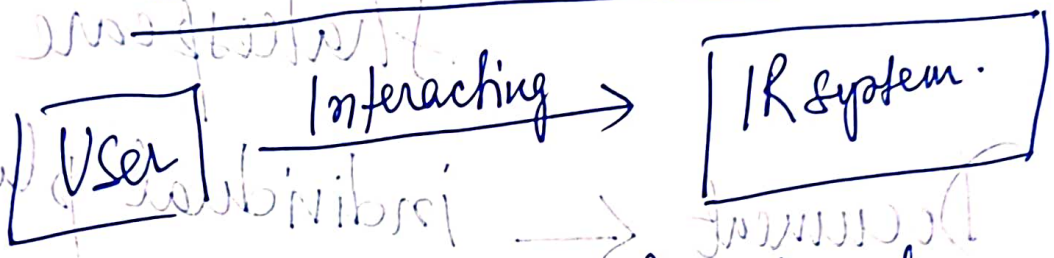
Boolean Retrieval



Document — unit of retrieval.

Doc ID — each document is identified by a Doc ID.

Corpus — collection of documents

User — have an information need.

↓

query → feed → IR system

Term — unit of information (word / phrase).

IR system ← tasked to retrieving relevant documents.

Corpus ← All plays written Shakespeare.

Document ← individual play.

Query ← a Boolean expression of terms connected by Boolean operators: AND, OR, NOT etc.

Query: Which all plays of Shakespeare contain the words Brutus AND Caesar but NOT Calpurnia.

Shell grep? ① grep all shakespeare's plays containing Brutus & Caesar.
② Go line by line & omit those that have Calpurnia.

Why does it not work this way?

① Very slow.

② NOT Calpurnia is skipping dorn things.

③ Romans (near) Countrymen.
within three words.

④ Ranked retrieval.

The term-incidence matrix.

Terms	Antonio & Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antonio	1	1	0	0	0	1
Brutus	1	1	0	1	0	0
Caesar	1	1	0	1	1	0
Cleopatra	1	0	0	0	0	0
Calpurnia	0	1	0	0	0	0
mercy	1	0	1	1	1	1
worser.	1	0	1	1	1	0

Documents

BRUTUS AND CAESAR NOT CALPURNIA.

[110100] & [110110] ∧ ¬[010000].

response: 110100.

- Antonio & Cleopatra > Act III, Scene ii

- Hamlet > Act III, Scene ii

Reductionist

Holist ("nurse is different").

What is the problem?

Bigger collections

N = 1 M documents

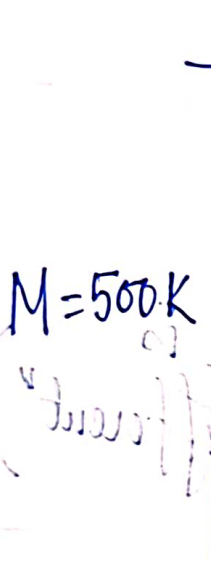
Each doc ← on avg → 1000 words

⇒ 6 bytes / word on avg

How much data on documents? 6GB.

M = 500K distinct terms across all documents

$N = 10^6$



$\frac{1}{2}$ trillion entries

Engineer's  sparse matrix.

Every column will have only 1000 s.

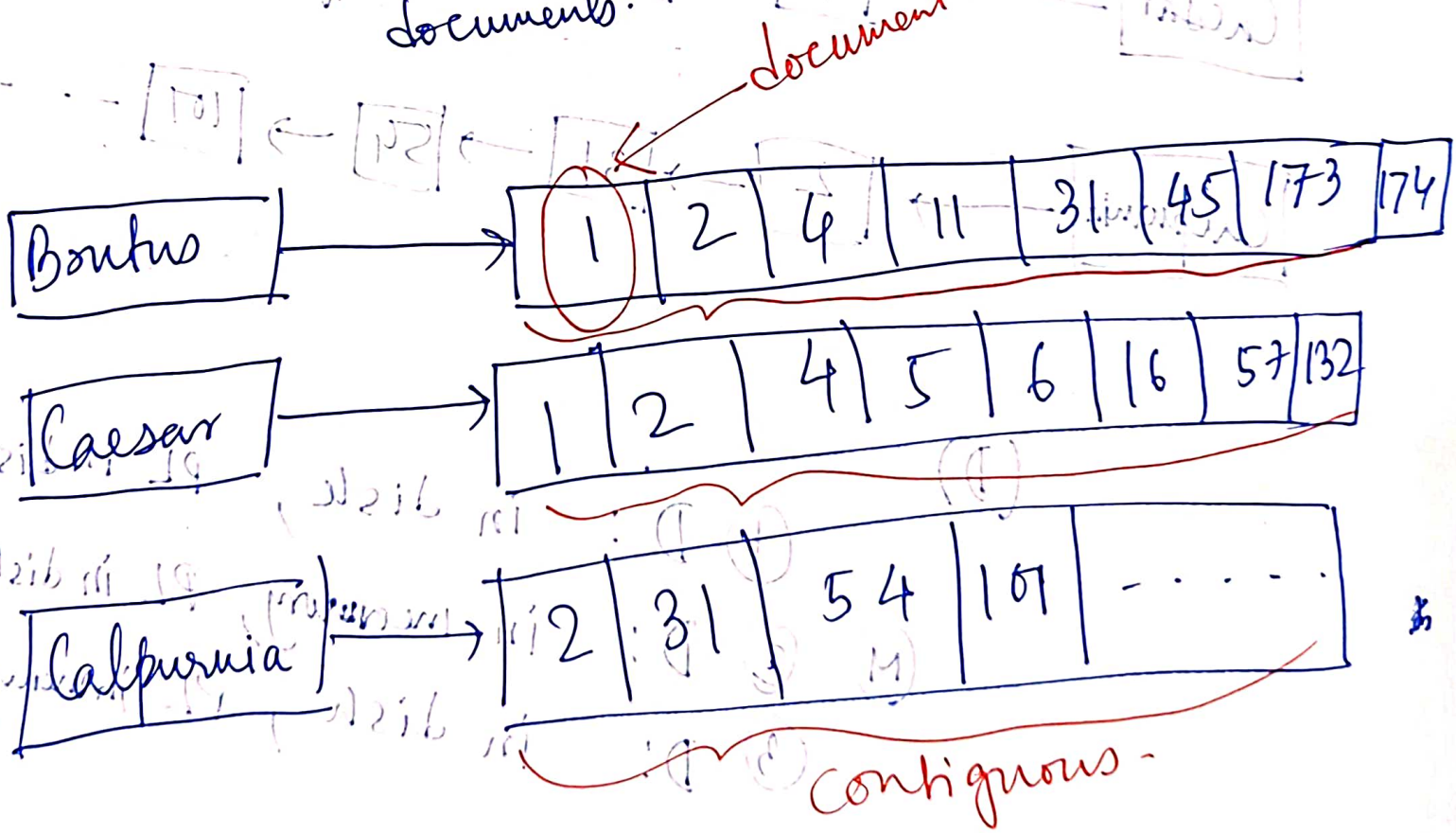
Rest every thing is Zero.

How many entries have info? 1 b.

Inverted Index.

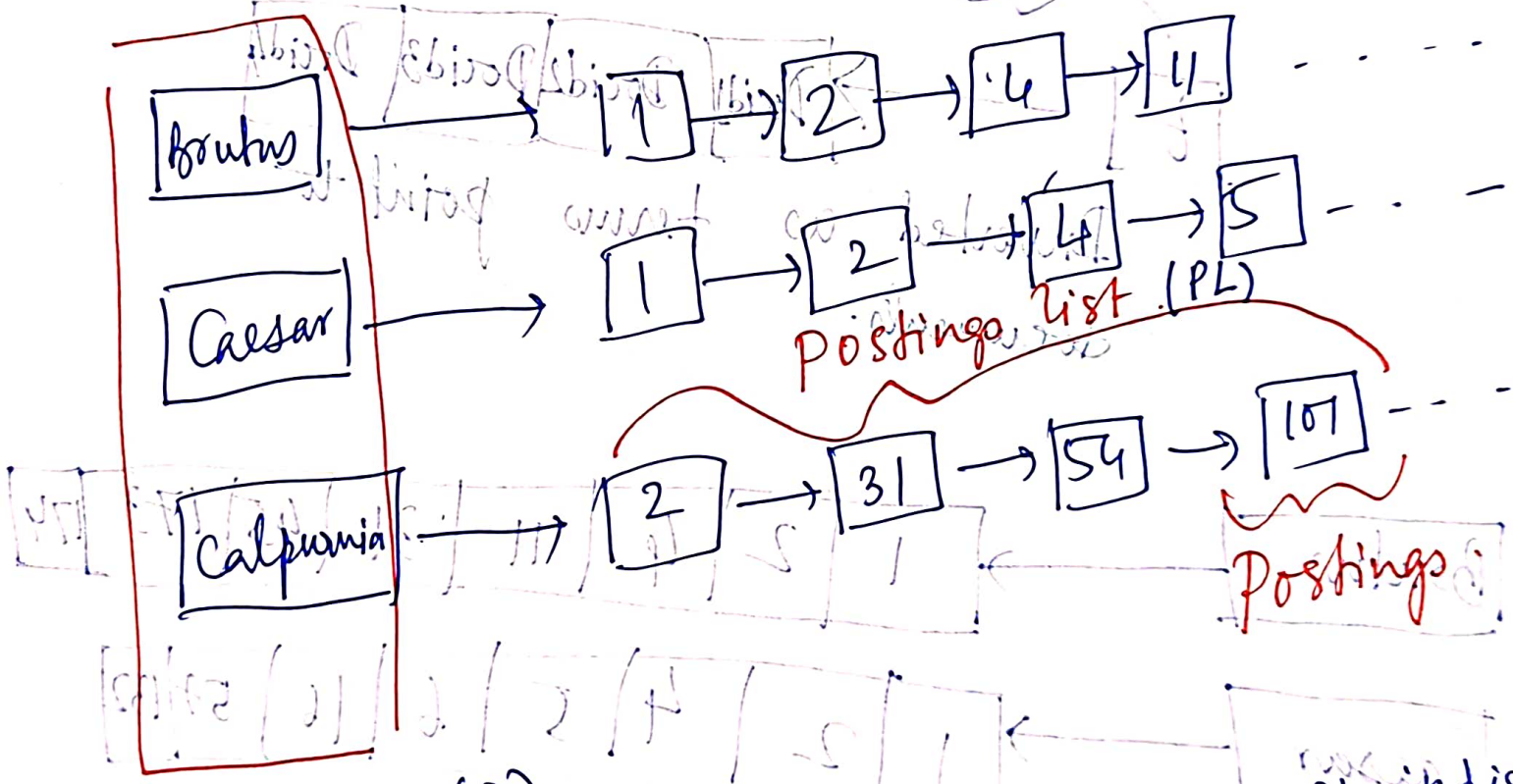
For each term t we store a list of all documents that contain t

Inverted as terms point to documents.



To keep it
Contiguous or not
(array)

Arrival of new docs?



(M)
① D: in disk, PL in disk
② D: in memory, PL in disk
③ D: in disk, PL in memory

Raw text documents

↓

Inverted index.

.... Friends, Romans, Countrymen ...

Tokenizer: split the individual tokens.

Friends | Romans | Countrymen

↓

Language models (Noralizer)

friend | roman | countrymen

Stems of a word.

↓ stemming.

↓

Indexer → Inverted index.

Tokenization

state * of * the * art.

Normalization:

U.S.A ≡ USA.

Stemming.

authorize, authorization

Should map to a single term.

Stop words / function words.

the, a, to, of

Linguistically ↓ no semantics → render syntax & semantics to other words.

→ very frequent.

Indexer.

Term	—	docID.
I	—	1
did	—	1
the	—	1
⋮		
Caesar	—	2
was	—	2
ambitious	—	2

Two times sorting

① Lexicographically on the terms.

② by the docID.

↳ ambitious — 2

Fast time sorting

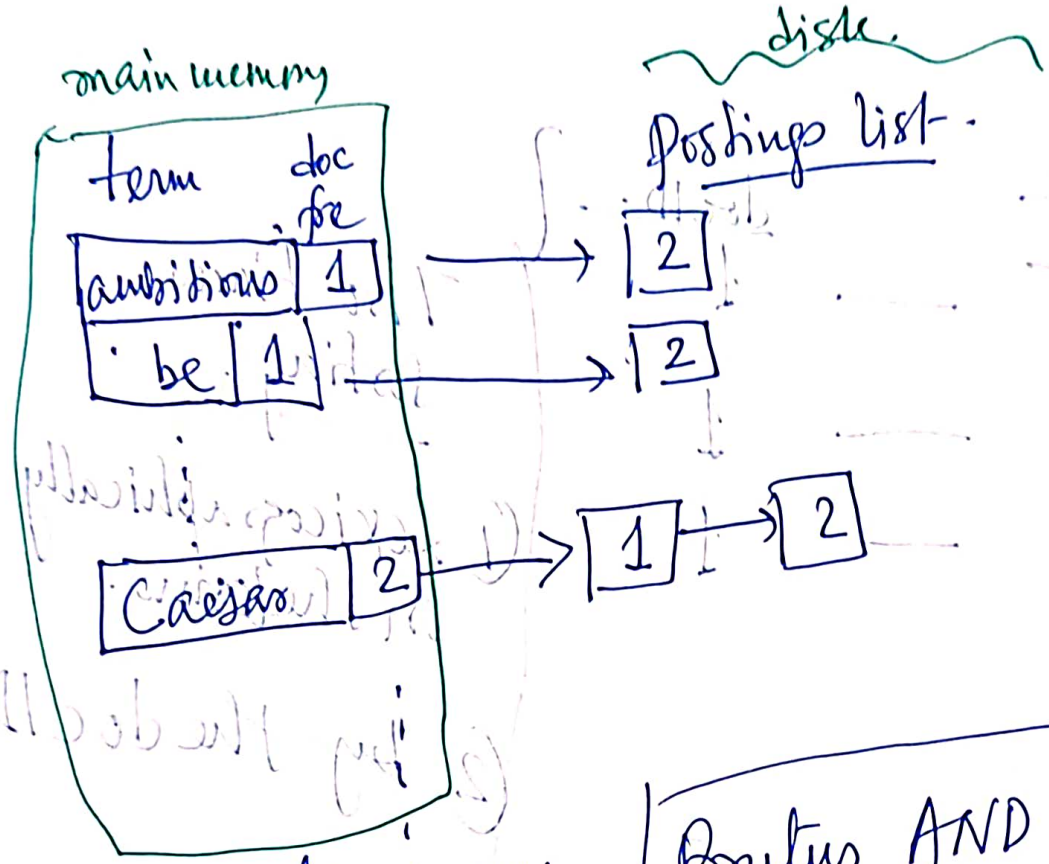
⋮		
Caesar	—	1
Caesar	—	2
⋮		
I	—	1
⋮		

2nd time

ambitious	—	2
be	—	1
be	—	2
⋮		
Caesar	—	1
Caesar	—	2
Caesar	—	2

=2

document frequency of the term.



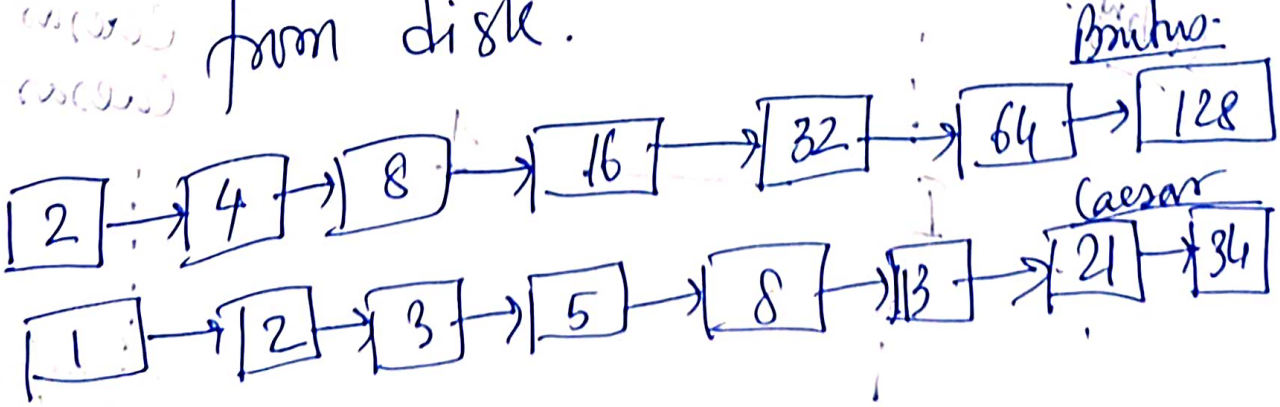
Execute query | Brutus AND Caesar.

Step 1: Locate Brutus in dictionary

2: Retrieve postings list of Brutus from disk.

3: Locate Caesar in dictionary

4: Retrieve postings list of Caesar from disk.



Merge Algorithm (Recall mergesort)

List 1 → |x|

List 2 → |y|

Complexity:

$O(X+Y)$ ⟶ Hence we kept the lists sorted.

INTERSECT (P_1, P_2)

1. answer ← <>
2. while $P_1 \neq$ NIL & $P_2 \neq$ NIL
3. do if docID(P_1) = docID(P_2)
4. then ADD(answer, docID(P_1))
5. $P_1 \leftarrow$ next(P_1)
6. $P_2 \leftarrow$ next(P_2)
7. else if docID(P_1) < docID(P_2)
8. then $P_1 \leftarrow$ next(P_1)
9. else $P_2 \leftarrow$ next(P_2)
10. return answer.

Any document as a set of terms.

→ ~~Ranking~~

One of the most popular retrieval system — last 3 decades.

Mac OS X spotlight → still uses this

Westlaw: http://www.westlaw.com/

legal experts.

700K users.

started → 1975.

ranking → 1992

federated search / parallel search) → 2010

Query:

What is the statute of limitations in cases involving the federal tort claims act?

LIMIT! /3 STATUTE ACTION /S FEDERAL /2 TORT

within 3 words.

same sentence.

2 words

/3 CLAIM.

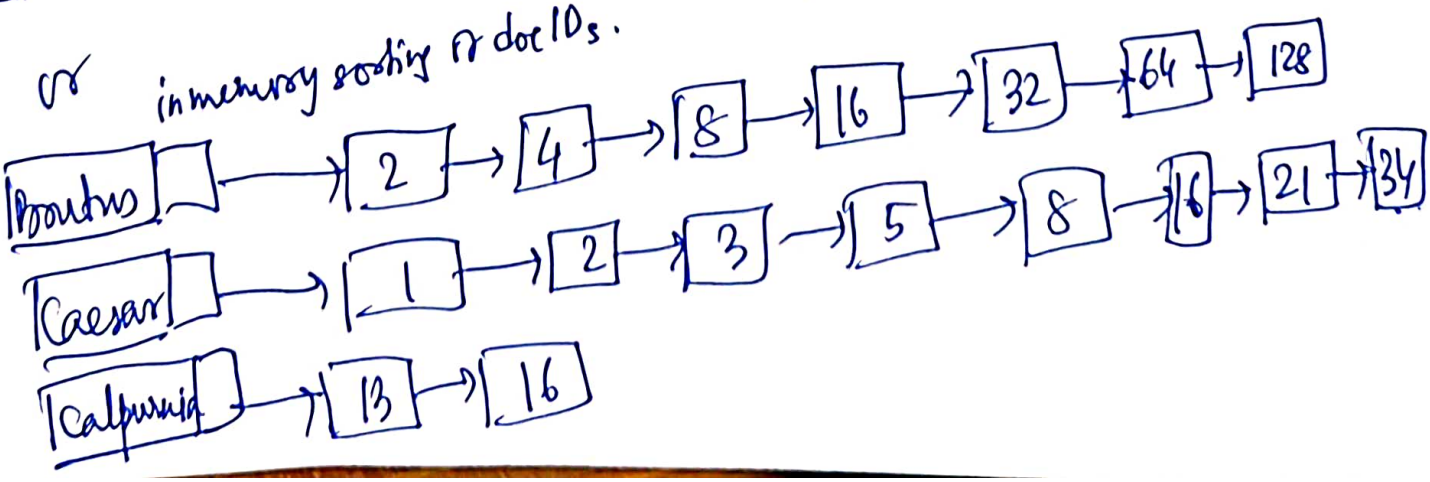
3 words.

All variants ← LIMIT* → LIMITED, LIMITATION.

~~Brutus AND CAT~~

Execute: Brutus AND Calpurnia AND Caesar

or in memory sorting of docIDs.



What about OR queries?

(doc freq madding OR corrd doc freq. coord) AND (ignoble OR strife).

Sum

Sum

order by the sum & then execute the ANDs.

Google search → Boolean queries.

$[w_1, w_2 \ldots w_n] \longrightarrow w_1$ AND $w_2 \ldots w_n$.

When does it fail?

- very few docs.
- all w_i are unfortunately in the anchor text
- all words present are variants of w_i