anybody.
- Students may share class notes with anybody.
- Students may not seek individual homework help from anybody other than the instructors, teaching assistants, or departmental tutors.
- If permitted, the use of open source or third party materials in student submissions must be clearly identified and credited. Assignment and project submissions must be substantially the work of the student who submits the work.

# Cooperation & Cheating

- Students who violate the Code should expect a penalty that is **greater than the value of the work in question up to and including failing the course.**
- A record of the incident **will** be sent to the Dean of Students office. If you have been involved in other Code violations, the Dean of Students may impose additional sanctions.

# Outline

- 1 Administration
- 2 Introduction
- 3 Inverted index
- 4 Processing Boolean queries
- 5 Query optimization
- 6 Course overview

# Definition of *information retrieval*

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).

# Definition of *information retrieval*

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).

# Definition of *information retrieval*

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).

# Definition of *information retrieval*

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).

# Definition of *information retrieval*

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).

# Definition of *information retrieval*

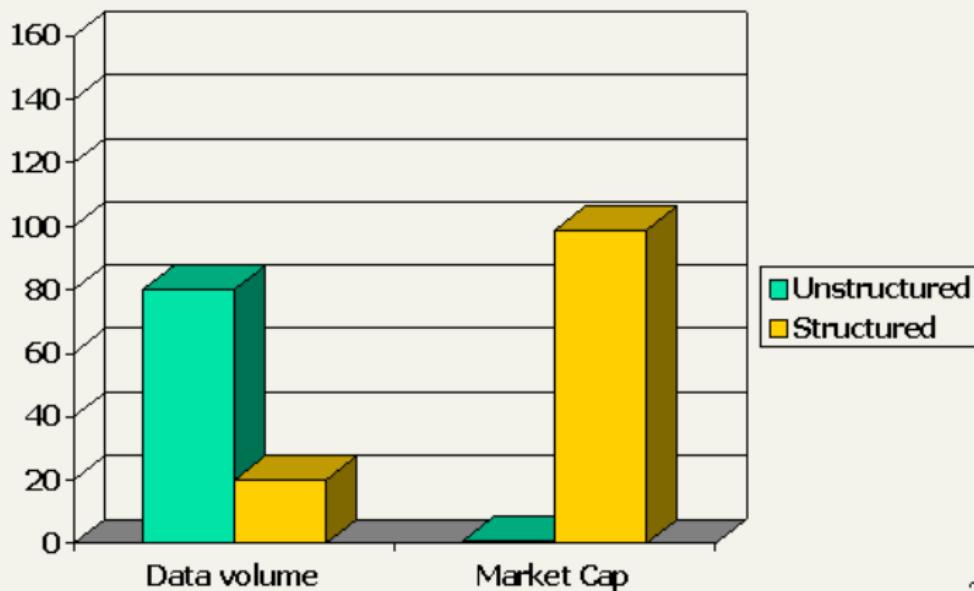
Information retrieval (IR) is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from within **large collections** (usually stored on computers).

# Definition of *information retrieval*

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).

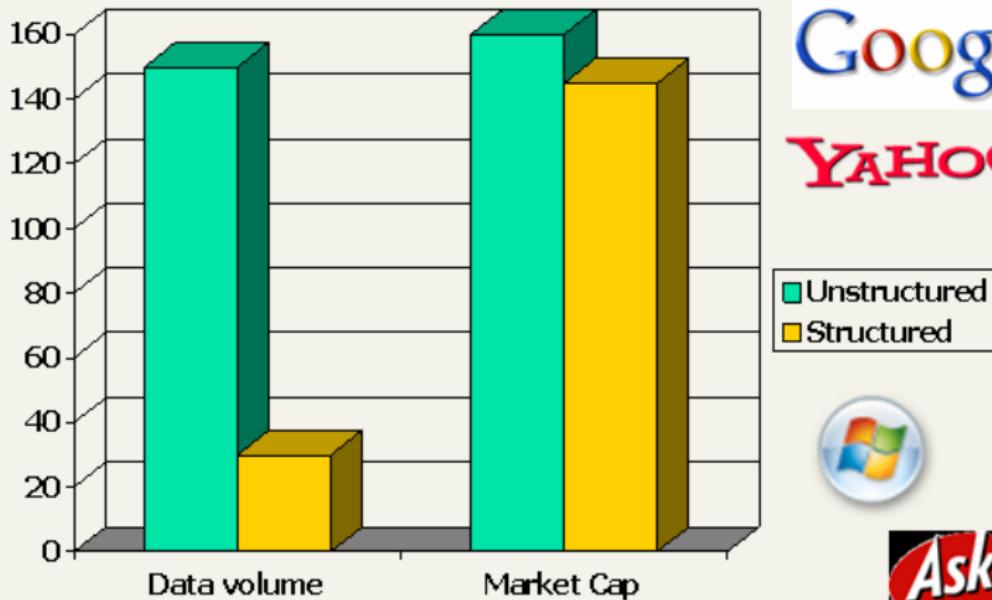
# Unstructured (text) vs. structured (database) data in 1996

---



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

---



Google™

YAHOO!®

Unstructured  
Structured



# Companies using IR

- Google, Yahoo, Microsoft: search web, email, choose ads
- Facebook: search friends' posts, choose wall
- Twitter: search tweets
- HP Autonomy: enterprise search
- Pandora: music (!) search

# Why you should take this course

- Take 2: IR is important
- Take 3:
  - This is a gateway course for data science and machine learning
  - Real-world applications of programming + data structures + probability theory

# Boolean retrieval

- The Boolean model is arguably the simplest model to base an information retrieval system on.
- Queries are Boolean expressions, e.g., CAESAR AND BRUTUS
- The search engine returns all documents that satisfy the Boolean expression.

# Boolean retrieval

- The Boolean model is arguably the simplest model to base an information retrieval system on.
- Queries are Boolean expressions, e.g., CAESAR AND BRUTUS
- The search engine returns all documents that satisfy the Boolean expression.

Does Google use the Boolean model?

# Does Google use the Boolean model?

- On Google, the default interpretation of a query  $[w_1 \ w_2 \dots w_n]$  is  $w_1 \text{ AND } w_2 \text{ AND } \dots \text{ AND } w_n$
- Cases where you get hits that do not contain one of the  $w_i$ :
  - anchor text
  - page contains variant of  $w_i$  (morphology, spelling correction, synonym)
  - long queries ( $n$  large)
  - boolean expression generates very few hits
- Simple Boolean vs. Ranking of result set
  - Simple Boolean retrieval returns matching documents in no particular order.
  - Google (and most well designed Boolean engines) rank the result set – they rank good hits (according to some estimator of relevance) higher than bad hits.

# Outline

- 1 Administration
- 2 Introduction
- 3 Inverted index
- 4 Processing Boolean queries
- 5 Query optimization
- 6 Course overview

# Unstructured data in 1650: Shakespeare



# Unstructured data in 1650

- 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 grep not the solution?

# Unstructured data in 1650

- 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 grep not the solution?
  - Slow (for large collections)
  - grep is line-oriented, IR is document-oriented
  - "NOT CALPURNIA" is non-trivial
  - Other operations (e.g., find the word ROMANS near COUNTRYMAN) not feasible

# Term-document incidence matrix

	Anthony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth	...
ANTHONY	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	

...

Entry is 1 if term occurs. Example: CALPURNIA occurs in *Julius Caesar*.

Entry is 0 if term doesn't occur. Example: CALPURNIA doesn't occur in *The tempest*.

# Term-document incidence matrix

	Anthony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth	...
ANTHONY	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	

...  
Entry is 1 if term occurs. Example: CALPURNIA occurs in Julius Caesar.

Entry is 0 if term doesn't occur. Example: CALPURNIA doesn't occur in The tempest.

# Term-document incidence matrix

	Anthony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth	...
ANTHONY	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	

...

Entry is 1 if term occurs. Example: CALPURNIA occurs in *Julius Caesar*.

Entry is 0 if term doesn't occur. Example: CALPURNIA doesn't occur in *The tempest*.

# Incidence vectors

- So we have a 0/1 vector for each term.
- To answer the query BRUTUS AND CAESAR AND NOT CALPURNIA:

# Incidence vectors

- So we have a 0/1 vector for each term.
- To answer the query BRUTUS AND CAESAR AND NOT CALPURNIA:
  - Take the vectors for BRUTUS, CAESAR, and CALPURNIA
  - Complement the vector of CALPURNIA
  - Do a (bitwise) AND on the three vectors
  - $110100 \text{ AND } 110111 \text{ AND } 101111 = 100100$

# 0/1 vector for BRUTUS

	Anthony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth	...
ANTHONY	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	
...							
result:	1	0	0	1	0	0	

# Answers to query

*Anthony 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.

# Bigger collections

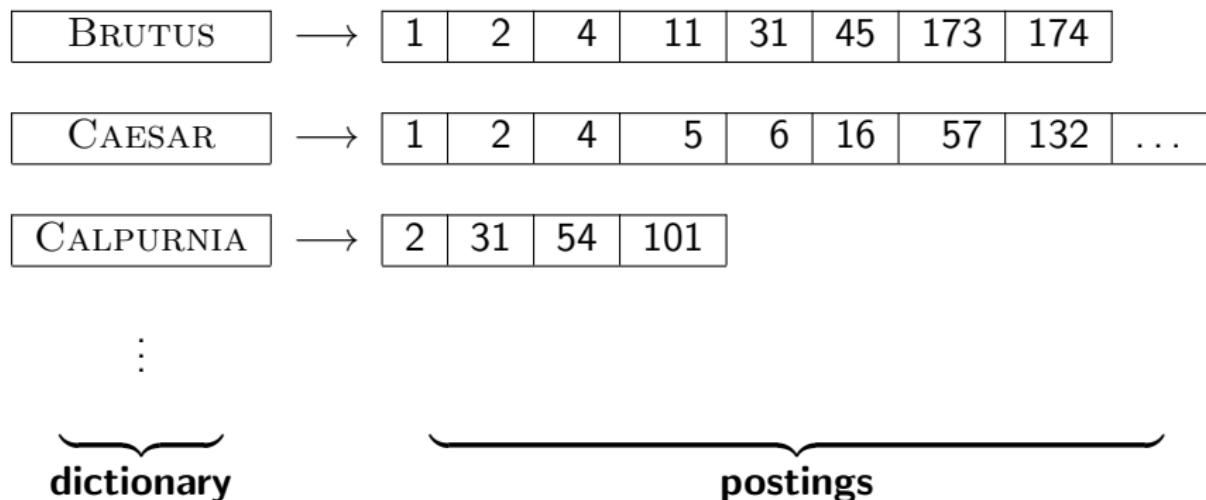
- Consider  $N = 10^6$  documents, each with about 1000 tokens
- $\Rightarrow$  total of  $10^9$  tokens
- On average 6 bytes per token, including spaces and punctuation  $\Rightarrow$  size of document collection is about  $6 \cdot 10^9 = 6$  GB
- Assume there are  $M = 500,000$  distinct terms in the collection
- (Notice that we are making a term/token distinction.)

# Can't build the incidence matrix

- $M = 500,000 \times 10^6 =$  half a trillion 0s and 1s.
- But the matrix has no more than one billion 1s.
  - Matrix is extremely sparse.
- What is a better representations?
  - We only record the 1s.

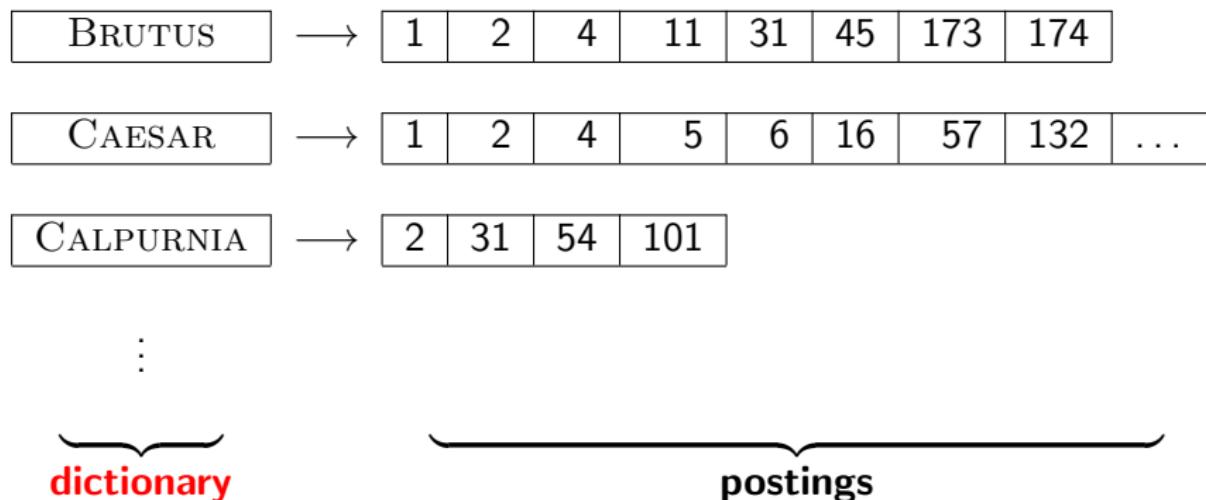
# Inverted Index

For each term  $t$ , we store a list of all documents that contain  $t$ .



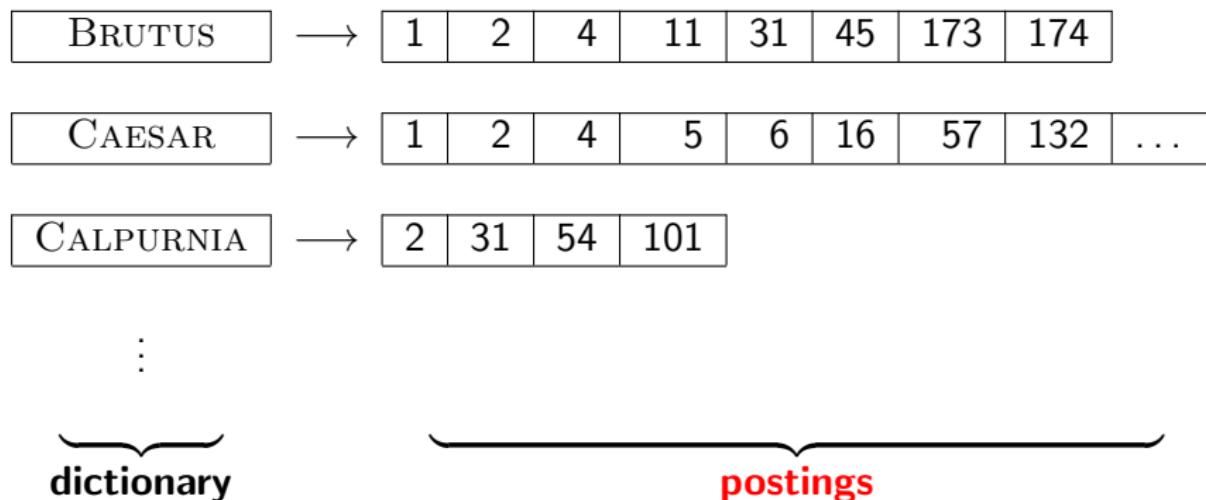
# Inverted Index

For each term  $t$ , we store a list of all documents that contain  $t$ .



# Inverted Index

For each term  $t$ , we store a list of all documents that contain  $t$ .



# Inverted index construction

- 1 Collect the documents to be indexed:

Friends, Romans, countrymen. So let it be with Caesar ...

- 2 Tokenize the text, turning each document into a list of tokens:

Friends Romans countrymen So ...

- 3 Do linguistic preprocessing, producing a list of normalized tokens, which are the indexing terms: friend roman

countryman so ...

- 4 Index the documents that each term occurs in by creating an inverted index, consisting of a dictionary and postings.

# Tokenization and preprocessing

**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:



**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

# Generate postings

**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

# Sort postings

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



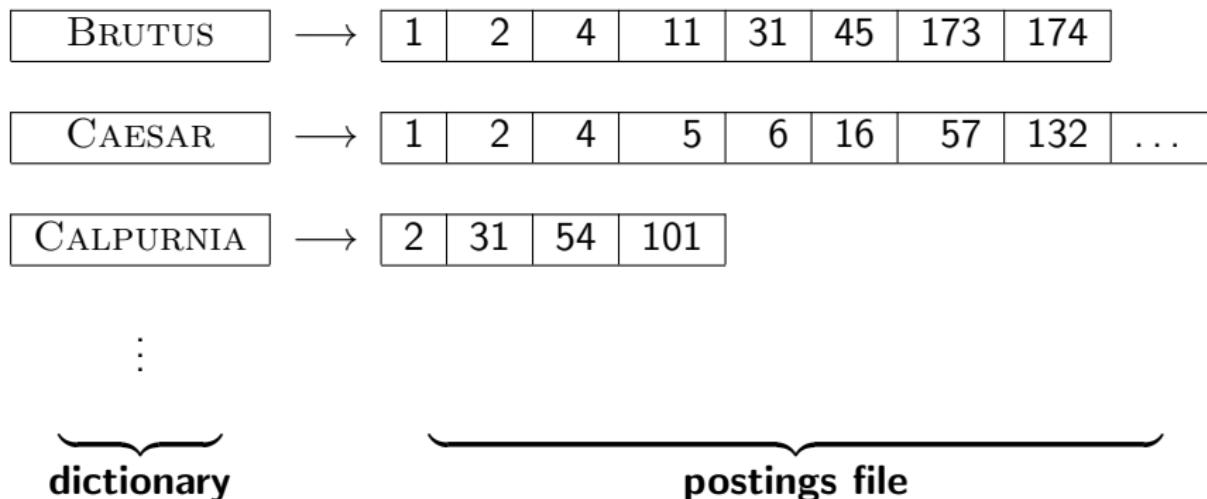
# Create postings lists, determine document frequency

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

term	doc.	freq.	→	postings	lists
ambitious	1		→	2	
be	1		→	2	
brutus	2		→	1	→ 2
capitol	1		→	1	
caesar	1		→	1	→ 2
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	
killed	1		→	2	
let	2		→	1	
me	1		→	2	
noble	1		→	2	
so	2		→	1	→ 2
the	2		→	2	
told	1		→	2	
you	1		→	2	
was	2		→	1	→ 2
with	1		→	2	

# Split the result into dictionary and postings file



# Later in this course

- Index construction: how can we create inverted indexes for large collections?
- How much space do we need for dictionary and index?
- Index compression: how can we efficiently store and process indexes for large collections?
- Ranked retrieval: what does the inverted index look like when we want the “best” answer?

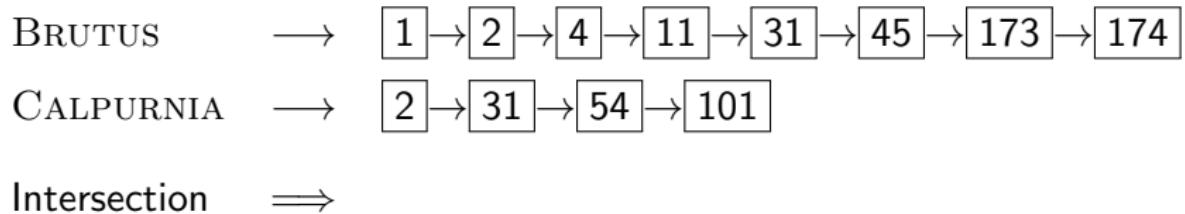
# Outline

- 1 Administration
- 2 Introduction
- 3 Inverted index
- 4 Processing Boolean queries
- 5 Query optimization
- 6 Course overview

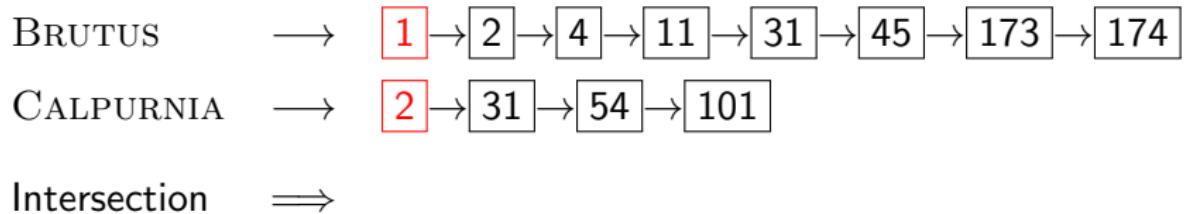
# Simple conjunctive query (two terms)

- Consider the query: BRUTUS AND CALPURNIA
- To find all matching documents using inverted index:
  - ① Locate BRUTUS in the dictionary
  - ② Retrieve its postings list from the postings file
  - ③ Locate CALPURNIA in the dictionary
  - ④ Retrieve its postings list from the postings file
  - ⑤ Intersect the two postings lists
  - ⑥ Return intersection to user

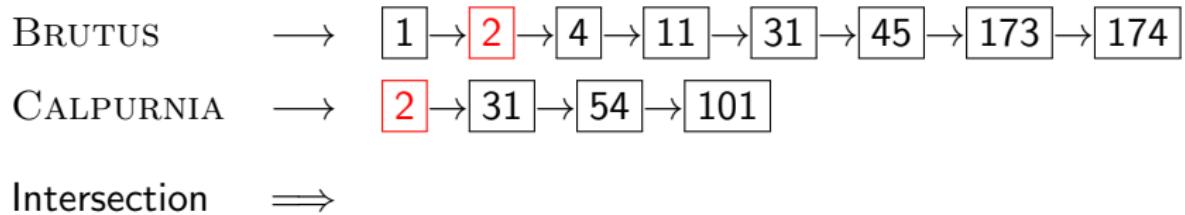
# Intersecting two postings lists



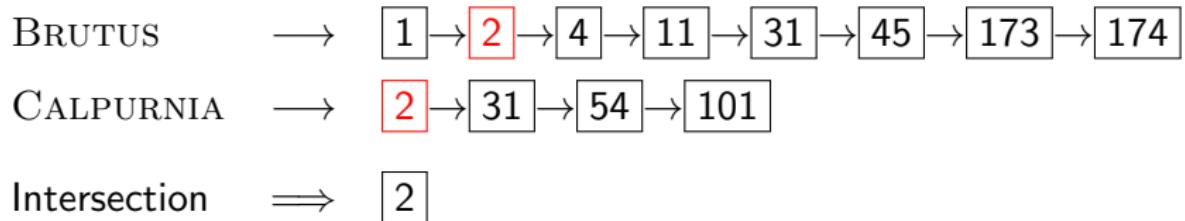
# Intersecting two postings lists



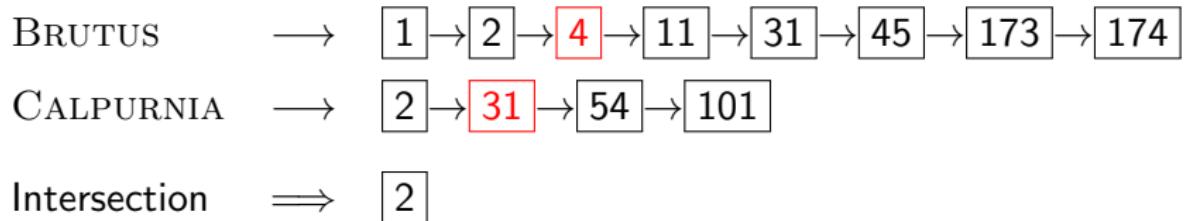
# Intersecting two postings lists



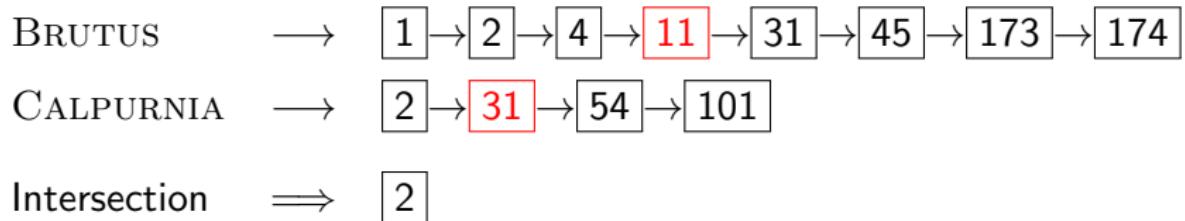
# Intersecting two postings lists



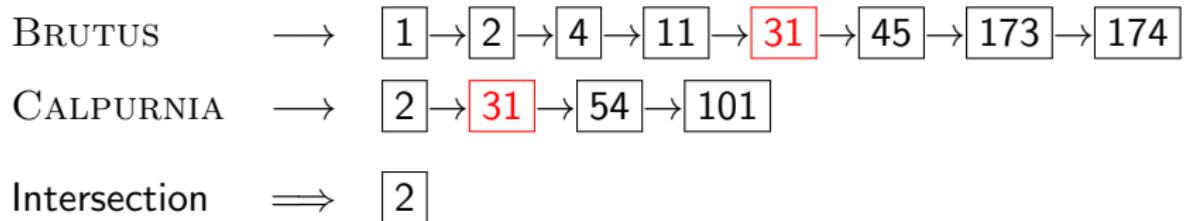
# Intersecting two postings lists



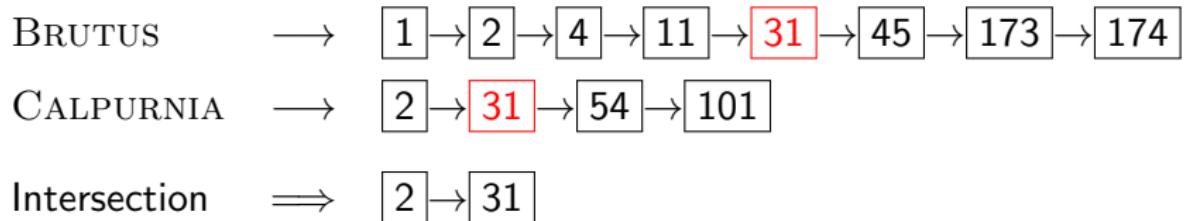
# Intersecting two postings lists



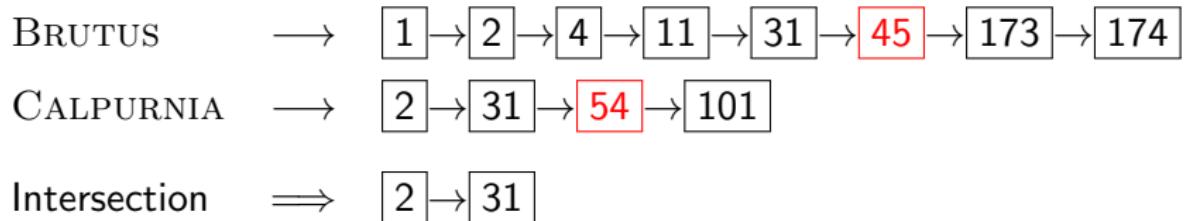
# Intersecting two postings lists



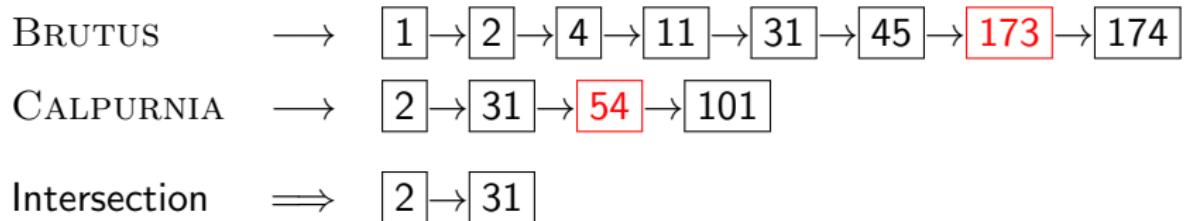
# Intersecting two postings lists



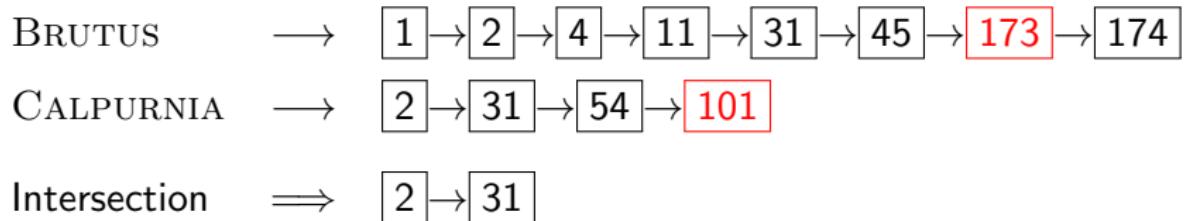
# Intersecting two postings lists



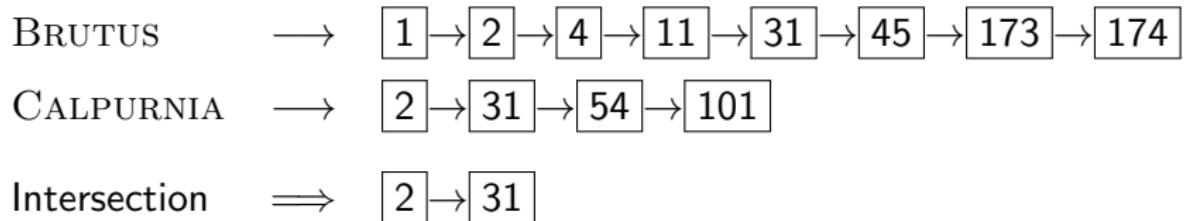
# Intersecting two postings lists



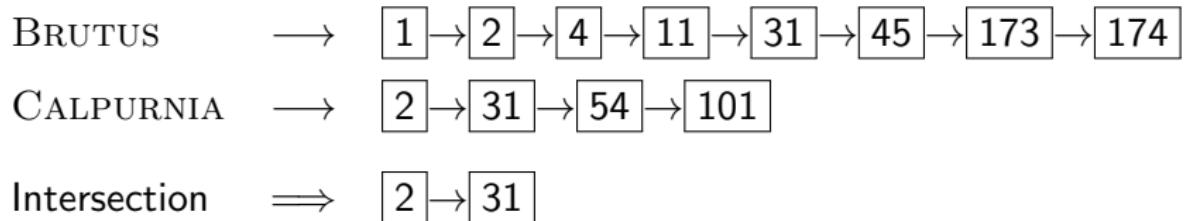
# Intersecting two postings lists



# Intersecting two postings lists

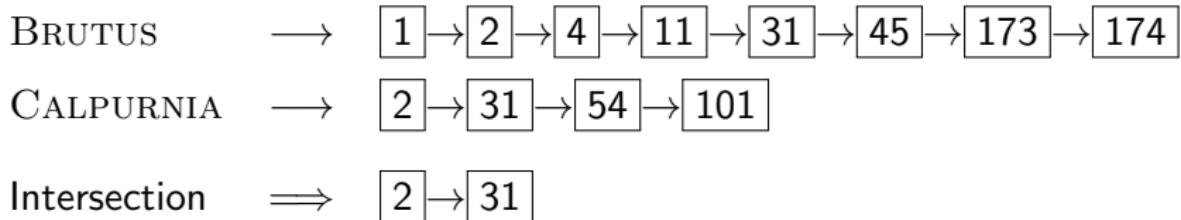


# Intersecting two postings lists



- This is linear in the length of the postings lists.

# Intersecting two postings lists



- This is linear in the length of the postings lists.
- Note: This only works if postings lists are sorted.

# Intersecting two postings lists

INTERSECT( $p_1, p_2$ )

```
1  answer ← ⟨ ⟩
2  while  $p_1 \neq \text{NIL}$  and  $p_2 \neq \text{NIL}$ 
3  do if  $\text{docID}(p_1) = \text{docID}(p_2)$ 
4      then ADD(answer, docID( $p_1$ ))
5           $p_1 \leftarrow \text{next}(p_1)$ 
6           $p_2 \leftarrow \text{next}(p_2)$ 
7      else if  $\text{docID}(p_1) < \text{docID}(p_2)$ 
8          then  $p_1 \leftarrow \text{next}(p_1)$ 
9          else  $p_2 \leftarrow \text{next}(p_2)$ 
10 return answer
```

# Query processing: Exercise

FRANCE → [1] → [2] → [3] → [4] → [5] → [7] → [8] → [9] → [11] → [12] → [13] → [14] → [15]

PARIS → [2] → [6] → [10] → [12] → [14]

LEAR → [12] → [15]

Compute hit list for ((paris AND NOT france) OR lear)

# Boolean queries

- The Boolean retrieval model can answer any query that is a Boolean expression.
  - Boolean queries are queries that use AND, OR and NOT to join query terms.
  - Views each document as a **set** of terms.
  - Is precise: Document matches condition or not.
- Primary commercial retrieval tool for 3 decades
- Many professional searchers (e.g., lawyers) still like Boolean queries.
  - You know exactly what you are getting.
- Many search systems you use are also Boolean: spotlight, email, intranet etc.

# Commercially successful Boolean retrieval: Westlaw

- Largest commercial legal search service in terms of the number of paying subscribers
- Over half a million subscribers performing millions of searches a day over tens of terabytes of text data
- The service was started in 1975.
- In 2005, Boolean search (called “Terms and Connectors” by Westlaw) was still the default, and used by a large percentage of users ...
- ... although ranked retrieval has been available since 1992.

# Westlaw: Example queries

*Information need:* Information on the legal theories involved in preventing the disclosure of trade secrets by employees formerly employed by a competing company

*Query:* “trade secret” /s disclos! /s prevent /s employe!

# Westlaw: Example queries

*Information need:* Requirements for disabled people to be able to access a workplace

*Query:* disab! /p access! /s work-site work-place (employment /3 place)

# Westlaw: Example queries

*Information need:* Cases about a host's responsibility for drunk guests

*Query:* host! /p (responsib! liab!) /p (intoxicat! drunk!) /p guest

# Westlaw: Comments

- Proximity operators:  $/3$  = within 3 words,  $/s$  = within a sentence,  $/p$  = within a paragraph
- Space is disjunction, not conjunction! (This was the default in search pre-Google.)
- Long, precise queries: incrementally developed, not like web search
- Why professional searchers often like Boolean search: precision, transparency, control
- When are Boolean queries the best way of searching? Depends on: information need, searcher, document collection, ...

# Outline

- 1 Administration
- 2 Introduction
- 3 Inverted index
- 4 Processing Boolean queries
- 5 Query optimization
- 6 Course overview

# Query optimization

- Consider a query that is an AND of  $n$  terms,  $n > 2$
- For each of the terms, get its postings list, then AND them together
- Example query: BRUTUS AND CALPURNIA AND CAESAR
- What is the best order for processing this query?

# Query optimization

- Example query: BRUTUS AND CALPURNIA AND CAESAR

# Query optimization

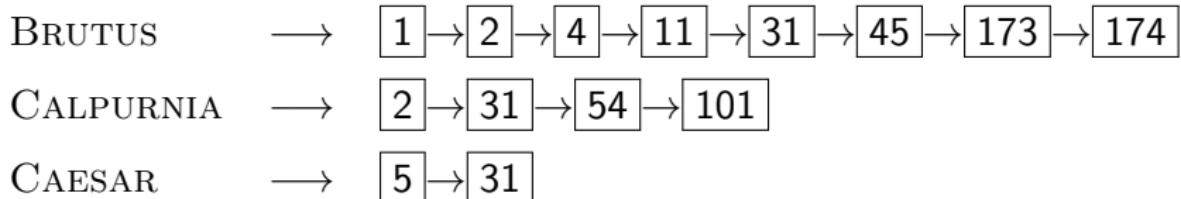
- Example query: BRUTUS AND CALPURNIA AND CAESAR
- Simple and effective optimization: Process in order of increasing frequency

# Query optimization

- Example query: BRUTUS AND CALPURNIA AND CAESAR
- Simple and effective optimization: Process in order of increasing frequency
- Start with the shortest postings list, then keep cutting further

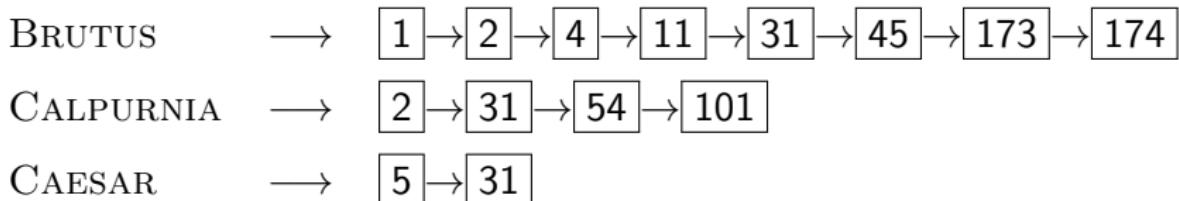
# Query optimization

- Example query: BRUTUS AND CALPURNIA AND CAESAR
- Simple and effective optimization: Process in order of increasing frequency
- Start with the shortest postings list, then keep cutting further



# Query optimization

- Example query: BRUTUS AND CALPURNIA AND CAESAR
- Simple and effective optimization: Process in order of increasing frequency
- Start with the shortest postings list, then keep cutting further
- In this example, first CAESAR, then CALPURNIA, then BRUTUS



# Optimized intersection algorithm for conjunctive queries

INTERSECT( $\langle t_1, \dots, t_n \rangle$ )

- 1  $terms \leftarrow \text{SORTBYINCREASINGFREQUENCY}(\langle t_1, \dots, t_n \rangle)$
- 2  $result \leftarrow \text{postings}(\text{first}(terms))$
- 3  $terms \leftarrow \text{rest}(terms)$
- 4 **while**  $terms \neq \text{NIL}$  and  $result \neq \text{NIL}$
- 5   **do**  $result \leftarrow \text{INTERSECT}(result, \text{postings}(\text{first}(terms)))$
- 6     $terms \leftarrow \text{rest}(terms)$
- 7 **return**  $result$

# More general optimization

- Example query: (MADDING OR CROWD) AND (IGNOBLE OR STRIFE)
- Get frequencies for all terms
- Estimate the size of each OR by the sum of its frequencies (conservative)
- Process in increasing order of OR sizes

# Outline

- 1 Administration
- 2 Introduction
- 3 Inverted index
- 4 Processing Boolean queries
- 5 Query optimization
- 6 Course overview

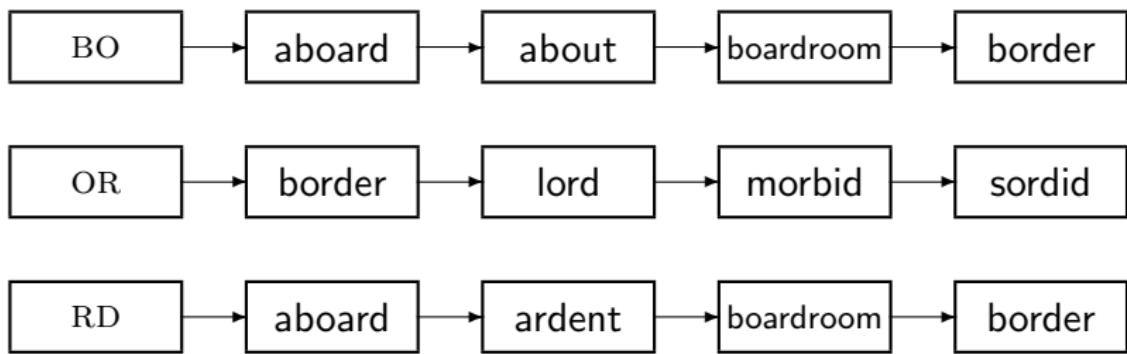
# Course overview

- We are done with Chapter 1 of IIR (IIR 01).
- Plan for the rest of the semester: 18–20 of the 21 chapters of IIR
- In what follows: teasers for most chapters – to give you a sense of what will be covered.

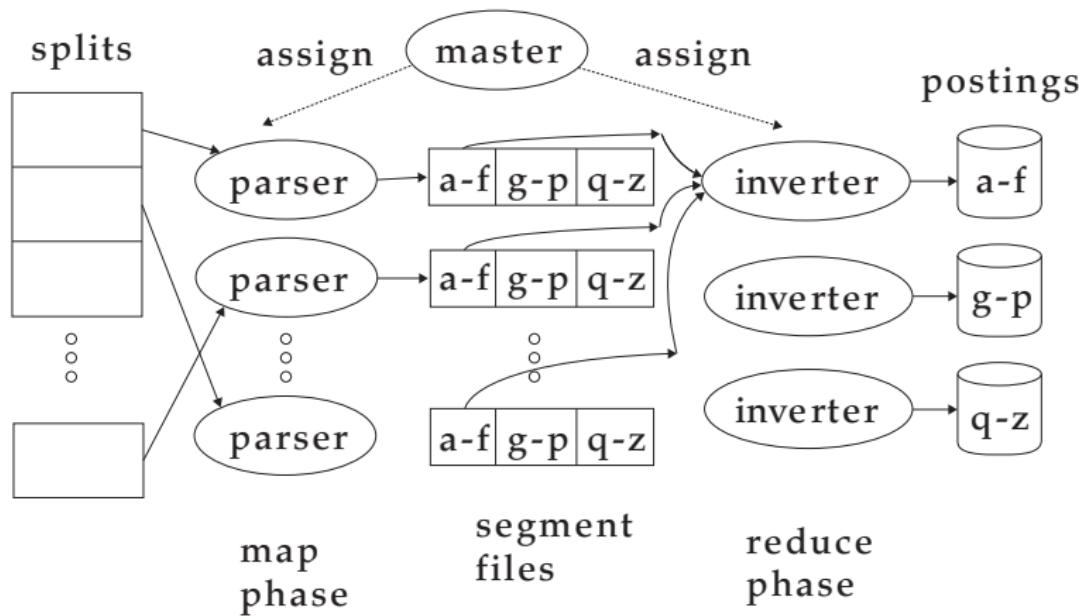
# IIR 02: The term vocabulary and postings lists

- Phrase queries: “STANFORD UNIVERSITY”
- Proximity queries: GATES NEAR MICROSOFT
- We need an index that captures **position information** for phrase queries and proximity queries.

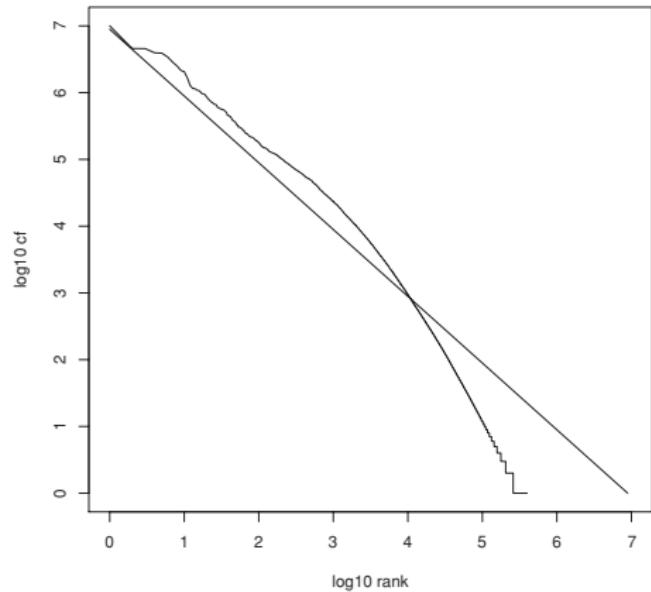
# IIR 03: Dictionaries and tolerant retrieval



# IIR 04: Index construction



# IIR 05: Index compression

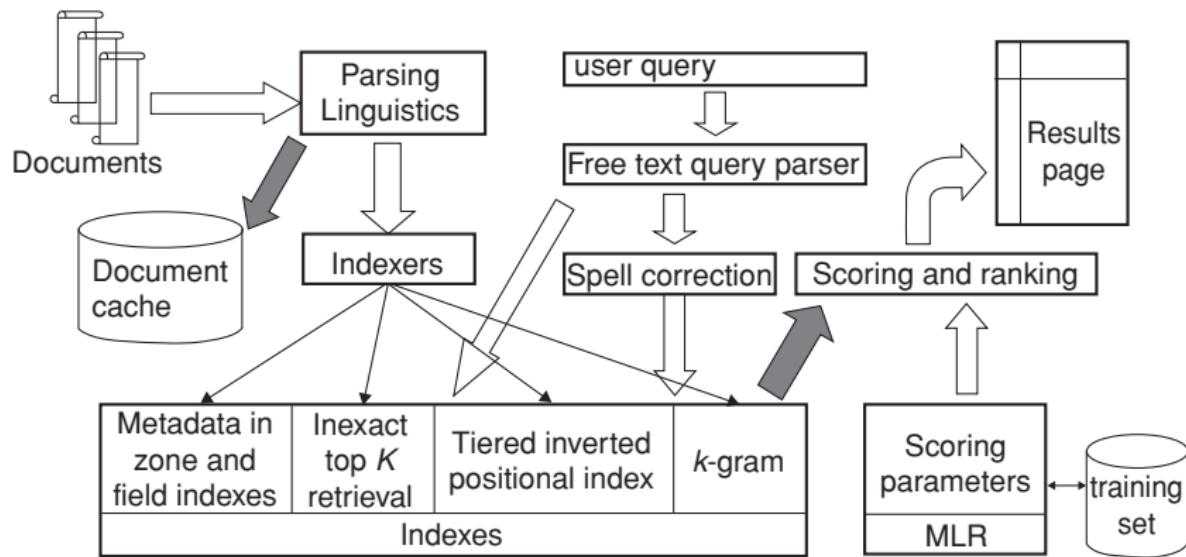


Zipf's law

# IIR 06: Scoring, term weighting and the vector space model

- Ranking search results
  - Boolean queries only give inclusion or exclusion of documents.
  - For ranked retrieval, we measure the proximity between the query and each document.
  - One formalism for doing this: [the vector space model](#)
- Key challenge in ranked retrieval: evidence accumulation for a term in a document
  - 1 vs. 0 occurrence of a query term in the document
  - 3 vs. 2 occurrences of a query term in the document
  - Usually: more is better
  - But by how much?
  - Need a scoring function that translates frequency into score or weight

# IIR 07: Scoring in a complete search system



# IIR 08: Evaluation and dynamic summaries



Web [Show options...](#)

Results 1 - 10

## [Manitoba - Wikipedia, the free encyclopedia](#)

Manitoba's capital and **largest city**, Winnipeg, .... According to Environment Canada, Manitoba ranked first for clearest skies year round, and ranked **second** ...

[Geography](#) - [History](#) - [Demographics](#) - [Economy](#)

[en.wikipedia.org/wiki/Manitoba](http://en.wikipedia.org/wiki/Manitoba) - [Cached](#) - [Similar](#)

## [List of cities in Canada - Wikipedia, the free encyclopedia](#)

Cities and towns in **Manitoba**. See also: List of communities in **Manitoba** .... Dartmouth - formerly the **second largest city** in Nova Scotia, now a Metropolitan ...  
[en.wikipedia.org/wiki/List\\_of\\_cities\\_in\\_Canada](http://en.wikipedia.org/wiki/List_of_cities_in_Canada) - [Cached](#) - [Similar](#)

[Show more results from en.wikipedia.org](#)

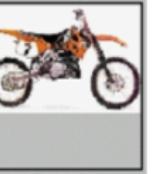
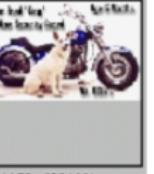
## [Canadian Immigration Information - Manitoba](#)

The **largest city** in the province is the capital, Winnipeg, with a population exceeding 706900. The **second largest city** is Brandon. **Manitoba** has received ...  
[www.canadavisa.com/about-manitoba.html](http://www.canadavisa.com/about-manitoba.html) - [Cached](#) - [Similar](#)

## [CBC Manitoba | EAL](#)

Lesson 57: Brandon - **Manitoba's Second Largest City**. For Teachers; For Students. Step One Open the Lesson: PDF (194kb) WORD (238kb) Microsoft Word ...  
[www.cbc.ca/manitoba/.../lesson-57-brandon---manitobas-second-largest.html](http://www.cbc.ca/manitoba/.../lesson-57-brandon---manitobas-second-largest.html) - [Cached](#)

# IIR 09: Relevance feedback & query expansion

						Browse	Search	Prev	Next	Random	
						(144538, 523493) 0.54182 0.231944 0.309876	(144538, 523835) 0.56319296 0.267304 0.295809	(144538, 523529) 0.584279 0.280881 0.303398	(144536, 253569) 0.64301 0.351395 0.293615	(144536, 253568) 0.650275 0.411745 0.23853	(144538, 523799) 0.66709197 0.358033 0.309039
						(144473, 16249) 0.6721 0.393922 0.278178	(144456, 249634) 0.675018 0.4639 0.211118	(144456, 253693) 0.676901 0.47645 0.200451	(144473, 16328) 0.700339 0.309002 0.391337	(144483, 265264) 0.70170796 0.36176 0.339948	(144478, 512410) 0.70297 0.469111 0.233859

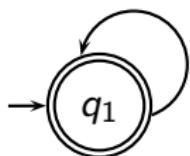
# IIR 11: Probabilistic information retrieval

$P(R|d, q)$  is modeled using term incidence vectors as  $P(R|\vec{x}, \vec{q})$

$$\begin{aligned} P(R = 1|\vec{x}, \vec{q}) &= \frac{P(\vec{x}|R = 1, \vec{q})P(R = 1|\vec{q})}{P(\vec{x}|\vec{q})} \\ P(R = 0|\vec{x}, \vec{q}) &= \frac{P(\vec{x}|R = 0, \vec{q})P(R = 0|\vec{q})}{P(\vec{x}|\vec{q})} \end{aligned}$$

- $P(\vec{x}|R = 1, \vec{q})$  and  $P(\vec{x}|R = 0, \vec{q})$ : probability that if a relevant or nonrelevant document is retrieved, then that document's representation is  $\vec{x}$
- Use statistics about the document collection to estimate these probabilities

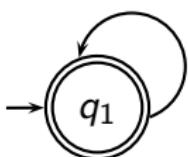
## IIR 12: Language models



$w$	$P(w q_1)$	$w$	$P(w q_1)$
STOP	0.2	toad	0.01
the	0.2	said	0.03
a	0.1	likes	0.02
frog	0.01	that	0.04
	...		...

This is a one-state probabilistic finite-state automaton – [a unigram language model](#) – and the state emission distribution for its one state  $q_1$ .

## IIR 12: Language models

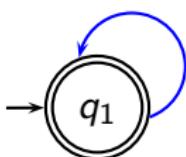


$w$	$P(w q_1)$	$w$	$P(w q_1)$
STOP	0.2	toad	0.01
the	0.2	said	0.03
a	0.1	likes	0.02
frog	0.01	that	0.04
	...		...

This is a one-state probabilistic finite-state automaton – [a unigram language model](#) – and the state emission distribution for its one state  $q_1$ .

STOP is not a word, but a special symbol indicating that the automaton stops.

## IIR 12: Language models



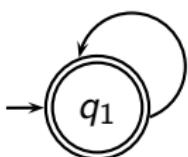
$w$	$P(w q_1)$	$w$	$P(w q_1)$
STOP	0.2	toad	0.01
the	0.2	said	0.03
a	0.1	likes	0.02
frog	0.01	that	0.04
	...		...

This is a one-state probabilistic finite-state automaton – [a unigram language model](#) – and the state emission distribution for its one state  $q_1$ .

STOP is not a word, but a special symbol indicating that the automaton stops.

frog

## IIR 12: Language models



$w$	$P(w q_1)$	$w$	$P(w q_1)$
STOP	0.2	toad	0.01
the	0.2	said	0.03
a	0.1	likes	0.02
frog	0.01	that	0.04
	...		...

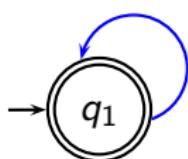
This is a one-state probabilistic finite-state automaton – [a unigram language model](#) – and the state emission distribution for its one state  $q_1$ .

STOP is not a word, but a special symbol indicating that the automaton stops.

frog

$$P(\text{string}) = 0.01$$

## IIR 12: Language models



w	$P(w q_1)$	w	$P(w q_1)$
STOP	0.2	toad	0.01
the	0.2	said	0.03
a	0.1	likes	0.02
frog	0.01	that	0.04
	...		...

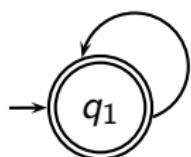
This is a one-state probabilistic finite-state automaton – [a unigram language model](#) – and the state emission distribution for its one state  $q_1$ .

STOP is not a word, but a special symbol indicating that the automaton stops.

frog said

$$P(\text{string}) = 0.01$$

## IIR 12: Language models



w	$P(w q_1)$	w	$P(w q_1)$
STOP	0.2	toad	0.01
the	0.2	said	0.03
a	0.1	likes	0.02
frog	0.01	that	0.04
	...		...

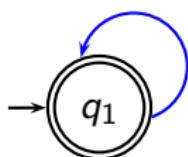
This is a one-state probabilistic finite-state automaton – [a unigram language model](#) – and the state emission distribution for its one state  $q_1$ .

STOP is not a word, but a special symbol indicating that the automaton stops.

frog said

$$P(\text{string}) = 0.01 \cdot 0.03$$

## IIR 12: Language models



w	$P(w q_1)$	w	$P(w q_1)$
STOP	0.2	toad	0.01
the	0.2	said	0.03
a	0.1	likes	0.02
frog	0.01	that	0.04
	...		...

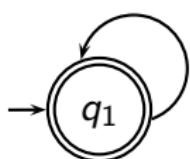
This is a one-state probabilistic finite-state automaton – [a unigram language model](#) – and the state emission distribution for its one state  $q_1$ .

STOP is not a word, but a special symbol indicating that the automaton stops.

frog said that

$$P(\text{string}) = 0.01 \cdot 0.03$$

## IIR 12: Language models



w	$P(w q_1)$	w	$P(w q_1)$
STOP	0.2	toad	0.01
the	0.2	said	0.03
a	0.1	likes	0.02
frog	0.01	that	0.04
	...		...

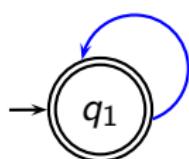
This is a one-state probabilistic finite-state automaton – [a unigram language model](#) – and the state emission distribution for its one state  $q_1$ .

STOP is not a word, but a special symbol indicating that the automaton stops.

frog said that

$$P(\text{string}) = 0.01 \cdot 0.03 \cdot 0.04$$

## IIR 12: Language models



$w$	$P(w q_1)$	$w$	$P(w q_1)$
STOP	0.2	toad	0.01
the	0.2	said	0.03
a	0.1	likes	0.02
frog	0.01	that	0.04
		...	...

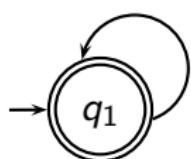
This is a one-state probabilistic finite-state automaton – [a unigram language model](#) – and the state emission distribution for its one state  $q_1$ .

STOP is not a word, but a special symbol indicating that the automaton stops.

frog said that toad

$$P(\text{string}) = 0.01 \cdot 0.03 \cdot 0.04$$

## IIR 12: Language models



w	$P(w q_1)$	w	$P(w q_1)$
STOP	0.2	toad	0.01
the	0.2	said	0.03
a	0.1	likes	0.02
frog	0.01	that	0.04
	...		...

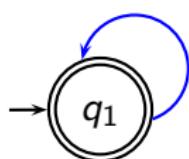
This is a one-state probabilistic finite-state automaton – [a unigram language model](#) – and the state emission distribution for its one state  $q_1$ .

STOP is not a word, but a special symbol indicating that the automaton stops.

frog said that toad

$$P(\text{string}) = 0.01 \cdot 0.03 \cdot 0.04 \cdot 0.01$$

## IIR 12: Language models



$w$	$P(w q_1)$	$w$	$P(w q_1)$
STOP	0.2	toad	0.01
the	0.2	said	0.03
a	0.1	likes	0.02
frog	0.01	that	0.04
		...	...

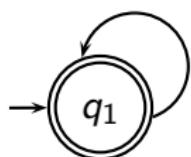
This is a one-state probabilistic finite-state automaton – [a unigram language model](#) – and the state emission distribution for its one state  $q_1$ .

STOP is not a word, but a special symbol indicating that the automaton stops.

frog said that toad likes

$$P(\text{string}) = 0.01 \cdot 0.03 \cdot 0.04 \cdot 0.01$$

## IIR 12: Language models



w	$P(w q_1)$	w	$P(w q_1)$
STOP	0.2	toad	0.01
the	0.2	said	0.03
a	0.1	likes	0.02
frog	0.01	that	0.04
	...		...

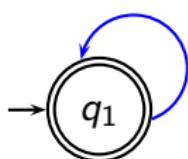
This is a one-state probabilistic finite-state automaton – [a unigram language model](#) – and the state emission distribution for its one state  $q_1$ .

STOP is not a word, but a special symbol indicating that the automaton stops.

frog said that toad likes

$$P(\text{string}) = 0.01 \cdot 0.03 \cdot 0.04 \cdot 0.01 \cdot 0.02$$

## IIR 12: Language models



w	$P(w q_1)$	w	$P(w q_1)$
STOP	0.2	toad	0.01
the	0.2	said	0.03
a	0.1	likes	0.02
frog	0.01	that	0.04
	...		...

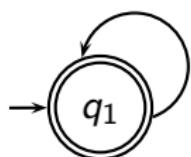
This is a one-state probabilistic finite-state automaton – [a unigram language model](#) – and the state emission distribution for its one state  $q_1$ .

STOP is not a word, but a special symbol indicating that the automaton stops.

frog said that toad likes frog

$$P(\text{string}) = 0.01 \cdot 0.03 \cdot 0.04 \cdot 0.01 \cdot 0.02$$

## IIR 12: Language models



$w$	$P(w q_1)$	$w$	$P(w q_1)$
STOP	0.2	toad	0.01
the	0.2	said	0.03
a	0.1	likes	0.02
frog	0.01	that	0.04
		...	...

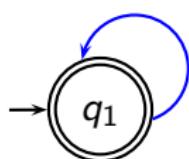
This is a one-state probabilistic finite-state automaton – [a unigram language model](#) – and the state emission distribution for its one state  $q_1$ .

STOP is not a word, but a special symbol indicating that the automaton stops.

frog said that toad likes frog

$$P(\text{string}) = 0.01 \cdot 0.03 \cdot 0.04 \cdot 0.01 \cdot 0.02 \cdot 0.01$$

## IIR 12: Language models



$w$	$P(w q_1)$	$w$	$P(w q_1)$
STOP	0.2	toad	0.01
the	0.2	said	0.03
a	0.1	likes	0.02
frog	0.01	that	0.04
	...		...

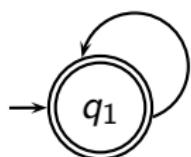
This is a one-state probabilistic finite-state automaton – [a unigram language model](#) – and the state emission distribution for its one state  $q_1$ .

STOP is not a word, but a special symbol indicating that the automaton stops.

frog said that toad likes frog STOP

$$P(\text{string}) = 0.01 \cdot 0.03 \cdot 0.04 \cdot 0.01 \cdot 0.02 \cdot 0.01$$

## IIR 12: Language models



w	$P(w q_1)$	w	$P(w q_1)$
STOP	0.2	toad	0.01
the	0.2	said	0.03
a	0.1	likes	0.02
frog	0.01	that	0.04
	...		...

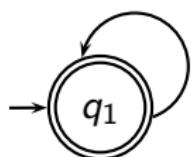
This is a one-state probabilistic finite-state automaton – [a unigram language model](#) – and the state emission distribution for its one state  $q_1$ .

STOP is not a word, but a special symbol indicating that the automaton stops.

frog said that toad likes frog STOP

$$P(\text{string}) = 0.01 \cdot 0.03 \cdot 0.04 \cdot 0.01 \cdot 0.02 \cdot 0.01 \cdot 0.2$$

## IIR 12: Language models



$w$	$P(w q_1)$	$w$	$P(w q_1)$
STOP	0.2	toad	0.01
the	0.2	said	0.03
a	0.1	likes	0.02
frog	0.01	that	0.04
		...	...

This is a one-state probabilistic finite-state automaton – [a unigram language model](#) – and the state emission distribution for its one state  $q_1$ .

STOP is not a word, but a special symbol indicating that the automaton stops.

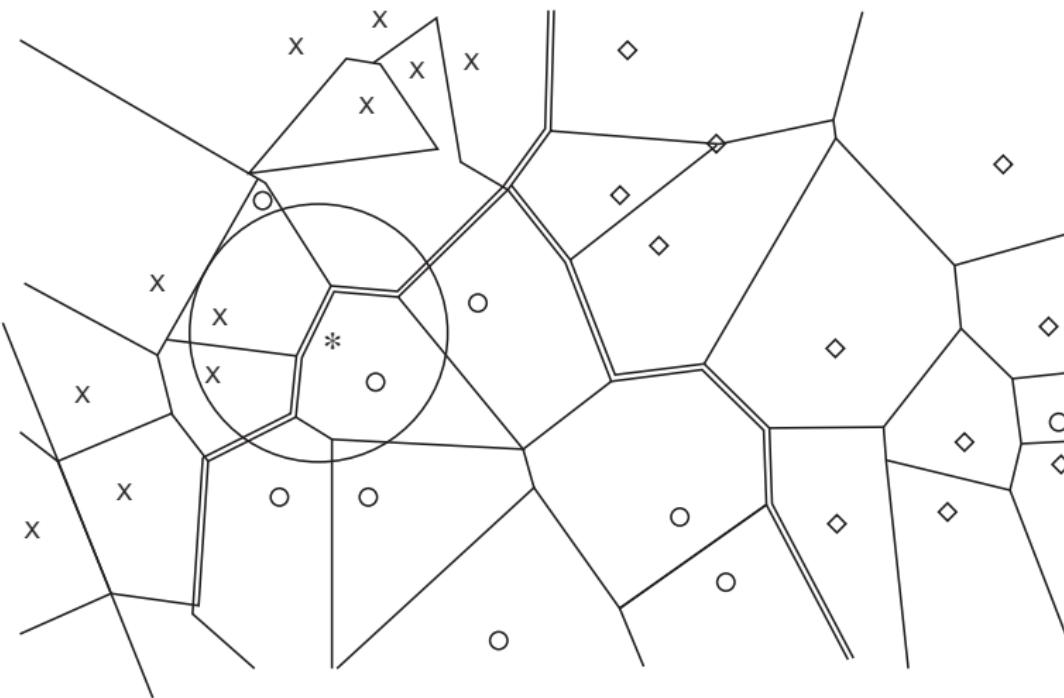
frog said that toad likes frog STOP

$$\begin{aligned}
 P(\text{string}) &= 0.01 \cdot 0.03 \cdot 0.04 \cdot 0.01 \cdot 0.02 \cdot 0.01 \cdot 0.2 \\
 &= 0.000000000048
 \end{aligned}$$

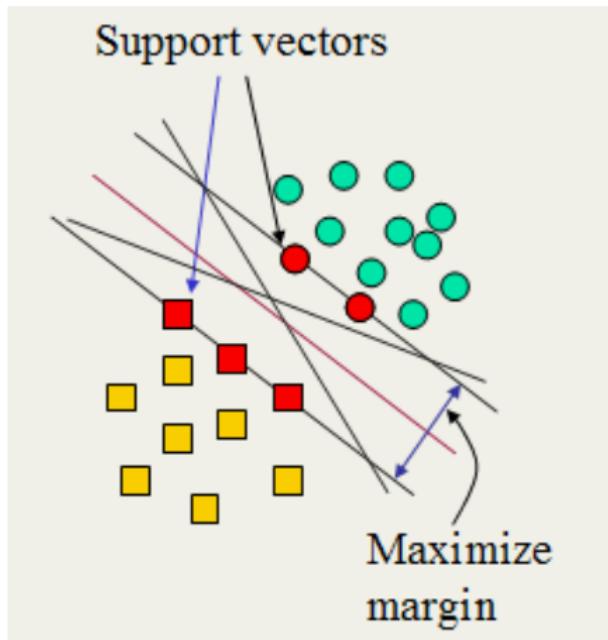
# IIR 13: Text classification & Naive Bayes

- Text classification = assigning documents automatically to predefined classes
- Examples:
  - Language (English vs. French)
  - Adult content
  - Region

# IIR 14: Vector classification



# IIR 15: Support vector machines (possibly skipped)



# IIR 16: Flat clustering

Vivísimo®  the Web  Advanced Search Help

**Clustered Results** Top 208 results of at least 20,373,974 retrieved for the query **jaguar** ([Details](#))

- ▶ [jaguar \(208\)](#)
  - ▶ [Cars \(74\)](#)
  - ▶ [Club \(34\)](#)
  - ▶ [Cat \(23\)](#)
  - ▶ [Animal \(13\)](#)
  - ▶ [Restoration \(10\)](#)
  - ▶ [Mac OS X \(8\)](#)
  - ▶ [Jaguar Model \(8\)](#)
  - ▶ [Request \(5\)](#)
  - ▶ [Mark Webber \(6\)](#)
  - ▶ [Maya \(5\)](#)
- More

Find in clusters:

1. [Jag-lovers - THE source for all Jaguar information](#) [new window] [frame] [cache] [preview] [clusters]  
... Internet! Serving Enthusiasts since 1993 The Jag-lovers Web Currently with 40661 members The Premier Jaguar Cars web resource for all enthusiasts Lists and Forums Jag-lovers originally evolved around its ...  
[www.jag-lovers.org](http://www.jag-lovers.org) - Open Directory 2, Wisenut 8, Ask Jeeves 8, MSN 9, Looksmart 12, MSN Search 18
2. [Jaguar Cars](#) [new window] [frame] [cache] [preview] [clusters]  
[...] redirected to [www.jaguar.com](http://www.jaguar.com)  
[www.jaguarcars.com](http://www.jaguarcars.com) - Looksmart 1, MSN 2, Lycos 3, Wisenut 6, MSN Search 9, MSN 29
3. <http://www.jaguar.com/> [new window] [frame] [preview] [clusters]  
[www.jaguar.com](http://www.jaguar.com) - MSN 1, Ask Jeeves 1, MSN Search 3, Lycos 9
4. [Apple - Mac OS X](#) [new window] [frame] [preview] [clusters]  
Learn about the new OS X Server, designed for the Internet, digital media and workgroup management.  
Download a technical factsheet.  
[www.apple.com/macosx](http://www.apple.com/macosx) - Wisenut 1, MSN 3, Looksmart 26

# IIR 17: Hierarchical clustering

<http://news.google.com>



## [Barack Obama, Vladimir Putin hold first in-person talk since start of Ukraine ...](#)

Canada.com - 6 hours ago

A video screen shows U.S. President Barack Obama and Russian President Vladimir Putin during the Oustrehe: commemorate the 70th anniversary of the Allied invasion at Normandy, in Oustrehe, France, June ...

Canada.com

## [Obama and Vladimir Putin dine separately with French President Francois ...](#)

New York Daily News - 10 hours ago

French President Francois Hollande engaged in some deft dinnertime diplomacy Thursday night - hosting separate meals so P: Vladimir Putin wouldn't have to break bread together. Hollande and Obama had dinner at ...



## [Topless feminist stabs wax Putin in France](#)

The Local.fr - Jun 5, 2014

The same day President Vladimir Putin was to arrive in France for D-Day anniversary events, radical feminist p: leader's statue in a Paris wax museum.

The Local.fr

## [Vladimir Putin meets Ukraine president-elect Petro Poroshenko at D-Day ...](#)

Telegraph.co.uk - 9 hours ago

Russian President Vladimir Putin and Ukraine's president-elect Petro Poroshenko discussed a ceasefire and other possible site countries in a brief but potentially significant meeting in France on Friday, French ...



## [Vote Possible to Get Stalingrad Name Back: Vladimir Putin](#)

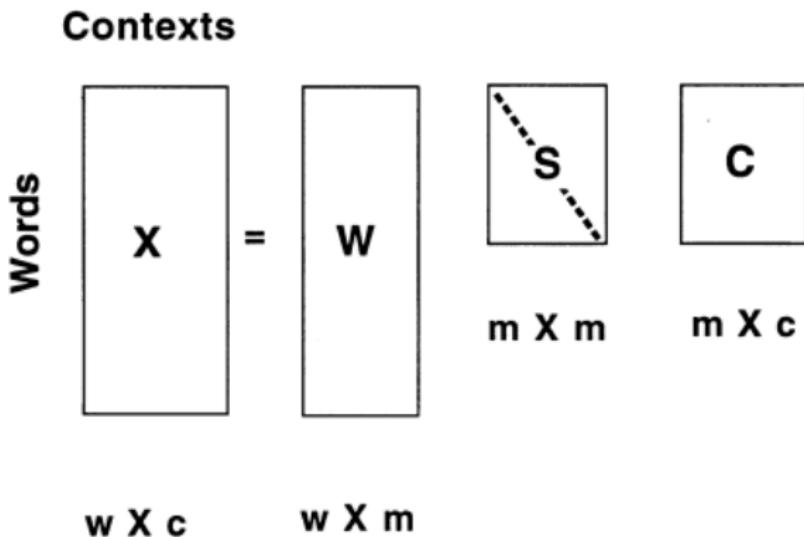
NDTV - 2 hours ago

Russian President Vladimir Putin speaks to the media at Benouville castle, Friday, June 6, 2014, where he arriv landings.

NDTV

# IIR 18: Latent Semantic Indexing

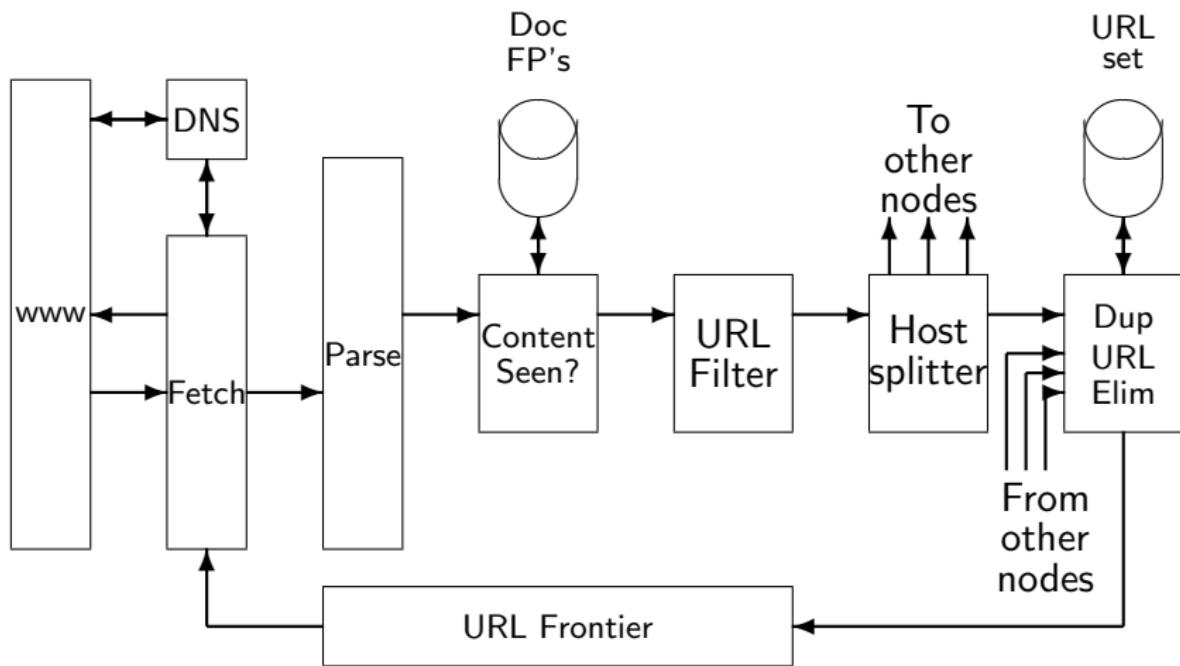
I



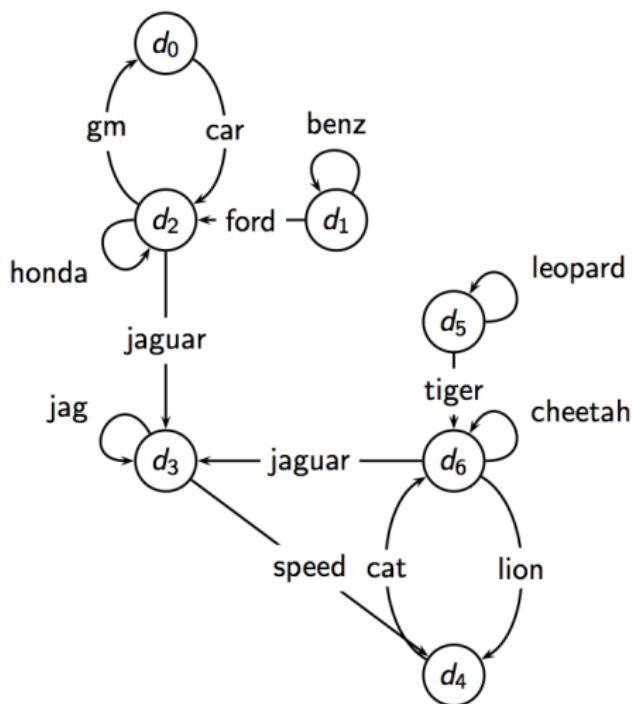
# IIR 19: The web and its challenges

- Unusual and diverse documents
- Unusual and diverse users and information needs
- Beyond terms and text: exploit link analysis, user data
- How do web search engines work?
- How can we make them better?

# IIR 20: Crawling



# IIR 21: Link analysis / PageRank

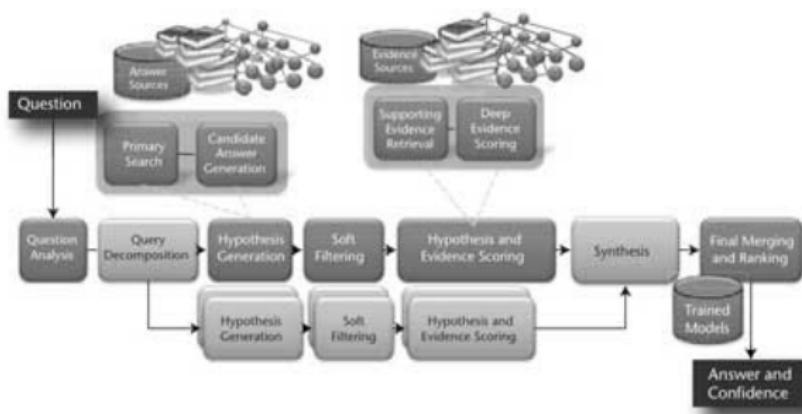


# If time permits: Evolution of the Google IR System



If time permits:

## Building Watson: An Overview of the DeepQA Project



# If time permits: Introduction to computational advertising

many benefits photographers, from beginners to professionals, will ...

## News for canon camera



### Canon launches photo saving and sharing platform in the ...

New York Daily News - 5 hours ago

Called Irstra, Canon's new cloud system launches this week in Europe ... However, if a picture has been snapped on a Canon camera, its EXIF ...

## More news for canon camera

## Amazon.com: Canon

[www.amazon.com/Canon-Camera-Photo/b?ie=UTF8...](http://www.amazon.com/Canon-Camera-Photo/b?ie=UTF8...) Amazon.com ~  
Canon EOS 6D 20.2 MP CMOS Digital SLR Camera with 3.0-Inch ... Canon. \$1,899.00  
\$1,899.00. Canon EOS. 3. Canon EOS Rebel T3 12.2 MP CMOS Digital ...

## Images for canon camera



Report images

## More images for canon camera

## Canon Camera Reviews - Canon Cameras

[www.Imaging-resource.com/MFR1.HTM?View=Canon...](http://www.Imaging-resource.com/MFR1.HTM?View=Canon...) Imaging Resource ~  
One of the old-line global leaders in the photo industry, Canon cameras cover the range from entry-level point & shoot models to high-end professional SLRs at ...

Ads

## Canon Camera

[www.bestbuy.com/](http://www.bestbuy.com/)

4.5 ★★★★★ rating for bestbuy.com  
Full Selection of Cameras Plus  
Free Shipping, Shop Now & Save!  
9 575 E Wetmore Rd, Tucson, AZ  
(520) 696-3442

## Canon Cameras at Sears®

[www.sears.com/Canon\\_Cameras](http://www.sears.com/Canon_Cameras)

4.4 ★★★★★ rating for sears.com  
Visit Sears® for Great Values and a  
Big Selection of Canon Cameras!  
9 4570 N Oracle Rd, Tucson, AZ  
(520) 690-2090

## Canon CAMERA

[www.bhphotovideo.com/FreeShip](http://www.bhphotovideo.com/FreeShip)

4.9 ★★★★★ advertiser rating  
Save on Professional Camcorders  
EOS-1D C Camera (Body Only)

## Canon Camera

[www.hsn.com/](http://www.hsn.com/)

4.4 ★★★★★ rating for hsn.com  
Save \$150+ Today on Canon® Cameras  
& Accessories. Free Shipping at HSN

## Câmera Canon at Amazon

[www.amazon.com/Cameras](http://www.amazon.com/Cameras)

Save on Câmera canon  
Free Shipping Available with Amazon

## 2014 Best Canon Camera

# Take-away

- Why you should take this course
- Admin issues
- Boolean Retrieval: Design and data structures of a simple information retrieval system
- What topics will be covered in this class?