https://vvtesh.sarahah.com/

Information Retrieval

Venkatesh Vinayakarao
venkateshv@cmi.ac.in

Chennai Mathematical Institute

What we find changes who we become.

-Peter Morville.

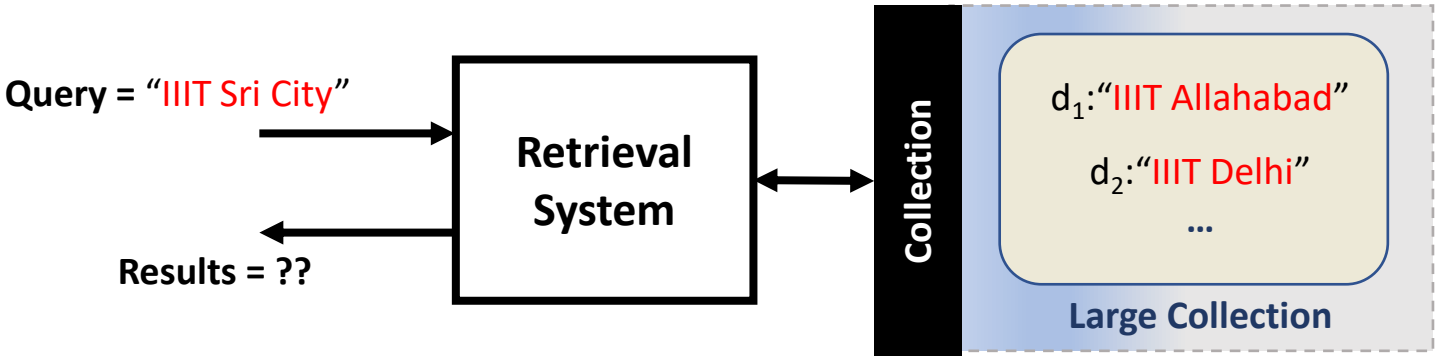
A Simple Retrieval System

Our first IR system.

Simple Retrieval Problem

- A **collection** with 5 **documents** having the following contents
  - d1: IIIT ALLAHABAD
  - d2: IIIT DELHI
  - d3: IIIT GUWAHATI
  - d4: IIIT KANCHIPURAM
  - d5: IIIT SRI CITY
- **Query** is
  - IIIT SRI CITY
- Which **document** will you match and why?

The Problem


Query = "IIIT Sri City"
Results = ??
Retrieval System
Collection
d_1:"IIIT Allahabad"
d_2:"IIIT Delhi"
...
Large Collection

One (bad) Approach

- First match the **term** IIIT.
  - Filter out documents that contain this term.
- Next match the **term** Sri.
  - Filter out documents that contain this term.
- Next match the **term** City.
  - Filter out documents that contain this term.

Three iterations!
Quiz: Can we do better?

Representing Documents as A Term-Document Incidence Matrix

Documents

Terms	d1	d2	d3	d4	d5
IIIT	1	1	1	1	1
ALLAHABAD	1	0	0	0	0
DELHI	0	1	0	0	0
GUWAHATI	0	0	1	0	0
KANCHIPURAM	0	0	0	1	0
SRI	0	0	0	0	1
CITY	0	0	0	0	1

Query Processing

Documents

Terms	d1	d2	d3	d4	d5
IIIT	**1**	**1**	**1**	**1**	**1**
ALLAHABAD	1	0	0	0	0
DELHI	0	1	0	0	0
GUWAHATI	0	0	1	0	0
KANCHIPURAM	0	0	0	1	0
SRI	0	0	0	0	1
CITY	0	0	0	0	1

All 1s in d5 for IIIT Sri City!

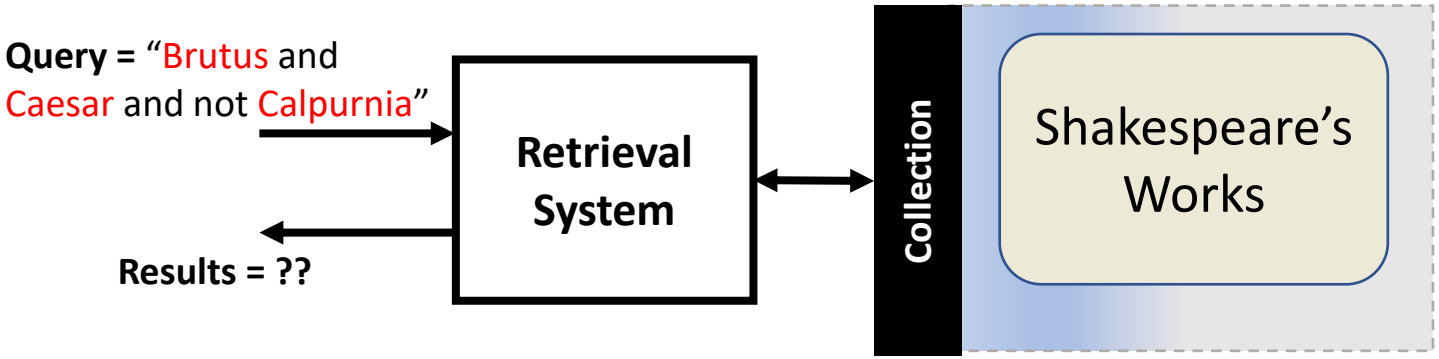
Boolean Retrieval

Match or No-Match! No ranking of results.

Boolean Retrieval Model

- We discuss a Boolean retrieval model which assumes the following:
  - Boolean queries are queries using *AND, OR* and *NOT* to join query terms.
  - Views each document as a set of words.
  - Either there is a match or no-match. We do not rank the results.
- Perhaps the simplest model to build an IR system.

Simple Conjunctive Queries



A Term-Document Incidence Matrix Example

Documents

Terms	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 and not Calpurnia"

What is the best way to get to the answer?

The Answer

"Brutus and Caesar and not Calpurnia"

1	1	0	1	0	0
1	1	0	1	1	1
1	0	1	1	1	1
AND					
1	0	0	1	0	0

Document 1 and 4 satisfy our query.

Disadvantages of term-document Matrix

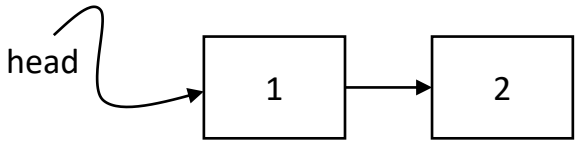
- To add new documents, we need to add new columns.
- For a large collection (say Millions of documents),
  - Each document has far fewer words from the dictionary.
  - So, the matrix is sparse.

Can we do better?

Instead of handling both 1s and 0s, can we only have the 1s?

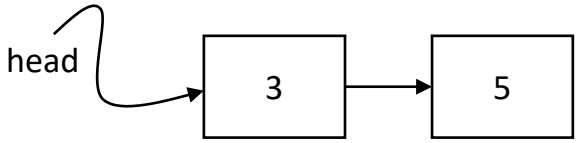
Arrays Vs. Linked Lists

- {1,1,0,0,0,0,0,0,0,…. 10K elements} is



A Linked List!

- {0,0,1,0,1,0,0,…..10K elements} is



A Linked List!

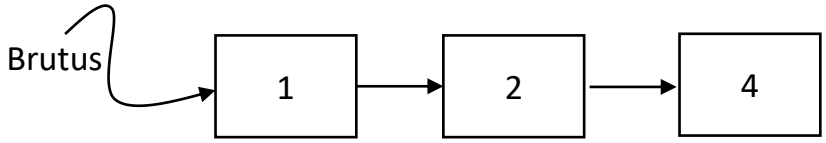
The Problem

- An n-Dimensional Vector can be represented as
  - an array of n elements.
    - Example: (1,1,1) is int[] A = {1,1,1}; in Java.
- So, a large vector {1,1,0,0,0,0,0,0,0,…. 10K elements} is
  - an array with 10K elements where only first two elements are 1s.

Is there a better way to represent this data?

Representing term-document Data

Terms	Documents					
	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



Linked List Idea in Practice

Tokenization

- Task
  - Chop documents into pieces.
  - Throw away characters such as punctuations.
  - Remaining words are called tokens.
- Example
  - Document 1
    - I did enact Julius Caesar. I was killed i' the Capitol; Brutus killed me.
  - Document 2
    - So let it be with Caesar. The noble Brutus hath told you Caesar was ambitious

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

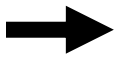
Term	docID
I	1
did	1
enact	1
julius	1
caesar	1
I	1
was	1
killed	1
i'	1
the	1
capitol	1
brutus	1
killed	1
me	1
so	2
let	2
it	2
be	2
with	2
caesar	2
the	2
noble	2
brutus	2
hath	2
told	2
you	2
caesar	2
was	2
ambitious	2

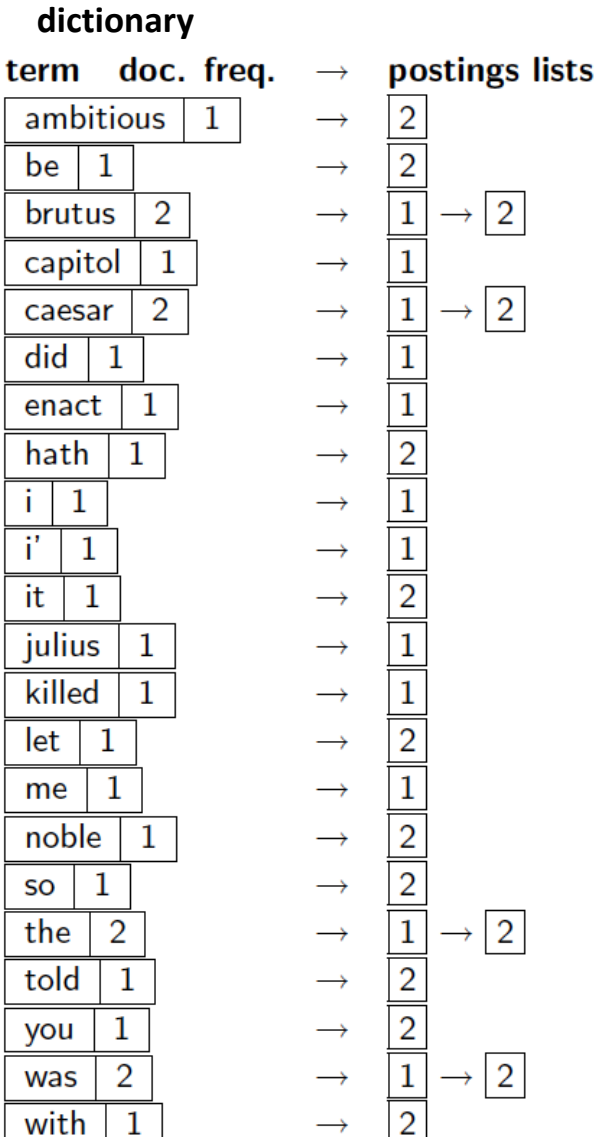
Term	docID
ambitious	2
be	2
brutus	1
brutus	2
capitol	1
caesar	1
caesar	2
caesar	2
did	1
enact	1
hath	1
I	1
I	1
i'	1
it	2
julius	1
killed	1
killed	1
let	2
me	1
noble	2
so	2
the	1
the	2
told	2
you	2
was	1
was	2
with	2

Inverted Index: Dictionary & Postings

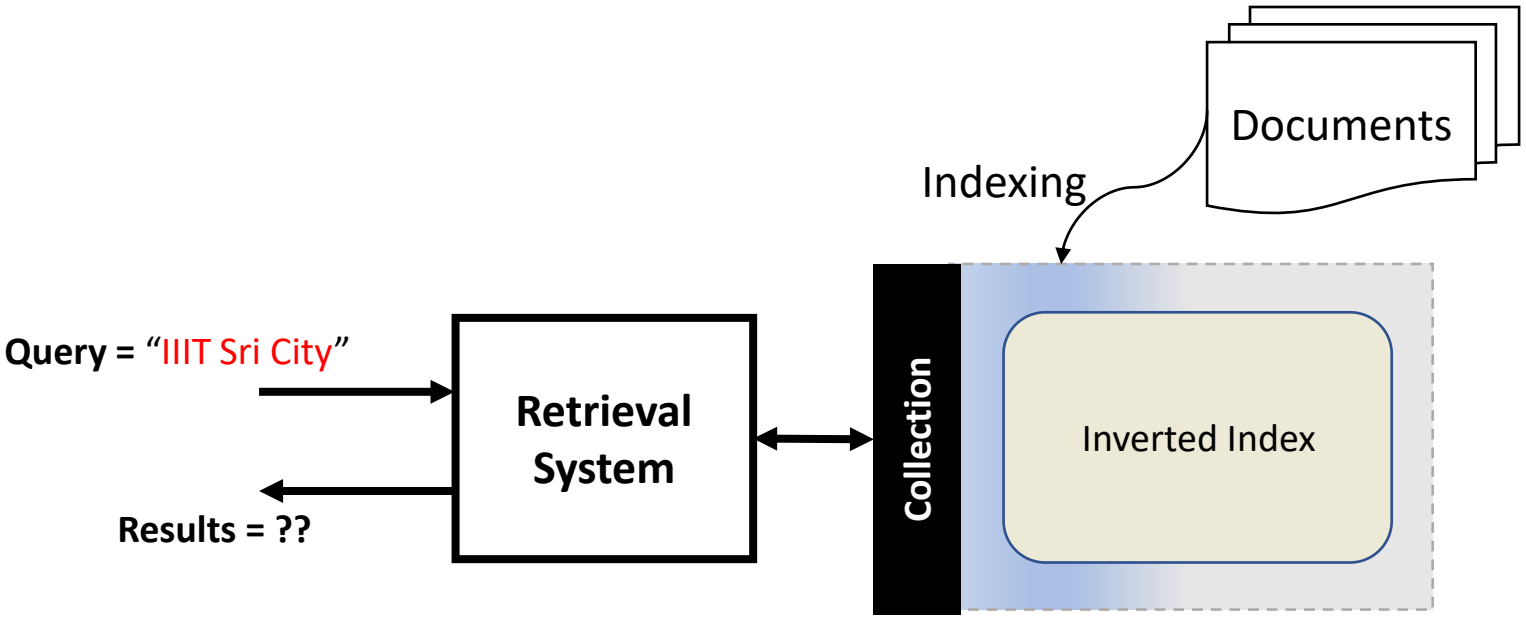
- Multiple term entries in a single document are merged.
- Split into Dictionary and Postings

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





The Big Picture



Thank You