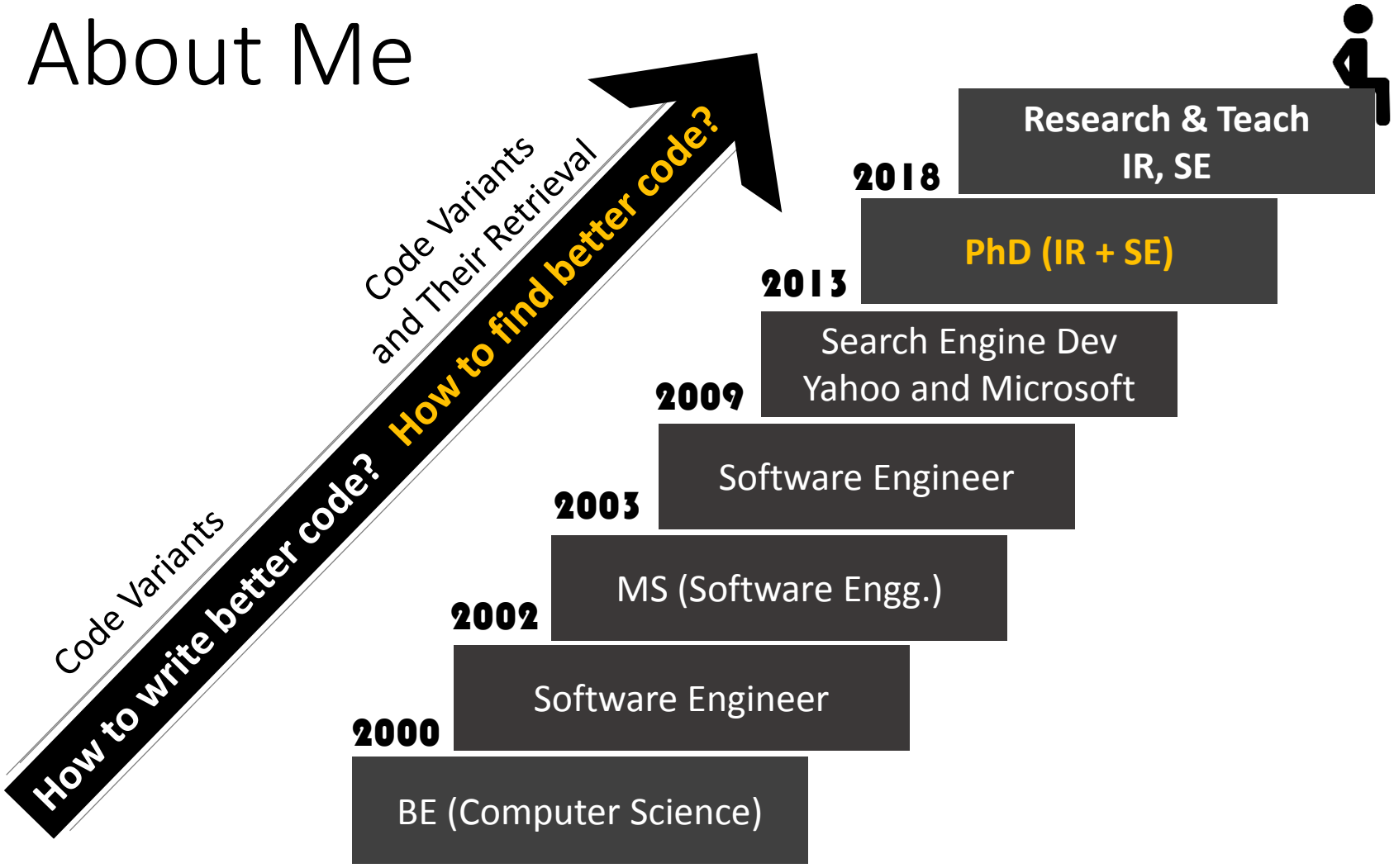
The Mathematics Behind Web Search Engines

Venkatesh Vinayakarao
venkateshv@cmi.ac.in
http://vvtesh.co.in

Chennai Mathematical Institute

Simplicity boils down to two steps. Identify the essential. Eliminate the rest. **–Leo Babauta.**

About Me



[Why code search is interesting?]
[Why code search is hard?]

Agenda

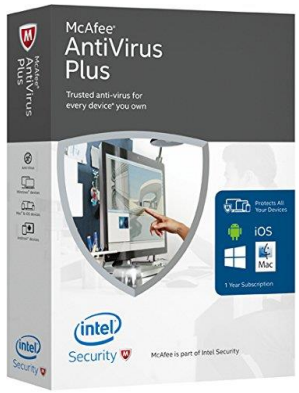
- Abstraction and Mathematical Models
- Information Retrieval
  - Vector Space Models
  - Indexing
  - Evaluation
- Research Trends and Open Problems

How to tame complexity?

Can we learn from the world around us?

How did we humans build this?

We could build these too!



Taming Complexity

Contents

Taming Complexity

Key Principles

1. Hierarchy
2. Abstraction
3. Keeping Related Things Together
4. …

Models

- A model is a representation of an *idea*, an *object*, a *process* or even a *system*
- Used as tools to understand (define, quantify, visualize, …) the real world.

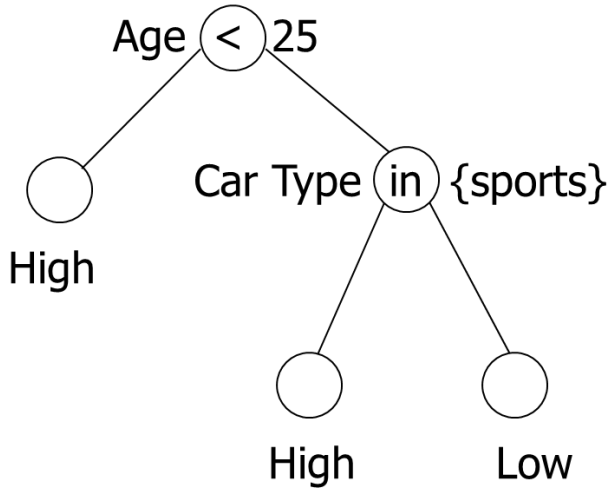
Decision Tree Model

Data Set

Age	Car Type	Risk
23	Family	High
17	Sports	High
43	Sports	High
68	Family	Low
32	Truck	Low
20	Family	High

Question: **What is the risk likely to be (high or low) if age is below 25?**

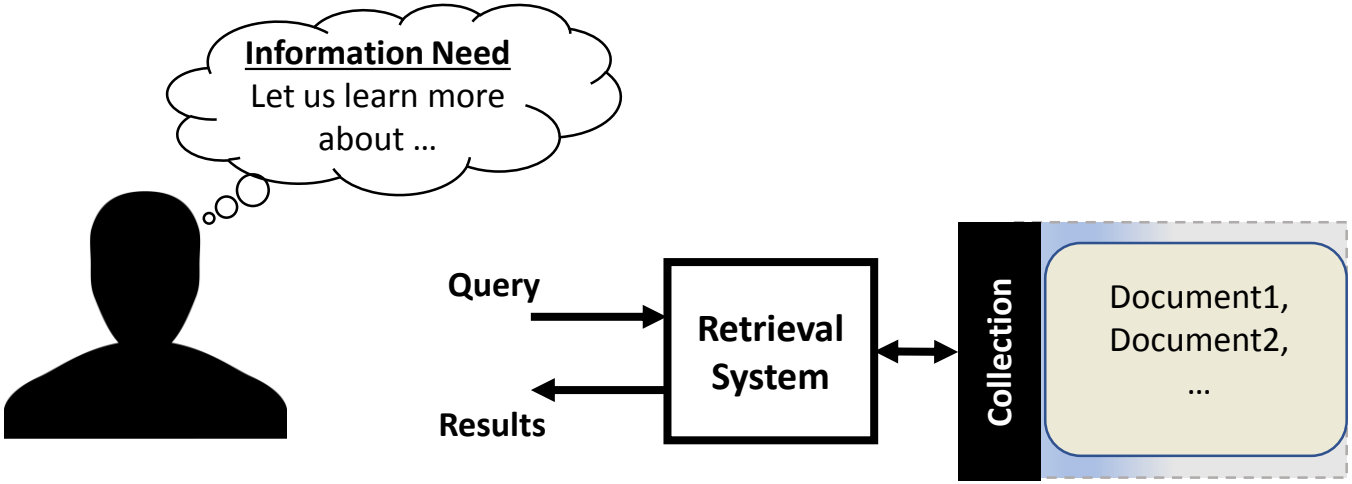
Decision Tree



Models in Information Retrieval

A Bag of Words Vector Space Model for Search Engines

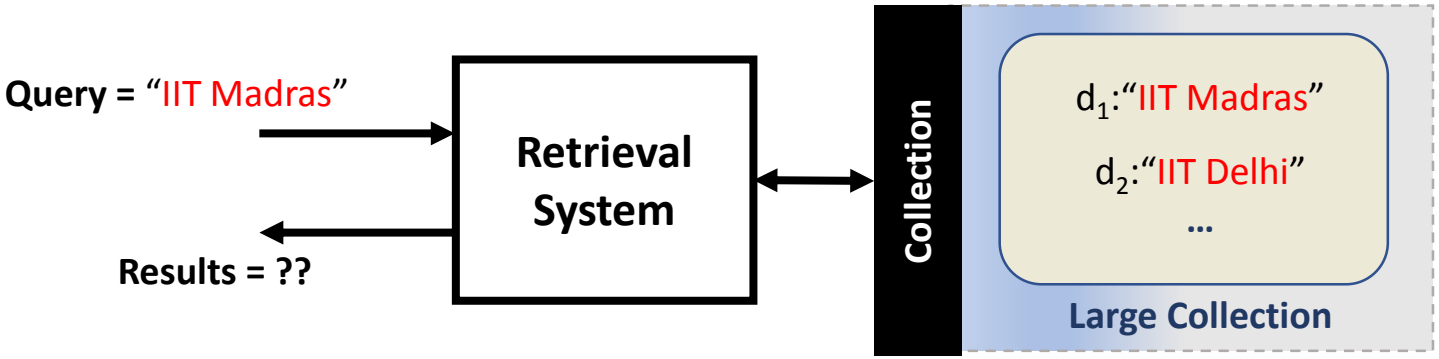
Search Engines



Simple Retrieval Problem

- A **collection** with 5 **documents** having the following contents
  - d1: IIT Madras
  - d2: IIT Delhi
  - d3: IIT Kanpur
  - d4: IIT Goa
  - d5: IIT Bombay
- **Query** is
  - IIT Madras
- Which **document** will you match and why?

The Problem: How to Build a Retrieval System?



One (bad) Approach

- First match the **term** IIT.
  - Filter out documents that contain this term.
- Next match the **term** Madras.
  - Filter out documents that contain this term.

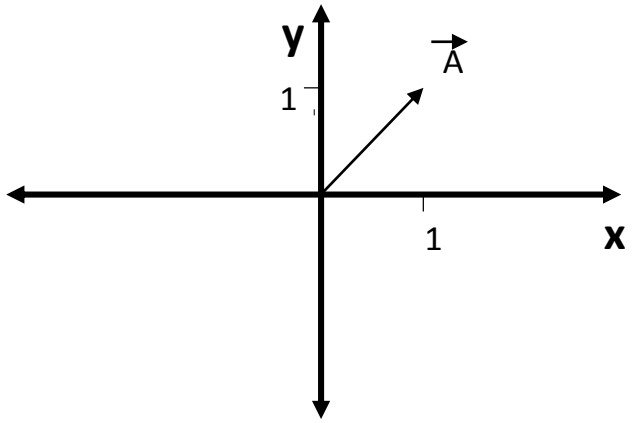
Multiple iterations!
Can we do better?

A Better Approach

Revisiting Vectors

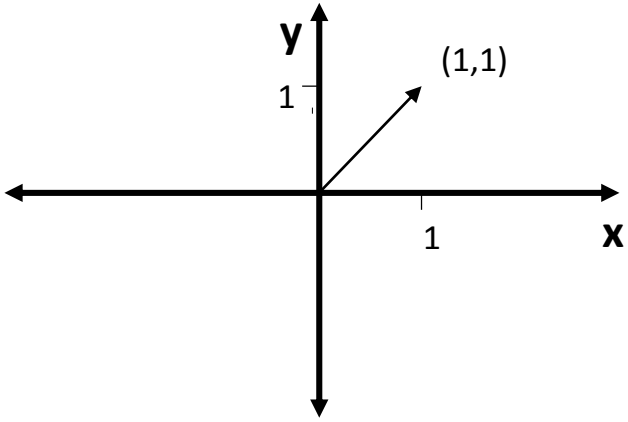
Vectors

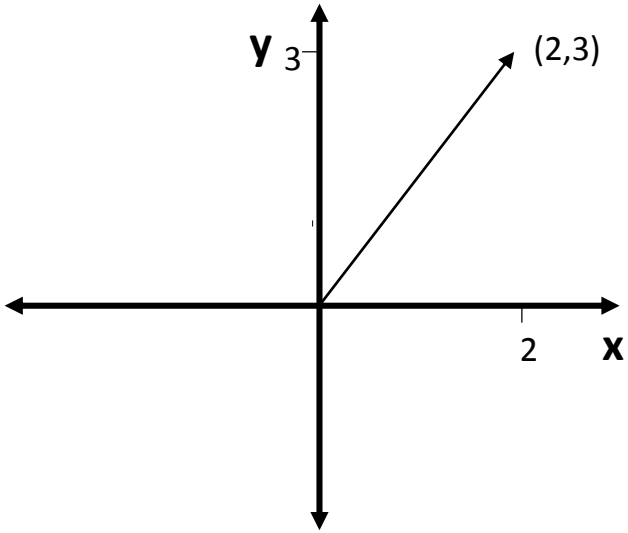
- Geometric entity which has magnitude and direction



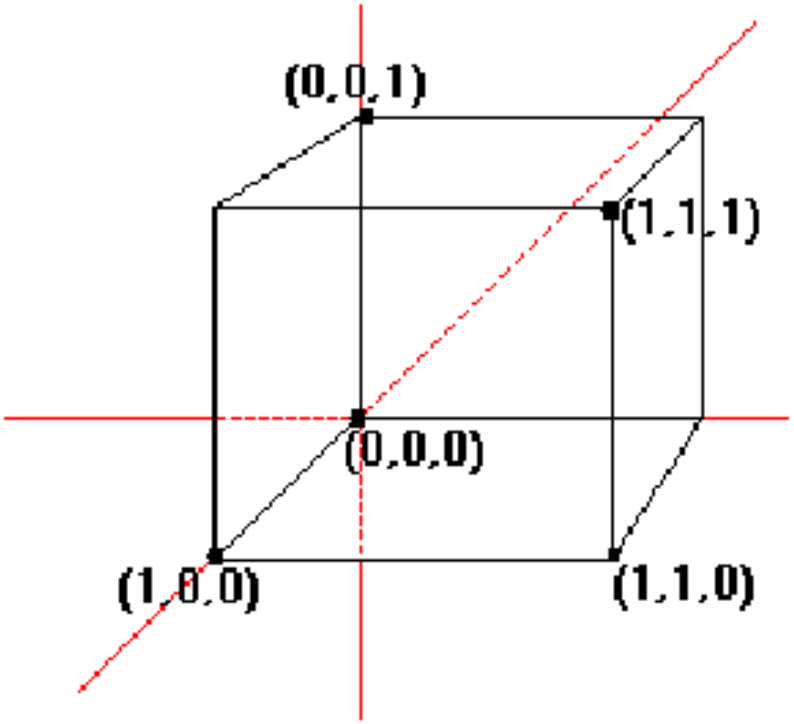
- If (x,y) is our vector of interest, this figure shows $\vec{A}$ vector = (1,1).

How is (2,3) Different?





What is (1,1,1) ?

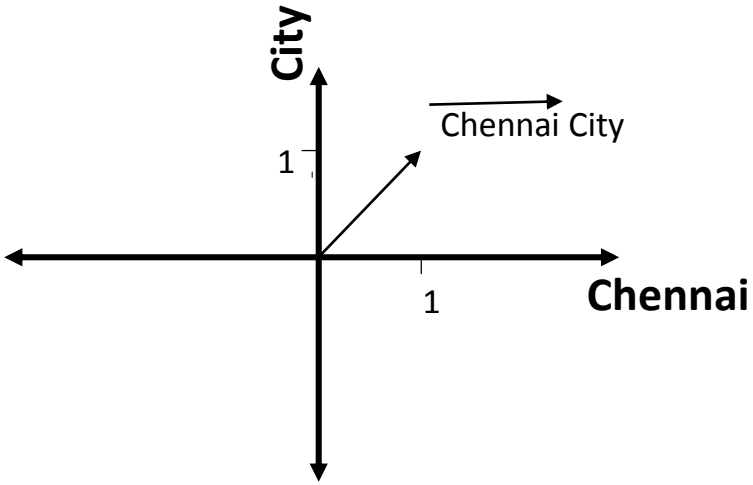


Remember!

A number is just a mathematical object. We give meaning to it!

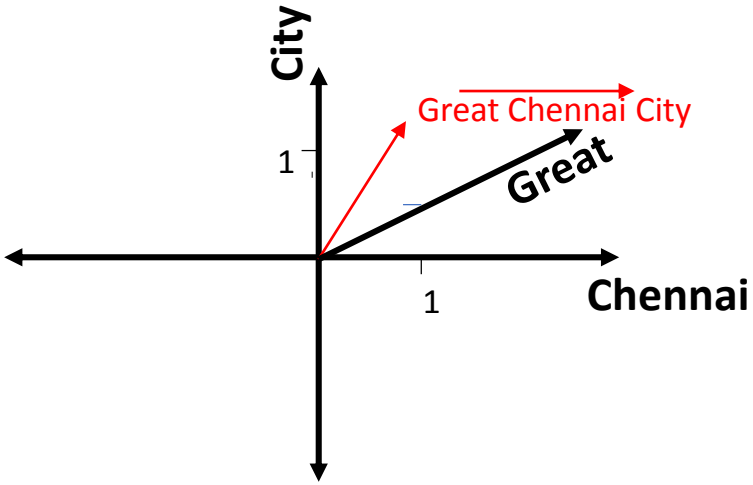
Sentences are Vectors

- "Chennai City" as a vector



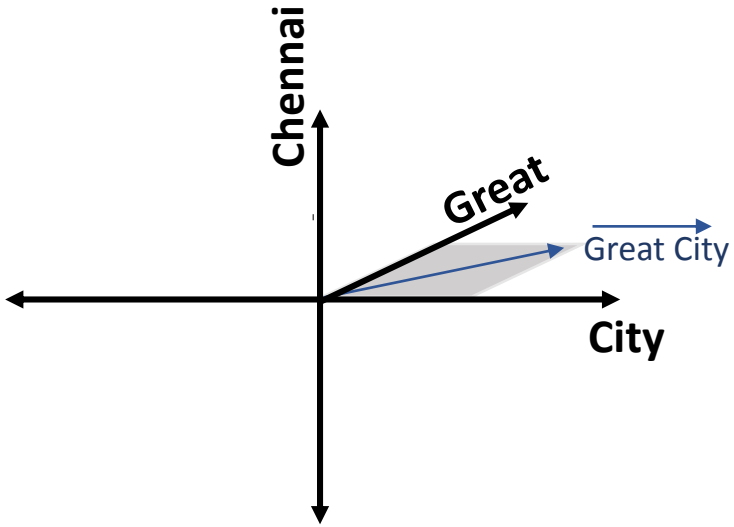
Sentences are Vectors

- "Great Chennai City" is a 3-dimensional vector



Sentences are Vectors

- On this 3D space,
  - "Great City" vector will lie on the x (City) and z (Great) plane.
  - "Great City" is (1,0,1).



Natural Language Phrases as Vectors

Let query q = "IIT Delhi".

Let document, d_1 = "IIT Madras" and d_2 = "IIT Delhi".

	IIT	Delhi	Madras
q	1	1	0
d_1	1	0	1
d_2	1	1	0

q = (1,1,0), d_1= (1,0,1) and d_2 = (1,1,0)

Quiz

- Considering the following vectors:

	IIIT	Sri	City	Delhi
q	1	1	1	0
d_1	1	1	1	0
d_2	1	0	0	1

- What is the Natural Language (NL) equivalent of (0,1,1,0) ?
- What is the NL equivalent of (1,0,0,1) ?
- What is the vector for Delhi?

Similarity Score

- D1 = “Chennai”
- D2 = “Delhi”

- Quiz
  - What is the angle between D1 and D2 vectors?
  - On a scale of 0 – 1, how similar are D1 and D2?

0 – 90 to 1 – 0: How?

	0°	30°	45°	60°	90°
sin θ	0	$\frac{1}{2}$	$\frac{1}{\sqrt{2}}$	$\frac{\sqrt{3}}{2}$	1
cos θ	1	$\frac{\sqrt{3}}{2}$	$\frac{1}{\sqrt{2}}$	$\frac{1}{2}$	0
tan θ	0	$\frac{1}{\sqrt{3}}$	1	$\sqrt{3}$	Not defined

Back to Trigonometry: Dot Product

- If x and y are non-unit vectors, what is the cosine of angle between them (cos Θ)?

$$\mathbf{a} \cdot \mathbf{b} = \|\mathbf{a}\| \, \|\mathbf{b}\| \cos(\theta)$$

$$Cosine\ Similarity = \cos(\theta) = \frac{a.b}{||a||\,||b||}$$

Matching Documents to Queries

- Document as a vector of term-occurrence

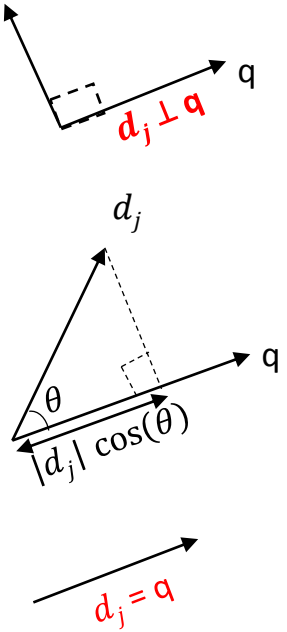
$$d_j = (w_{1j}, w_{2j}, \ldots w_{nj})$$

- Query as a vector of term-occurrence

$$q = (w_{1q}, w_{2q}, \ldots w_{mq})$$

- Similarity between these vectors can be represented as

$$Cosine\ Similarity = \cos(\theta) = \frac{d_j . q}{||d_j||\ ||q||}$$



Example

Let query q = "BITS Pilani".

Let document, d_1 = "BITS Pilani Goa Campus" and d_2 = "IIIT Delhi".

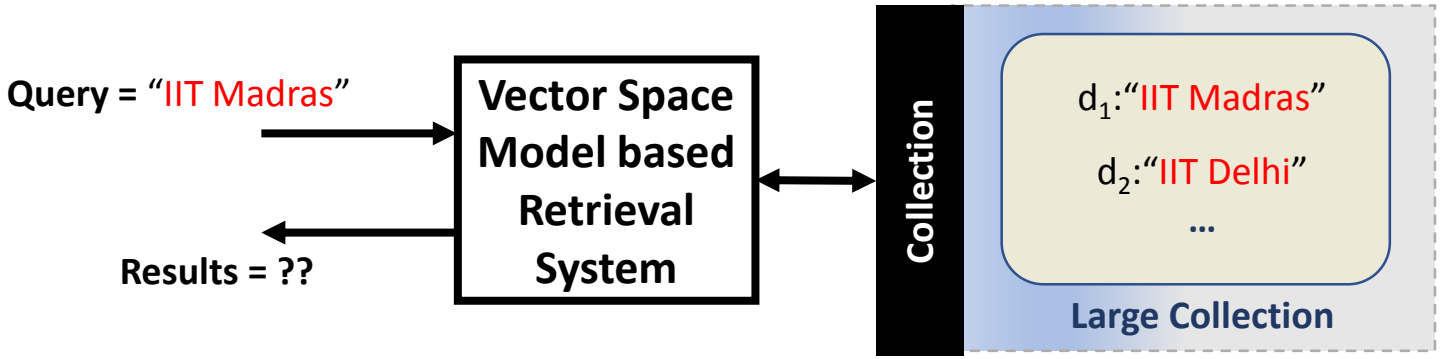
	BITS	Pilani	Goa	Campus	IIIT	Delhi
q	1	1	0	0	0	0
d_1	1	1	1	1	0	0
d_2	0	0	0	0	1	1

In our VSM, q = (1,1,0,0,0,0), d_1= (1,1,1,1,0,0) and d_2 = (0,0,0,0,1,1)

similarity(d_1, q) = $$\frac{d_1.q}{||d_1||\,||q||} = \frac{1.1 + 1.1}{\sqrt{1^2+1^2+1^2+1^2}\,\sqrt{1^2+1^2}} = 0.71.$$

similarity(d_2, q) = $$\frac{d_2.q}{||d_2||\,||q||} = 0.$$

The Problem: How to Build a Retrieval System?



What if some terms repeat in the sentences?

Which of the Following are Sets?

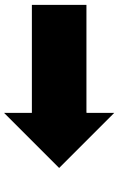
- {1, 2, 3, 4, 5, 6, 5, 7, 8, 9, 10, 11, 12, 13}
- {A, B, C, D, E, F, G, H, I, I, J, K, L, M, N, O}
- {apple, banana, orange, apple, banana, orange}

Bag

- {1, 2, 3, 4, 5, 6, 5, 7, 8, 9, 10, 11, 12, 13}
- {A, B, C, D, E, F, G, H, I, I, J, K, L, M, N, O}
- {apple, banana, orange, apple, banana, orange}

Set of Words Representation

- "IIIT Sri City" → {IIIT, Sri, City}
- "IIIT Sri City, Sri City" → {IIIT, Sri, City}

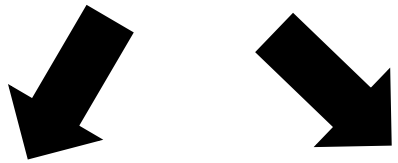


	IIIT	Sri	City
q	1	1	1

Leads to same term-document matrix

Bag of Words Representation

- "IIIT Sri City" → {IIIT, Sri, City}
- "IIIT Sri City, Sri City" → [IIIT, Sri, Sri, City, City]



IIIT Sri City

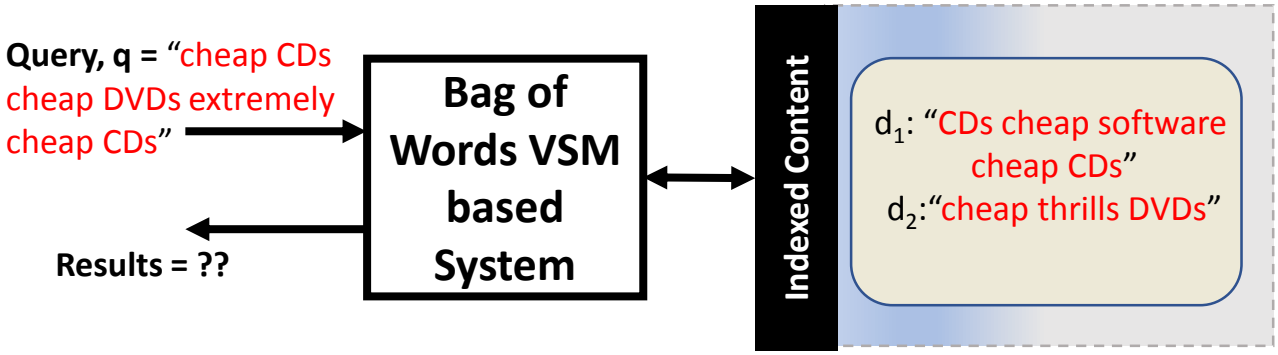
	IIIT	Sri	City
q	1	1	1

IIIT Sri City, Sri City

	IIIT	Sri	City
q	1	2	2

Leads to different term-document matrix

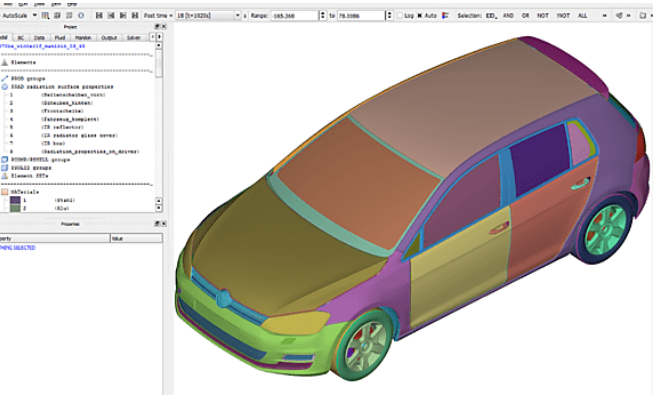
Which Document to Retrieve?

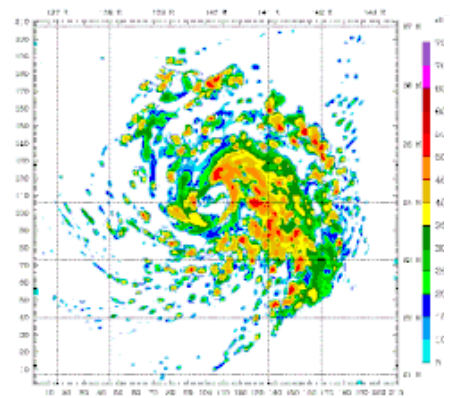


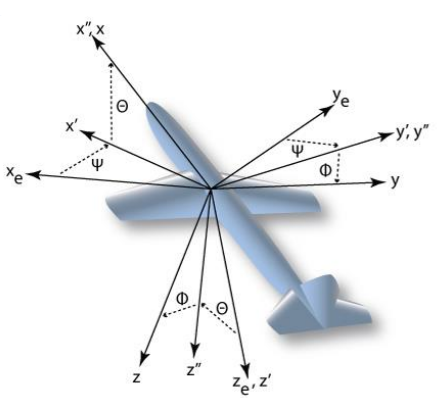
	cheap	CDs	DVDs	extremely	software	thrills
q	3	2	1	1	0	0
d_1	2	2	0	0	1	0
d_2	1	0	1	0	0	1

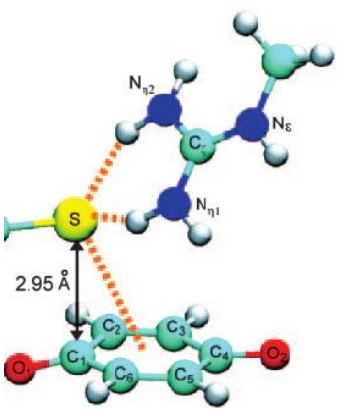
$sim(q,d_1) = 0.86$

$sim(q,d_2) = 0.59$









"Abstraction is one of the greatest visionary tools ever invented by human beings to imagine, decipher, and depict the world."

Jerry Saltz

Indexing

Tokenization

- Task
  - Chop documents into pieces.
  - Throw away characters such as punctuations.
  - Remaining terms 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

Example taken from Manning's IR book

Sort

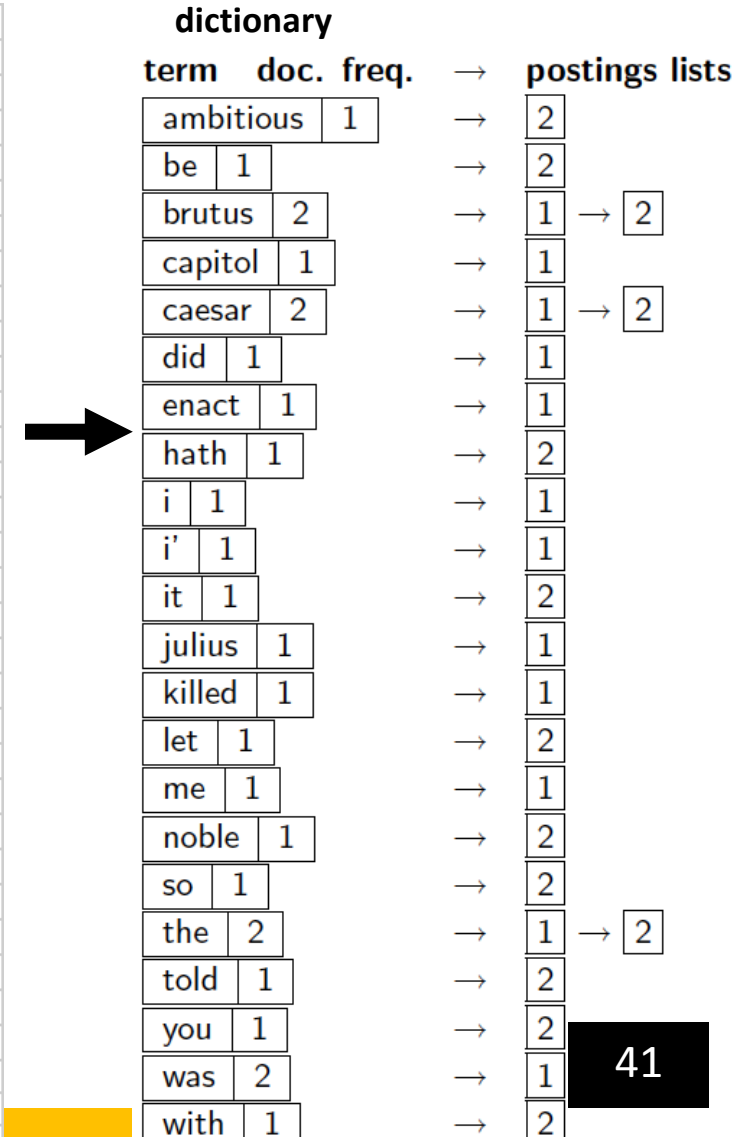
Term	docID
I	1
did	1
enact	1
julius	1
caesar	1
I	1
was	1
killed	1
i'	1
the	1
capitol	1
brutus	1
killed	1
me	1
so	2
let	2
it	2
be	2
with	2
caesar	2
the	2
noble	2
brutus	2
hath	2
told	2
you	2
caesar	2
was	2
ambitious	2

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	

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



Boolean queries: Exact match

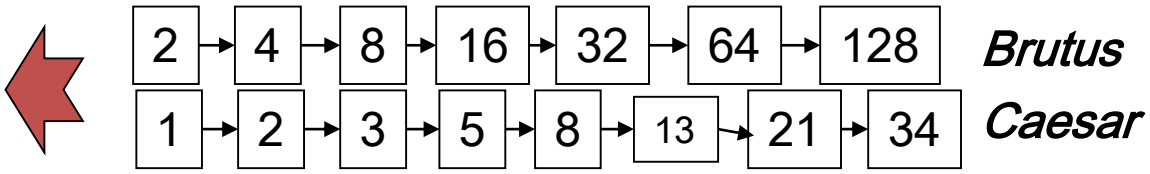
- The Boolean retrieval model is being able to ask a query that is a Boolean expression:
  - Boolean Queries are queries using *AND, OR* and *NOT* to join query terms.
    - Views each document as a set of words.
    - Is precise: document matches condition or not.
  - Just a simple IR system.

Query processing: AND

- Consider processing the query:

  Brutus *AND* ***Caesar***

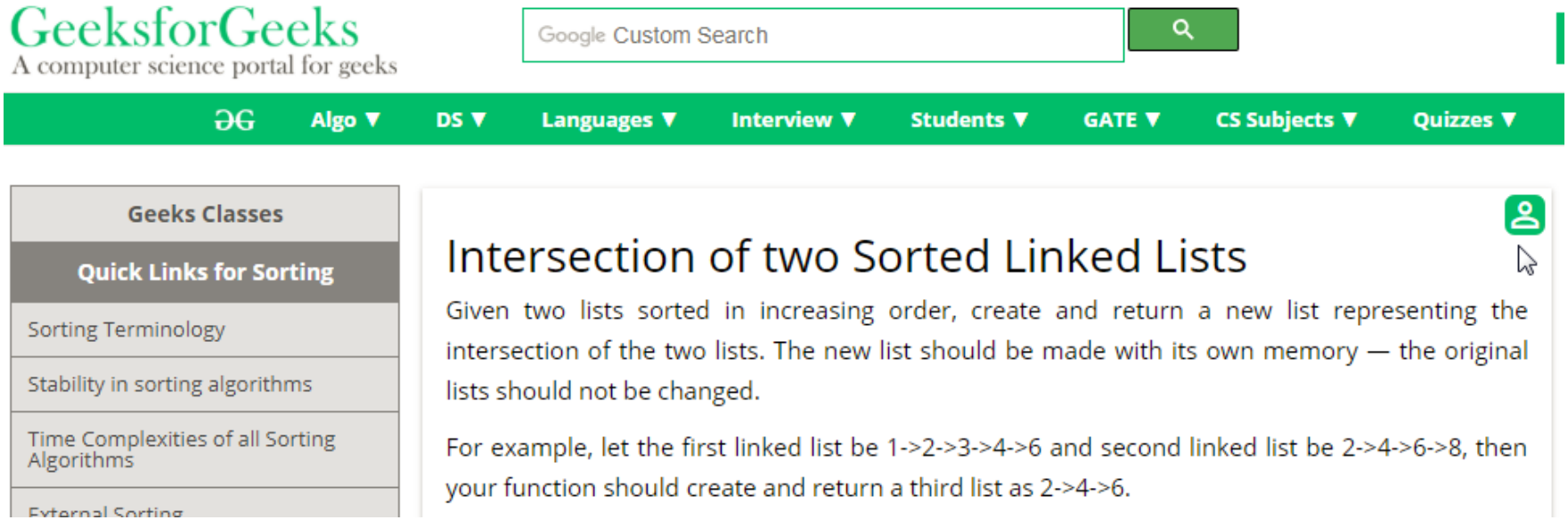
  - Locate ***Brutus*** in the Dictionary;
    - Retrieve its postings.
  - Locate ***Caesar*** in the Dictionary;
    - Retrieve its postings.
  - "Merge" the two postings (intersect the document sets):



Example taken from Manning's IR book

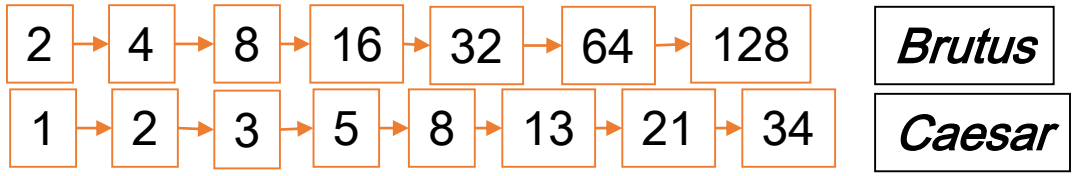
Common Interview Question

- https://www.geeksforgeeks.org/intersection-of-two-sorted-linked-lists/



The Merge

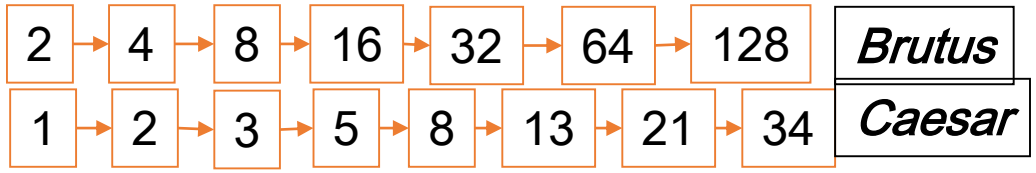
- Walk through the two postings simultaneously
  - Clue: Use two pointers



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

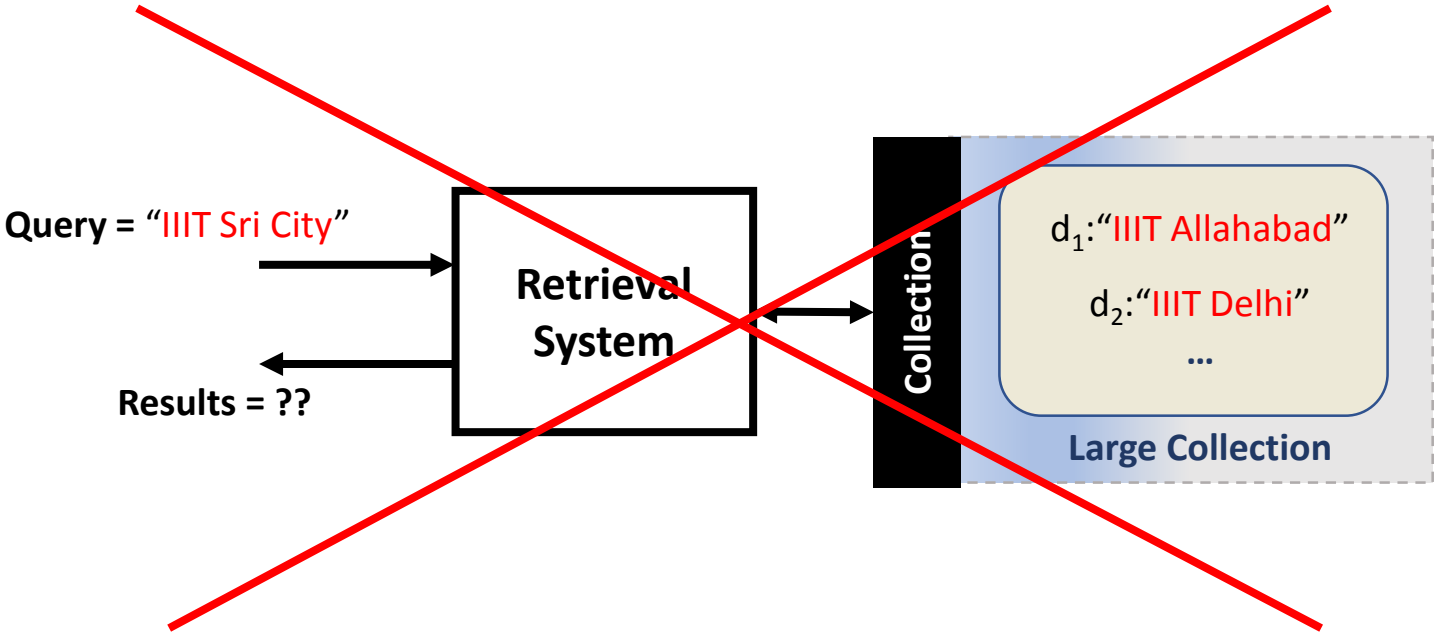
The Merge

- Walk through the two postings simultaneously
  - Clue: Use two pointers

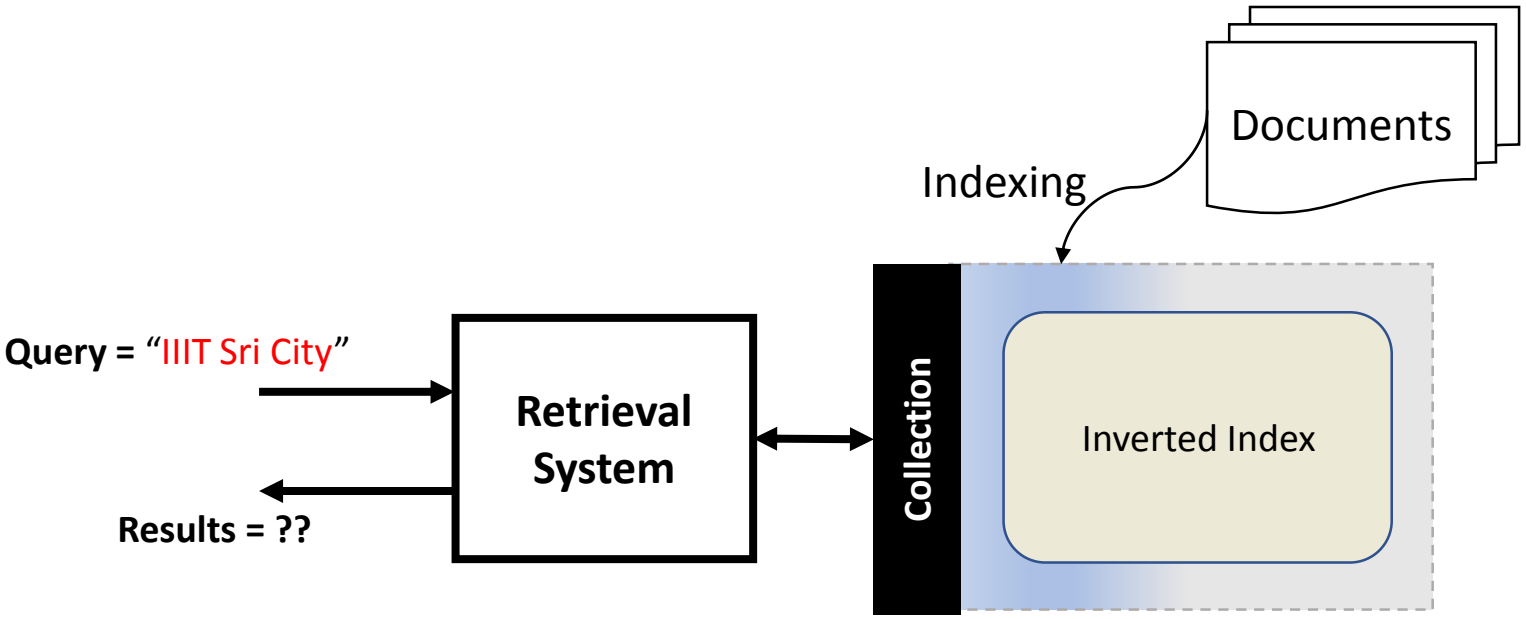


If the list lengths are *x* and *y*, the merge takes O(*x*+*y*) operations.
Crucial: postings sorted by docID.

The Big Picture

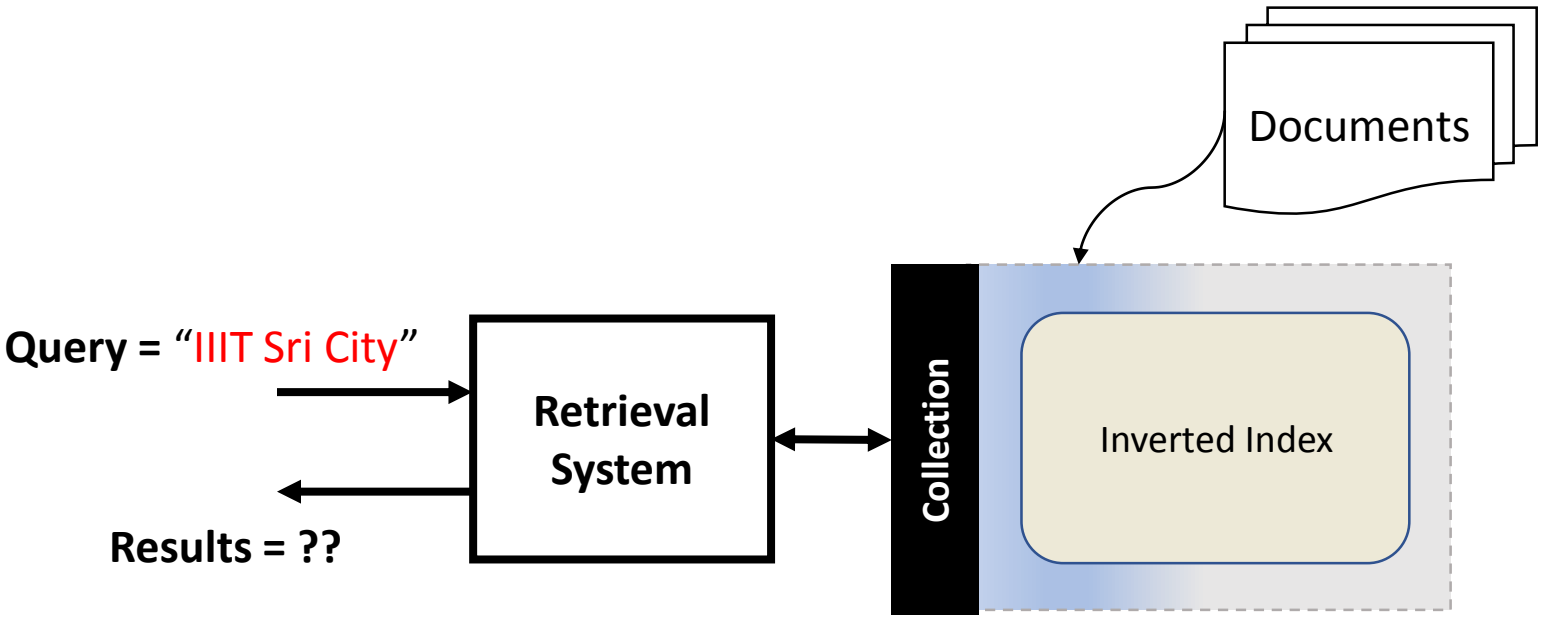


The Big Picture



Phrasal Queries

- What if we do not want to match IIIT Delhi?



One (bad) Approach

- Index all biwords
  - Friends, Romans, Countrymen → Friends Romans, Romans Countrymen
- What is the problem?
- How do you match the query IIIT Sri City, Chittoor?
  - IIIT Sri AND Sri City AND City Chittoor must exist.

A Better Approach

- Store Positional Information

<term, number of docs containing term;
doc1: position1, position2 … ;
doc2: position1, position2 … ;
etc.>

Positional Index

to, 993427:
⟨ 1, 6: ⟨7, 18, 33, 72, 86, 231⟩;
2, 5: ⟨1, 17, 74, 222, 255⟩;
4, 5: ⟨8, 16, 190, 429, 433⟩;
5, 2: ⟨363, 367⟩;
7, 3: ⟨13, 23, 191⟩; . . . ⟩

be, 178239:
⟨ 1, 2: ⟨17, 25⟩;
4, 5: ⟨17, 191, 291, 430, 434⟩;
5, 3: ⟨14, 19, 101⟩; . . . ⟩

"to" appears six times in d1 at positions 7, 18, ….
"to" appears 993K times overall.

Which document is likely to contain "to be"?

Example taken from Manning's IR book

Proximity Search

- IIIT /3 Chittoor
  - /k means “within k words of (on either side)”

- Merging postings is expensive
  - Index well-known phrases such as “Taj Mahal”

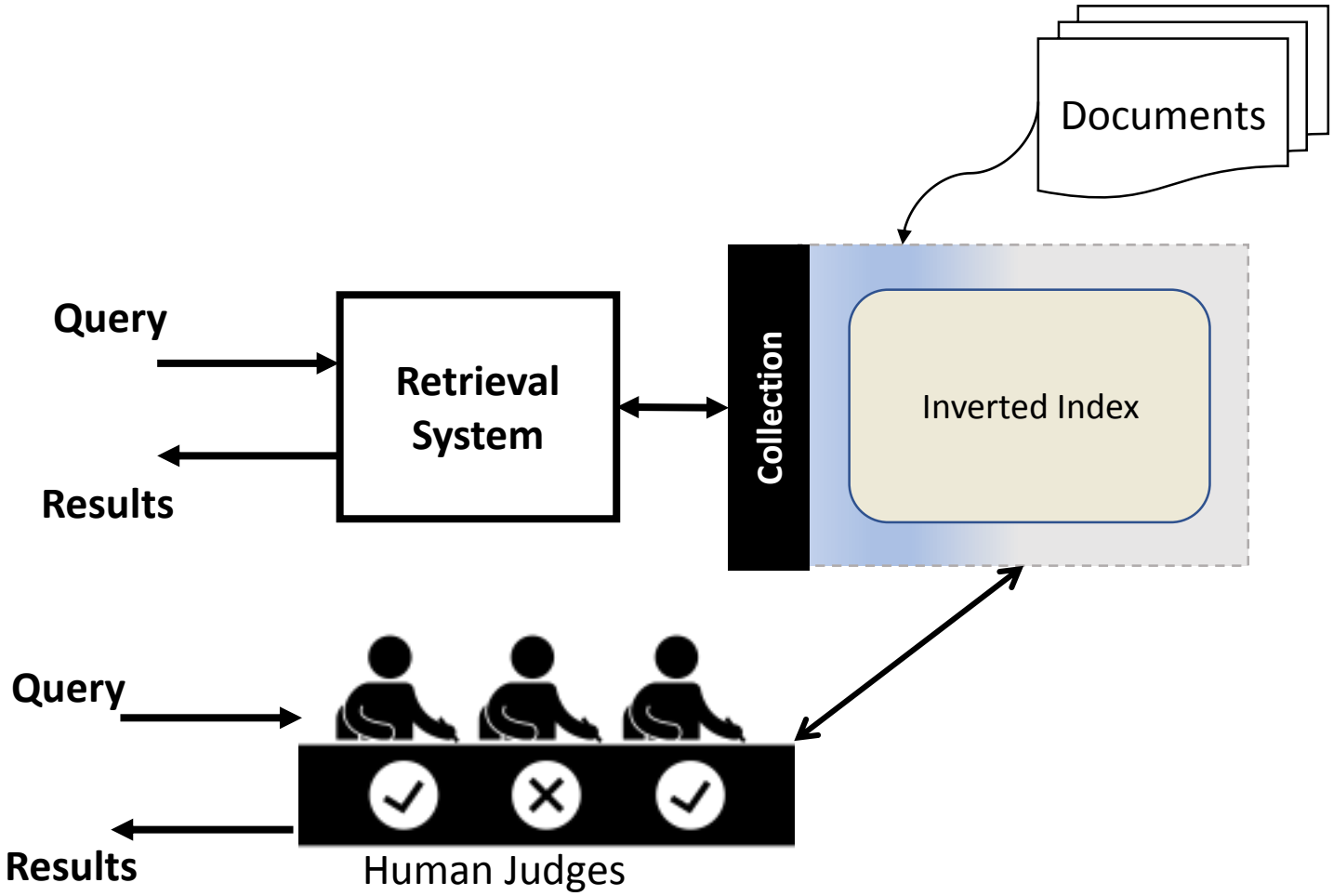
Evaluation

Comparing Retrieval Systems

How Good is Our System?

- A **collection** having the following contents
  - d1: IIIT ALLAHABAD
  - d2: IIIT DELHI
  - d3: IIIT GUWAHATI
  - d4: CMI
  - d5: IIIT SRI CITY
  - d6: KREA SRI CITY
- **Query** is
  - SRI CITY
- **Result** is
  - IIIT SRI CITY
  - KREA SRI CITY

Evaluation


Documents
Query
Retrieval System
Results
Collection
Inverted Index
Query
Results
Human Judges

How Good is Our System?

- A **collection** having the following contents
  - d1: IIIT ALLAHABAD
  - d2: IIIT DELHI
  - d3: IIIT GUWAHATI
  - d4: CMI
  - d5: IIIT SRI CITY
  - d6: KREA SRI CITY
- **Query** is
  - IIIT
- **Result** is
  - IIIT SRI CITY
  - KREA SRI CITY

Objective

We want all relevant documents and only relevant documents

Relevance

- How many **relevant** documents?
  - Four (IIIT SRI CITY, IIIT ALLAHABAD, IIIT DELHI, IIIT GUWAHATI)
- How many **retrieved** documents?
  - Two (IIIT SRI CITY, KREA SRI CITY)

How to quantify the “goodness” of our system?

Terminology

- Documents we see in results are "**positive**"
  - Positive
    - + IIIT SRI CITY,
    - + KREA SRI CITY
  - Negative
    - - IIIT ALLAHABAD
    - - IIIT DELHI
    - - IIIT GUWAHATI
    - - CMI

Terminology

- Documents that we correctly classify are "**true**"
  - Positive
    - + IIIT SRI CITY (true)
    - + KREA SRI CITY
  - Negative
    - - IIIT ALLAHABAD
    - - IIIT DELHI
    - - IIIT GUWAHATI
    - - CMI (true)

Here, query is "IIIT"

Quiz

- All retrieved results =
  1. tp + fp
  2. tp + fn
  3. tn + fp
  4. tn + fn

Legend

tp = true positive

tn = true negative

fp = false positive

fn = false negative

Quiz

- All retrieved results =
  1. tp + fp
  2. tp + fn
  3. tn + fp
  4. tn + fn

Legend

tp = true positive

tn = true negative

fp = false positive

fn = false negative

Quiz

- All relevant results =
  1. tp + fp
  2. tp + fn
  3. tn + fp
  4. tn + fn

Legend

tp = true positive

tn = true negative

fp = false positive

fn = false negative

Quiz

- All relevant results =
  1. tp + fp
  2. tp + fn
  3. tn + fp
  4. tn + fn

Legend

tp = true positive

tn = true negative

fp = false positive

fn = false negative

You have 100% Precision

- Everything you retrieved were relevant.
  - fp = 0

You have 100% Recall when

- You retrieved everything that were relevant. (Note: You could have retrieved more).
  - fn = 0

Example

- The following list of Rs and Ns represents relevant (R) and nonrelevant (N) returned documents in a list of 20 documents retrieved in response to a query from a collection of 10,000 documents. This list shows 6 relevant documents. Assume that there are 8 relevant documents in total in the collection. Calculate Precision and Recall.

RRNNN NNNRN RNNNR NNNNR

Precision and Recall

- Precision = 6/20
- Recall = 6/8

Precision and Recall

Precision: fraction of retrieved docs that are relevant

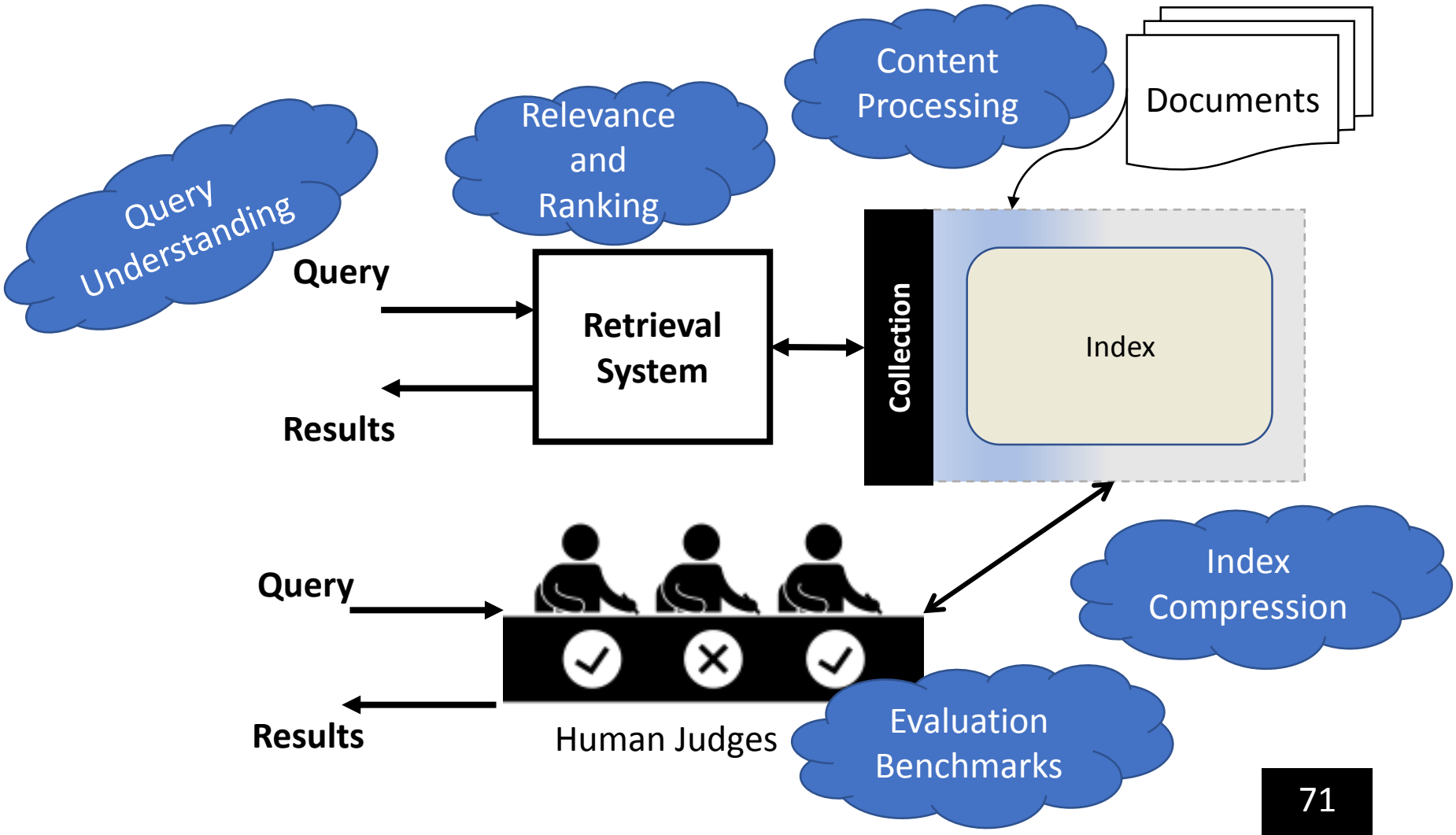
$$= tp/(tp + fp)$$

Recall: fraction of relevant docs that are retrieved

$$= tp/(tp + fn)$$

	Relevant	Nonrelevant
Retrieved	tp	fp
Not Retrieved	fn	tn

Information Retrieval – Road Ahead



Research Trends and Open Problems

Domain Specific Retrieval, Entity Retrieval, Embeddings, LtR, Term Weighting Schemes, and Index Compression

1. Domain Specific Retrieval

Domain Specific Retrieval

- Music Retrieval
- Multimedia Retrieval
- Legal Document Search
- Local Search
- …

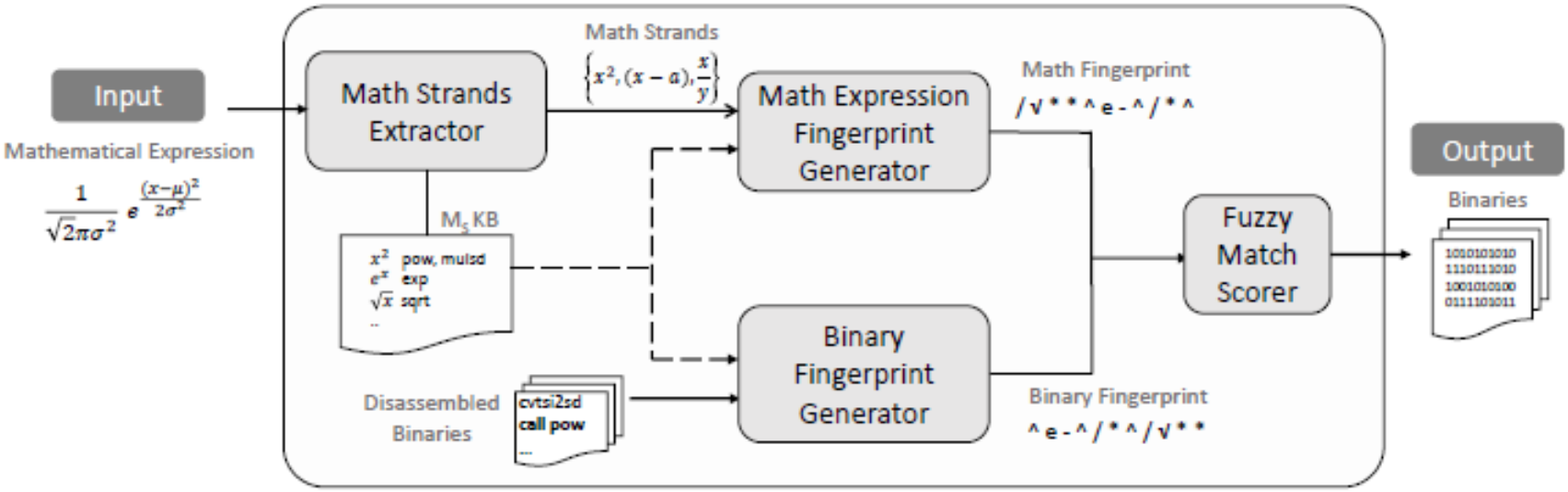
2. Entity Retrieval

Entities over Documents

infobox on google

Entity Retrieval

- Searching for specific type of entities in code.
  - Mathematical expressions
    - On 60 binaries, we show F1 Score = 61%.



  - Can we identify mathematical expressions in source code?
- Can we extend this to other entities (algorithms, data structures)?

Co-authored with Ridhi et al., in MSR ’17.

Publications

- **Gathering and ranking photos of named entities with high precision, high recall, and diversity, WSDM 2010.**
- **Entity linking and retrieval for semantic search, WSDM 2014**
- **ANNE: Improving Source Code Search using Entity Retrieval Approach, WSDM 2017**
- **Identifying Entity Properties from Text with Zero-shot Learning., SIGIR 2019**
- **and many more …**

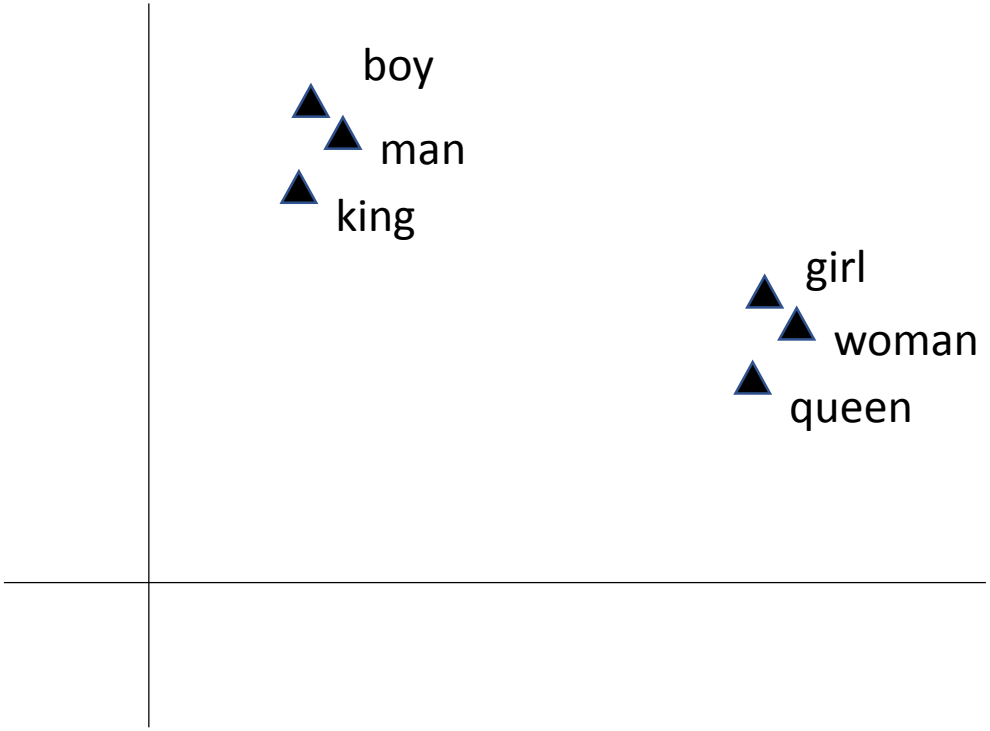
Applications of Entity Retrieval

- Question & Answer Automation
  - Generating automated quiz
  - Automated Evaluation
  - Chatbots
  - …

3. Vector Space Models

word2vec

Embedding words into the vector space



Mikolov et al., 2013.

Word2Vec and Variants

- Word Embeddings
  - Can we identify word relationships?
    - Is female + king = queen?
  - Can we learn to reduce the vocabulary size?
  - word2vec for small corpora
  - word2vec in non-English languages

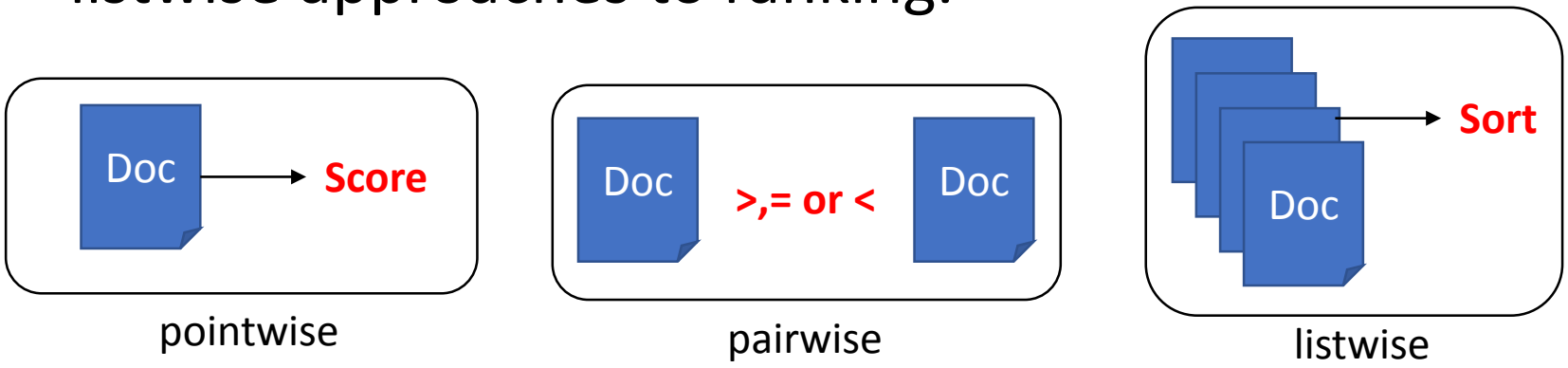
Publications

- **Table2Vec: Neural Word and Entity Embeddings for Table Population and Retrieval, SIGIR 2019.**
- **Network Embedding as Matrix Factorization: Unifying DeepWalk, LINE, PTE, and node2vec, WSDM 2018.**
- **Embedding of Embedding (EOE): Joint Embedding for Coupled Heterogeneous Networks, WSDM 2017.**

4. Learning to Rank

Learning to Rank (LtR)

- Objective: To learn an effective ranking function from a large number of query-document examples.
- Key Challenge: Gold standard datasets.
- Majorly categorized into pointwise, pairwise and listwise approaches to ranking.



Research Trends in LtR

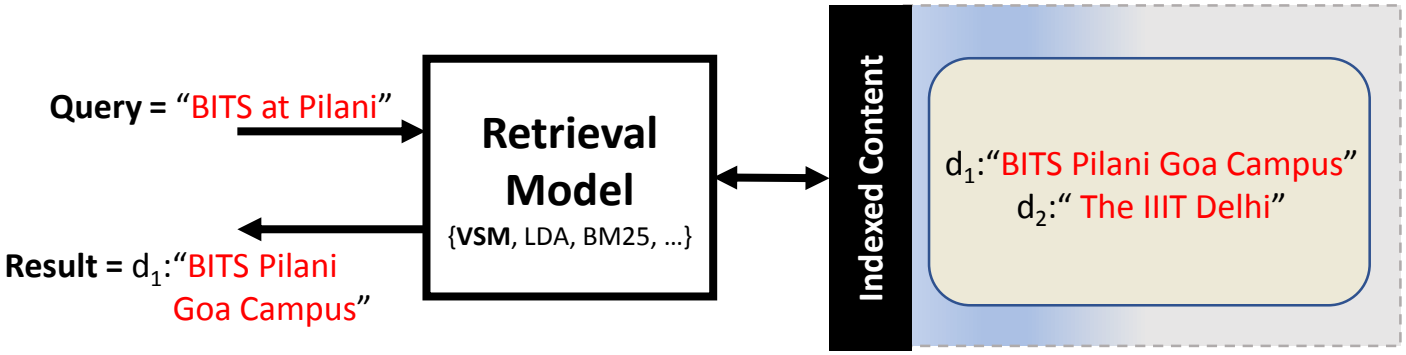
- Preparing gold standard datsets
  - How many queries are required?
  - How many relevant documents per query need to be identified?
  - What should be the ratio of relevant and non-relevant documents per query?
- Extracting and designing features from query-document examples.
- Designing effective and scalable LtR algorithms.
- Rank Aggregation: Merging several ranking lists into a single better ranking list.

Publications

- **A General Framework for Counterfactual Learning-to-Rank, SIGIR 2019.**
- **To Model or to Intervene: A Comparison of Counterfactual and Online Learning to Rank from User Interactions, SIGIR 2019.**
- **Care to Share?: Learning to Rank Personal Photos for Public Sharing, WSDM 2018.**
- **Position Bias Estimation for Unbiased Learning to Rank in Personal Search, WSDM 2018.**

5. Relevance and Ranking

Not Every Word is Important!



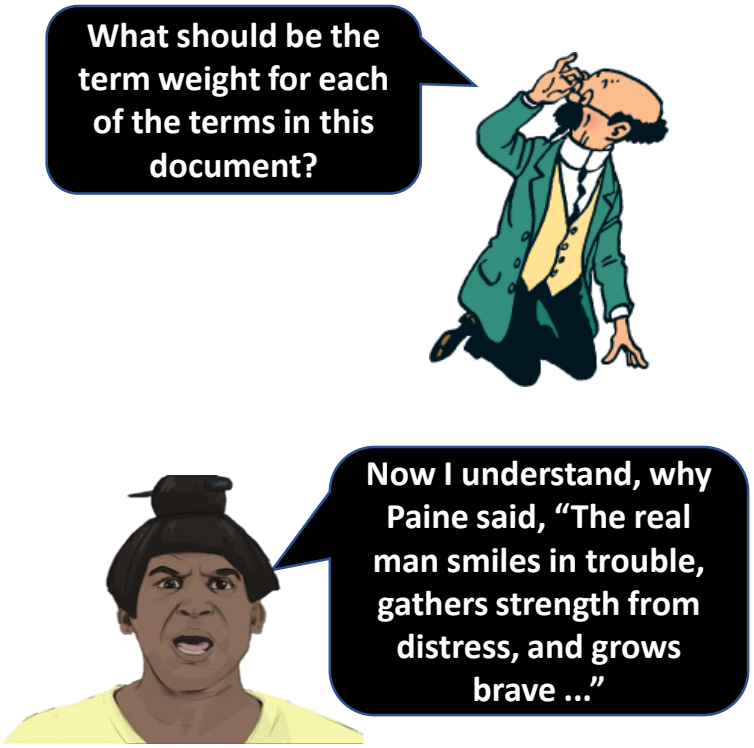
Let us add Term Weights

	BITS	the (* 0)	Pilani	Goa	Campus	IIIT	Delhi
q	1	1 * 0 = 0	1	0	0	0	0
d_1	1	0 * 0 = 0	1	1	1	0	0
d_2	0	1 * 0 = 0	0	0	0	1	1

$sim(q,d_1) = 0.71$

$sim(q,d_2) = 0$

Term Weighting with Inverse Document Frequency



Indexed Content

d_1: *"An inverse document frequency factor is incorporated which diminishes the weight of terms that occur very frequently in the document set and increases the weight of terms that occur rarely."*

d_2: *"A high weight in tf–idf is reached by a high term frequency (in the given document) and a low document frequency of the term in the whole collection of documents; the weights hence tend to filter out common terms "*

Inverse Document Frequency

$$idf(t, D) = \log \frac{N}{|\{d \in D : t \in d\}|}$$

where N = |D| = Total no. of documents.

$$idf(\text{"the"}, \{\mathbf{d_1}, \mathbf{d_2}\}) = \log \frac{2}{2} = 0$$

$$idf(\text{"batsmen"}, \{\mathbf{d_1}, \mathbf{d_2}\}) = \log \frac{2}{1} = 0.3$$

But, do you see any problem? Clue… divide by zero.

Indexed Content

d_1: *"An inverse document frequency factor is incorporated which diminishes **the** weight of terms that occur very frequently in **the** document set and increases **the** weight of terms that occur rarely."*

d_2: *"Sachin Ramesh Tendulkar is a former Indian international cricketer and a former captain of **the** Indian national team, regarded as one of **the** greatest batsmen of all time."*

Variants

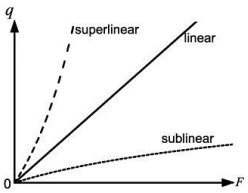
Wikipedia lists the following IDF and TF-IDF variants.

IDF Variants*

weighting scheme	IDF weight ($n_t = \|\{d \in D : t \in d\}\|$)
unary	1
inverse document frequency	$\log \frac{N}{n_t} = -\log \frac{n_t}{N}$
inverse document frequency smooth	$\log\left(1 + \frac{N}{n_t}\right)$
inverse document frequency max	$\log\left(\frac{\max_{\{t' \in d\}} n_{t'}}{1 + n_t}\right)$
probabilistic inverse document frequency	$\log \frac{N - n_t}{n_t}$

TF-IDF Variants*

document term weight	query term weight
$f_{t,d} \cdot \log \frac{N}{n_t}$	$\left(0.5 + 0.5\frac{f_{t,q}}{\max_t f_{t,q}}\right) \cdot \log \frac{N}{n_t}$
$1 + \log f_{t,d}$	$\log(1 + \frac{N}{n_t})$
$(1 + \log f_{t,d}) \cdot \log \frac{N}{n_t}$	$(1 + \log f_{t,q}) \cdot \log \frac{N}{n_t}$



Sub-linear Damping

Add-One Smoothing

Normalization

*See Wikipedia page on TF-IDF (https://en.wikipedia.org/wiki/Tf%E2%80%93idf) for more information.

Publications

- **A novel TF-IDF weighting scheme for effective ranking, SIGIR 2013.**
- **Improving Retrieval Performance for Verbose Queries via Axiomatic Analysis of Term Discrimination Heuristic, SIGIR 2017.**
- Term Weighting Schemes for Latent Dirichlet Allocation, ACL 2000.
- **Proposing a new term weighting scheme for text categorization, AAAI 2006**

6. Query Refinement

Global (User/Result-Independent) Query Refinement

- Automatic Thesaurus Generation
  - Fast = rapid
  - Tall = height?
  - Sound = noise?
  - Restaurant = Hotel = Motel?
- How to handle domain specific phrases?
- Slangs!
- …

How to automate the thesaurus generation?

Co-occurrence Analysis

MAINFRAMES

Mainframes are primarily referred to large computers with rapid, advanced processing capabilities that can execute and perform tasks equivalent to many Personal Computers (PCs) machines networked together. It is characterized with high quantity Random Access Memory (RAM), very large secondary storage devices, and high-speed processors to cater for the needs of the computers under its service.

Consisting of advanced components, mainframes have the capability of

MAINFRAMES

Mainframes usually are referred those computers with fast, advanced processing capabilities that could perform by itself tasks that may require a lot of Personal Computers (PC) Machines. Usually mainframes would have lots of RAMs, very large secondary storage devices, and very fast processors to cater for the needs of those computers under its service.

Due to the advanced components mainframes have, these computers

Source: https://web.stanford.edu/~jurafsky

Co-occurrence Analysis

- Term-Document Matrix
  - How often does individual terms appear in a document?
- Term-Term Matrix
  - How often terms co-occur?

Quiz: Which books are similar?

	Book1	Book2	Book3	Book4
cricket	400	10	355	3
football	5	5	4	4
hockey	9	330	10	200
tennis	2	6	12	4

Co-occurrence Analysis

- Two documents are similar if the document vectors are similar.

	Book1	Book2	Book3	Book4
cricket	400	10	355	3
football	5	5	4	4
hockey	9	330	10	200
tennis	2	6	12	4

Book1 and Book3 seem to be on cricket.
Book2 and Book4 are about hockey.

Co-occurrence Analysis

- Two terms are similar if the term vectors are similar.

	Book1	Book2	Book3	Book4
boundary	400	310	355	389
four	515	225	390	400
movie	9	4	8	1
film	2	6	9	2

Of course, context is important!

Magic with Matrices

Transpose

- If A is as given below, what is A^T?

$$\mathbf{A} = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix} \qquad \mathbf{A^T} = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix}$$

Co-occurrence Analysis

- Assume a Boolean term-document matrix A.
- What does AA^T mean?

$$
\begin{array}{c} \\ t_1 \\ t_2 \\ t_3 \\ t_4 \end{array}
\begin{array}{c}
\begin{array}{cccc} d_1 & d_2 & d_3 & d_4 \end{array} \\
\begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix}
\end{array}
\begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix}
=
\begin{array}{c}
\begin{array}{cccc} t_1 & t_2 & t_3 & t_4 \end{array} \\
\begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}
\end{array}
\begin{array}{c} \\ t_1 \\ t_2 \\ t_3 \\ t_4 \end{array}
$$

Publications

- **Merchandise Recommendation for Retail Events with Word Embedding Weighted Tf-idf and Dynamic Query Expansion, SIGIR 2018.**
- **Learning Parametric Models for Context-Aware Query Auto-Completion via Hawkes Processes, WSDM 2017.**
- **Retrieval Consistency in the Presence of Query Variations, SIGIR 2017.**
- **Intent-Aware Semantic Query Annotation, SIGIR 2017.**

7. Index Compression

Google Data Centers

- Contain commodity machines distributed around the world.
- Estimate: 0.9 million servers = 3 million processors/cores (Gartner 2007)
- Estimate: Google installs 100,000 servers each quarter.
  - Based on expenditures of 200–250 million dollars per year
- This would be 10% of the computing capacity of the world!?!

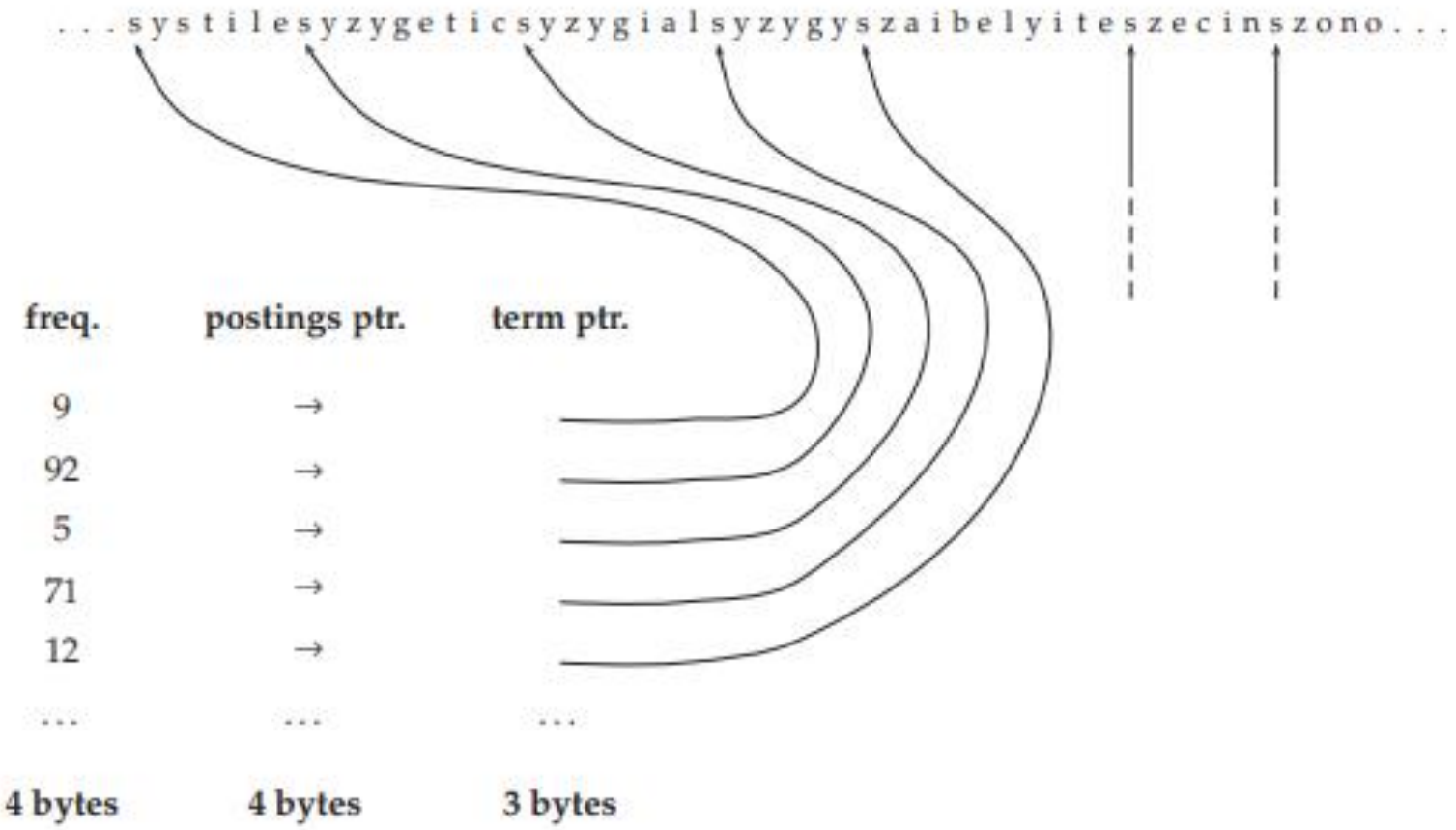
Sources: cecs.wright.edu/~tkprasad/, DataCenterKnowledge

Dictionary as a Sorted Array

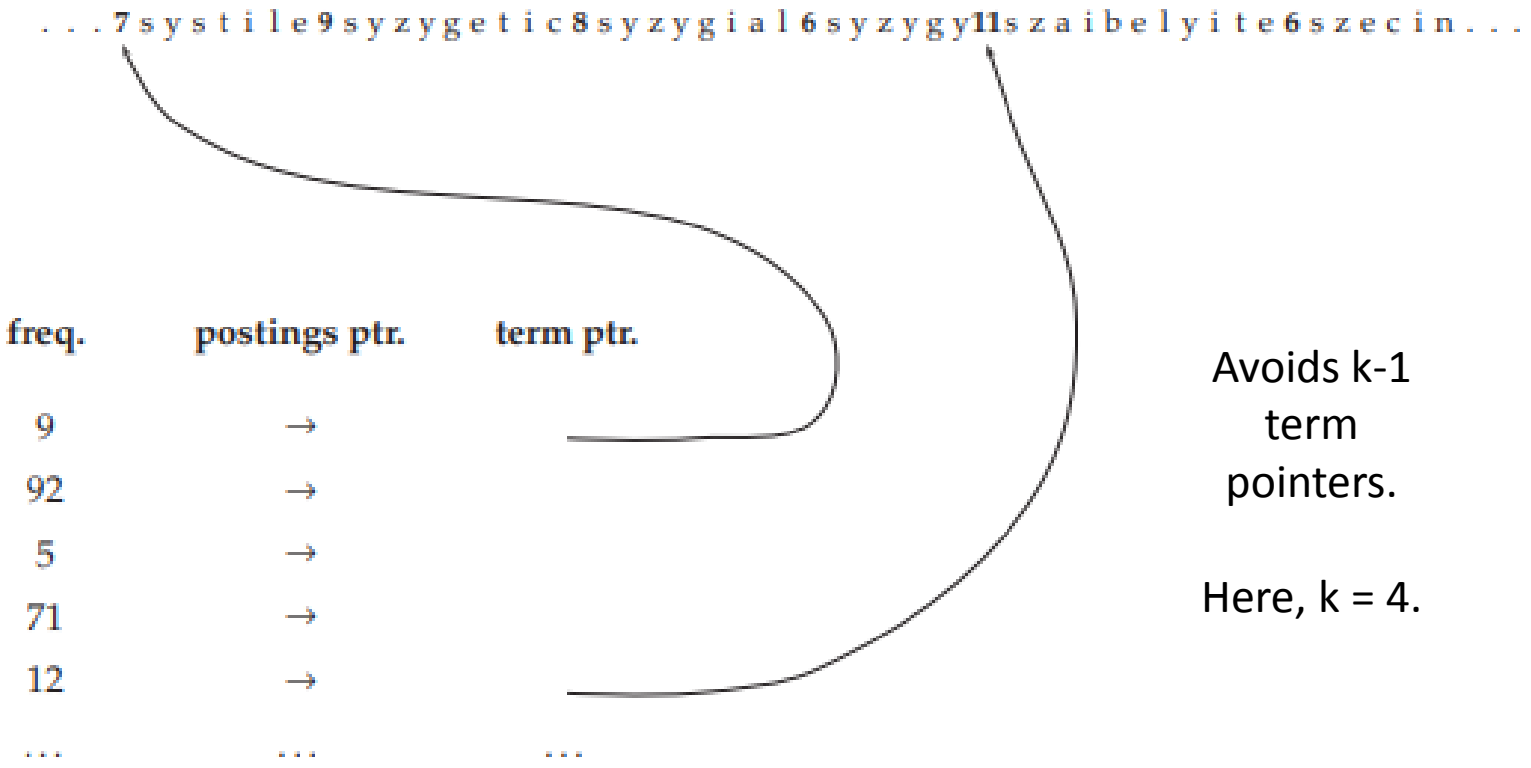
term	document frequency	pointer to postings list
a	656,265	→
aachen	65	→
…	…	…
zulu	221	→
20 bytes	4 bytes	4 bytes

Apply binary search to search the term array!

Dictionary as a String



Blocked Storage



Avoids k-1 term pointers.

Here, k = 4.

Front Coding

One block in blocked compression ($k = 4$) . . .
8automata8automate9automatic10automation

. . . further compressed with front coding.
8automat∗a1⋄e2 ⋄ ic3⋄ion

Can you front-code at k = 3, "interspecies","interstellar","interstate"?

Publications

- **Fast Dictionary-Based Compression for Inverted Indexes, WSDM 2019.**
- **Index Compression for BitFunnel Query Processing, SIGIR 2018.**
- **Index Compression Using Byte-Aligned ANS Coding and Two-Dimensional Contexts, WSDM 2018.**
- **Index compression is good, especially for random access, CIKM 2007.**

Summary

- Mathematical models play a key role in building large software systems.
- Vector space models act as good abstraction for information retrieval.
- Research groups on IR are highly active in various areas. Particularly, we discussed:
  - Domain Specific Retrieval, Entity Retrieval, Learning to Rank and Index Management.

Thank You

If you're changing the world, you're working on important things. You're excited to get up in the morning. – Larry Page.