



# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

## Computing Scores in a Complete Search System

---

Bhaskarjyoti Das

Department of Computer Science Engineering

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

## Efficient Scoring and Ranking : the Idea

**Bhaskarjyoti Das**

Department of Computer Science Engineering

## Recap : Vector Space Scoring

---

### ▪Vector space scoring

- Represent **queries as vectors** in the **document space**
- Terms being considered are the “**terms present in query**”
- We have a tf-idf **weight for each term (present in query)** in **each doc**
- The tf-idf score of a document is **sum of tf-idf score for every term (present in query)**
- Length normalization : **tf-idf document vector -> unit vector**
- Rank documents according to their **cosine similarity** to the **query vector in this space**

### ▪Vector space scoring is:

- Entirely **query dependent**
- Additive on term contributions** – no conditionals etc.
- Context insensitive** (general issue with bag of words)

## TAAT vs. DAAT Technique

---

### ■ TAAT = “Term At A Time”

- tf-idf scores for all docs computed concurrently
- **one query term at a time**
- We will have partial scores
- Sum over all terms

### ■ DAAT = “Document At A Time”

- tf-idf score for each doc
- **One document at a time**
- For each document, we have the full score

## The Problem Statement

---

- Score computation is a large (10s of %) fraction of the CPU work on a query
  - Generally, we have a tight budget on latency (how fast the contents within a pipe can be transferred from the client to the server and back. Say, 250ms)
  - CPU provisioning doesn't permit scoring every document on every query
- Today we'll look at ways of cutting CPU usage for scoring, without compromising the quality of results (much)
- Basic idea: avoid scoring docs that won't make it into the top  $K$

## Safe vs. Non-safe Ranking is the Question

---

- What is **safe ranking**?
- What is **non-safe ranking**?
- Is it ok to be **non-safe**?
  - If it is – then how do we ensure **we don't get too far** from the safe solution?
  - How do we measure if we are far?

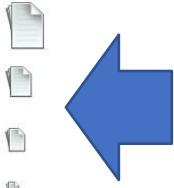
## Safe Ranking ?

---

- The terminology “safe ranking” is used for methods that guarantee that the  $K$  docs returned are the  $K$  absolute highest scoring documents
- When we output the top  $K$  docs, we have a proof that these are indeed the top  $K$

## Non-safe Ranking ?

- Non-safe ranking may be okay
  - Ranking function is only a proxy for user happiness
  - Still approximate at best !
  - Documents close to top K may be just fine



Retrieved Documents

Ranking Function  
as a proxy

User Happiness

## Generic Approach

---

- Find a **set  $A$  of *contenders***, with  $K < |A| \ll N$
- $A$  **does not necessarily contain the top  $K$ , but has many** docs from among the top  $K$
- Return the top  $K$  docs in  $A$
- Think of  $A$  as **pruning non-contenders**
- Will look at several schemes following this approach

## Optimization by Heuristics

---

- Essentially **Optimization by heuristics** is the way to go
- We will study several such **clever optimization methods**
- Point to note
  - These **optimization methods are mostly heuristics.**
  - We can use one or more **of them together**
  - **Not a formulae** for sure shot success
  - Very **similar to “normalization techniques”** in **NLP** where you can use several of them!



**PES**  
UNIVERSITY

THANK  
YOU

---

**Bhaskarjyoti Das**  
Department of Computer Science Engineering



**PES**  
UNIVERSITY

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

## Computing Scores in a Complete Search System

---

**Bhaskarjyoti Das**

Department of Computer Science Engineering

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

## Index Elimination

**Bhaskarjyoti Das**

Department of Computer Science Engineering

## Index Elimination

---

- Basic cosine computation algorithm only considers docs containing at least one query term
- We always need not compute all  $N$  cosines
  - So we only fully score some  $A$  documents saving CPU cycles when  $A < N$
- Take this further:
  - Strategy 1 : Only consider high-idf query terms
  - Strategy 2 : Only consider docs containing many query terms

## Only High-idf Query Terms

---

- For a query such as *catcher in the rye*
- Only accumulate scores from *catcher* and *rye*
- Intuition: *in* and *the* contribute little to the scores ( tf will be high but idf will offset that) and so they don't alter rank-ordering much
- Benefit:
  - many docs get **eliminated** from **set A of contenders**

## Only Docs Containing Many Query Terms

---

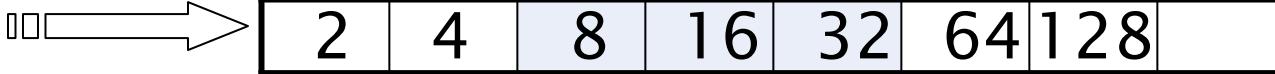
- **Basic algorithm:** Any doc with **at least one query term** is a **candidate** for the **top  $K$  output list**
- **Heuristics :** For multi-term queries, only compute scores for docs **containing several of the query terms**
  - Say, at least 3 out of 4
  - Imposes a “**soft conjunction**” (**implicit conjunction**) on queries seen on web search engines (early Google)
- Benefit:
  - many docs get **eliminated** from **set A of contenders**

## 3 of 4 Query Terms

**Antony**



**Brutus**



**Caesar**



**Calpurnia**



Scores only computed for docs 8, 16 and 32.



# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

## Champion Lists

**Bhaskarjyoti Das**

Department of Computer Science Engineering

## Champion Lists

---

- Precompute **for each dictionary term  $t$ , the  $r$  docs of highest weight in  $t$ 's postings**
  - Call this the **champion list for  $t$**
  - (aka fancy list or top docs for  $t$ )
- Note that  $r$  has to be chosen at index build time
  - **Thus, it is possible that  $r < K$**
- At query time, **only compute scores for docs in the champion list of some query term**
  - Pick the  $K$  top-scoring docs from amongst these

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

## Efficient Cosine Ranking

**Bhaskarjyoti Das**

Department of Computer Science Engineering

## Cosine Similarity is Only a Proxy

---

- User has a **task in mind** -> a **query formulation**
- Cosine **matches docs to query**
- Thus **cosine is anyway a proxy** for **user happiness**
- If we get a list of  **$K$  docs “close” to the top  $K$**  by cosine measure, should be ok

## Efficient Cosine Ranking

---

- Find the  $K$  docs in the collection “nearest” to the query  $\Rightarrow K$  largest query-doc cosines
- The idea: not to compute all the cosines
- Efficient ranking:
  - Computing a single cosine efficiently.
  - Choosing the  $K$  largest cosine values efficiently



**PES**  
UNIVERSITY

THANK  
YOU

---

**Bhaskarjyoti Das**  
Department of Computer Science Engineering



**PES**  
UNIVERSITY

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

## Computing Scores in a Complete Search System

---

**Bhaskarjyoti Das**

Department of Computer Science Engineering

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

## Efficient Cosine Ranking Part 2

**Bhaskarjyoti Das**

Department of Computer Science Engineering

## Computing Single Cosine Efficiently : Efficient Cosine Ranking

---

- What we're doing in basic algorithm  $\Rightarrow$  solving the **K-nearest neighbor problem** for a query vector
- In general, we do not know how to do this **efficiently for high-dimensional spaces**
- But it is **solvable for short queries (=low dimension)**
- **Standard indexes support this well**
  - computing for only those terms present in the query

## Computing Single Cosine Efficiently : Unweighted Queries

---

- No weighting on query terms
- All document vectors are unit vector whose tips lie on a hypersurface of same radius
- Then for ranking, don't need to normalize query vector
- Simplification of algorithm
- But may not be acceptable

## Computing K Largest Cosines: Selection vs. Sorting

---

- Typically we want to retrieve the **top  $K$  docs** (in the **cosine ranking for the query**)
- We need to calculate the **cosines**
- We need to **sort** on calculated cosine values
- Can we replace sort with selection ?

## Computing K Largest Cosines: Using Heap for Selecting Top K

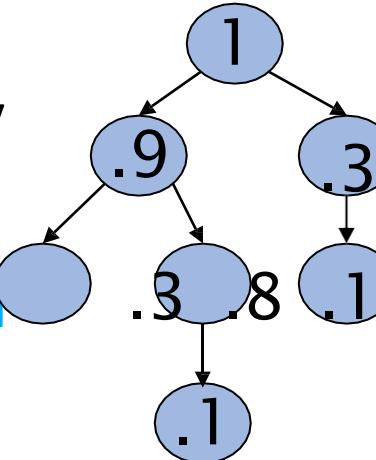
- Maxheap is a complete Binary tree in which **each node's value > the values of children**. **Root will have the max value**. Typically implemented as array where the root element will be at Arr[0].

- **Retrieving the root is O(1) operation**. Can **build a maxheap of n elements** in linear time **O(n)**.

- **Insert or delete one element** will take **O(log n)**. For top **K cosines**, we can successively do **K delete** operations i.e. "**K log n**" time

- Building a **maxheap of n cosines** and **retrieving K** will be **O(n+K log n)**

- For  $n=1M$ ,  $K=100$ , this is **about 10% of the cost of sorting**.



# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

## Static Quality Scores

**Bhaskarjyoti Das**

Department of Computer Science Engineering

## Static Quality Scores : Authority

---

- **Relevance** is being modeled by **cosine scores**
- **Authority** is typically a **query-independent property** of a document
- We want top-ranking documents to be **both** *relevant and authoritative*
- Examples of authority signals
  - Wikipedia among websites
  - Articles in certain newspapers
  - A paper with many citations
  - Many bitly's, diggs or del.icio.us marks
  - (Pagerank)

Quantitative

## Modelling Authority

---

- Assign to each document a *query-independent quality score* in [0,1] to each document  $d$ 
  - Denote this by  $g(d)$  i.e. *goodness score*
  - *This is a static quality score ( does not change with change in query)*
- Thus, a quantity like the **number of citations** is scaled into [0,1]

## Net Score

---

- Consider a **simple total score combining cosine relevance and authority**
- $\text{net-score}(q,d) = g(d) + \cosine(q,d)$ 
  - Can use **some other linear combination**
  - Any function of the **two “signals” of user happiness**
- Now we seek the top  $K$  docs by **net score**

## Top K by Net Score

---

- Idea:
  - Originally, posting list **ordered by docid**
  - Now, **order all postings by  $g(d)$**
  - *Intersect algorithm requires posting list ordered by a common ordering !*
- **Key: now  $g(d)$  is the common ordering for all postings**
- Thus, we can still concurrently traverse query terms' postings for
  - Postings intersection
  - Cosine score computation

## Why Order Posting by $g(d)$ ?

---

- Under  $g(d)$ -ordering top-scoring docs likely to appear early in postings traversal
- In time-bound applications (say, we have to return whatever search results we can in 50 ms), this allows us to stop postings traversal early short of computing scores for all docs in postings

## Extend Champion Lists with $g(d)$ -Ordering

---

- Can combine champion lists with  $g(d)$ -ordering
  - Maintain for each term a champion list of the  $r$  docs with highest  $g(d) + \text{tf-idf}_{td}$
  - Seek top- $K$  results from only the docs in these  $g(d)$  ordered champion lists



THANK  
YOU

---

**Bhaskarjyoti Das**  
Department of Computer Science Engineering



**PES**  
**UNIVERSITY**

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

## Computing Scores in a Complete Search System

---

**Bhaskarjyoti Das**

Department of Computer Science Engineering

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

**Impact Ordering, High Low Lists,  
Cluster Pruning**

**Bhaskarjyoti Das**  
Department of Computer Science Engineering

## Impact Ordered Postings

---

- We **do not order posting list by docid or even  $g(d)$  score**
- **Instead we sort each postings list by tf-idf weight  $wf_{t,d}$**
- We only want to compute scores for docs for which  $wf_{t,d}$  is high enough (impactful docids in a term specific manner)
- We still follow the same algorithm but suitably modify for storing the partial scores for each doc as not all postings are in a common order!
- We can even stop after computing only K scores along the posting lists for each term
- **How do we compute scores in order to pick off top K?**
  - Two ideas follow

## Impact Ordered Postings : Early terminations

---

- When traversing  $t$ 's postings, **stop early** after either
  - a **fixed number of  $r$  docs**
  - $wf_{t,d}$  drops **below some threshold**
- Take the **union of the resulting sets of docs**
  - One from the postings of each query term
- **Compute only the scores for docs in this union**

## Impact Ordered Postings : idf Ordered Items

---

- When considering the postings of query terms
- Look at them in **order of decreasing idf**
  - **High idf terms likely to contribute most to score**
- As **we update score contribution** from each query term
  - **Stop if doc scores relatively unchanged**
- Can apply to cosine or some other net scores

## High and Low List ?

---

- For each term, we maintain two postings lists called *high* and *low*
  - Think of *high* as the **champion list**
  - When traversing postings on a query, only traverse *high* lists first
    - If we get more than  $K$  docs, select the top  $K$  and stop
    - Else proceed to get docs from **the low lists**
- This strategy can be used even for simple cosine scores, without global quality  $g(d)$
- Essentially **segmenting index** into **two tiers**

## Cluster Pruning : Pre Processing

---

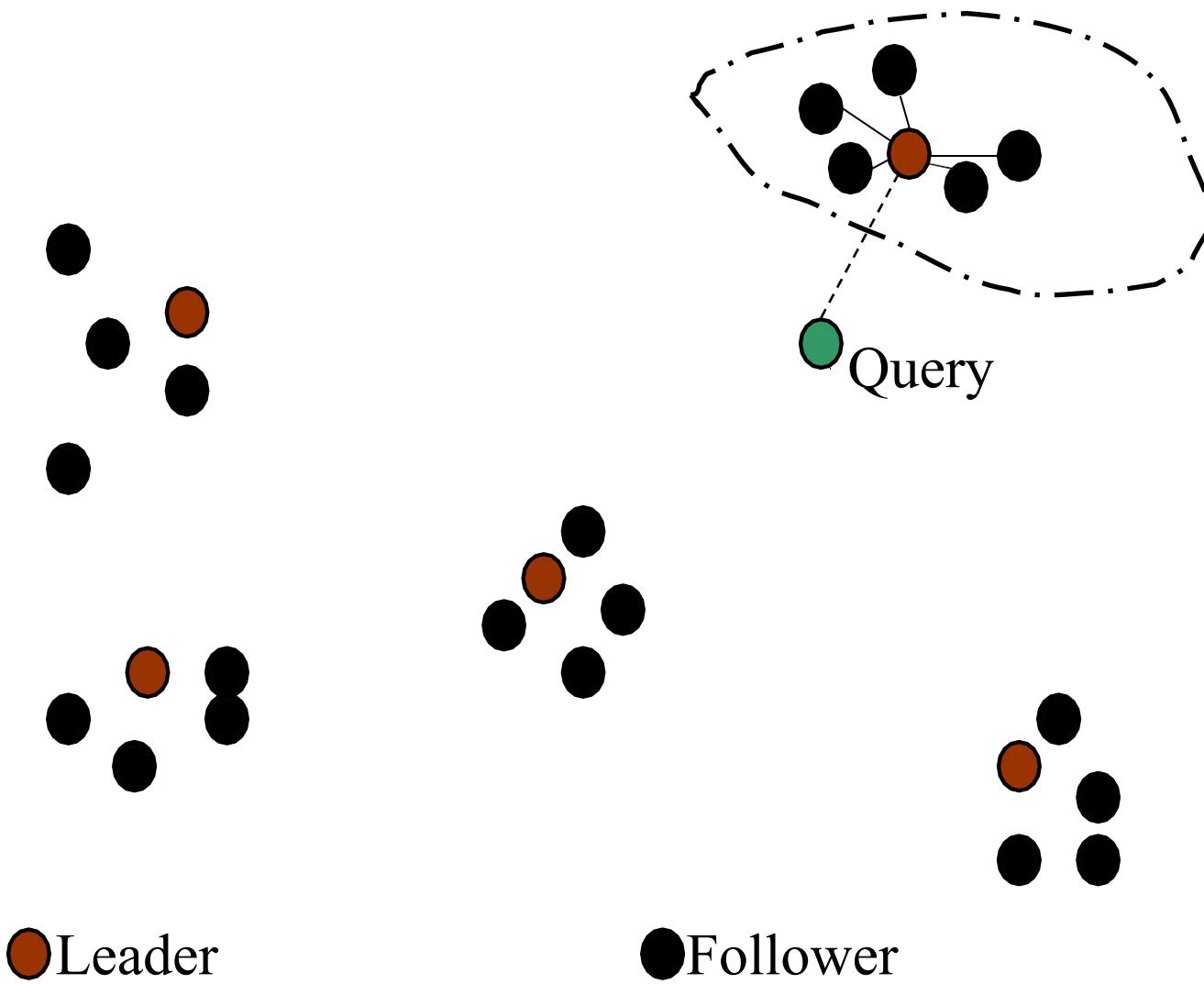
- Pick  $\sqrt{N}$  docs at random: call these *leaders*
- Cluster the other documents around the leaders : For every other doc, pre-compute nearest leader
- Docs attached to a leader: its *followers*;
- Likely: each leader has  $\sim \sqrt{N}$  followers.

## Cluster Pruning : Query Processing

---

- Process a query as follows:
  - Given query  $Q$ , find its nearest leader  $L$ .
  - Seek  $K$  nearest docs from among  $L$ 's followers.

## Visualization



## Why Random Sampling ?

---

By choosing the leaders at random

1. Fast to choose  $\sqrt{N}$  random numbers
2. Leaders chosen this way will be very likely to reflect data distribution



## General Variants of the Previous Method

---

- Bring more “fuzziness” into the method
  - Have **each follower** attached to  **$b_1=3$  (say) nearest leaders.**
  - From **query, find  $b_2=4$  (say) nearest leaders**
- Since each follower can attach to  $b_1$  leaders, we are now looking at **a larger set of followers.**
- Can **recurse** on leader/follower construction.



**PES**  
**UNIVERSITY**

**THANK  
YOU**

---

**Bhaskarjyoti Das**  
Department of Computer Science Engineering



**PES**  
UNIVERSITY

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

## Computing Scores in a Complete Search System

---

**Bhaskarjyoti Das**

Department of Computer Science Engineering

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

**Parametric and Zone Indexes, Tiered  
Indexes, Query Term Proximity, Query  
Parser, Aggregating Scores**

**Bhaskarjyoti Das**  
Department of Computer Science Engineering

## Parametric and Zone Indexes

---

- Thus far, a doc has been a sequence of terms
- In fact documents have multiple parts, some with special semantics:
  - Author
  - Title
  - Date of publication
  - Language
  - Format
  - etc.
- These constitute the metadata about a document

## Fields

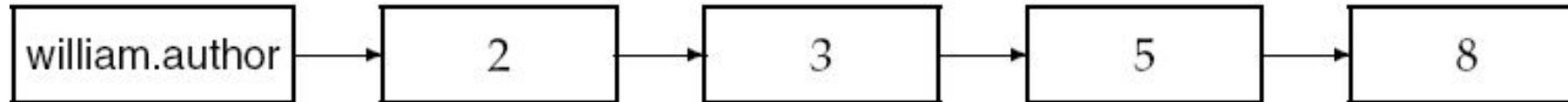
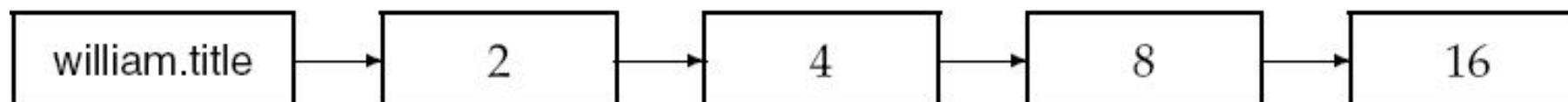
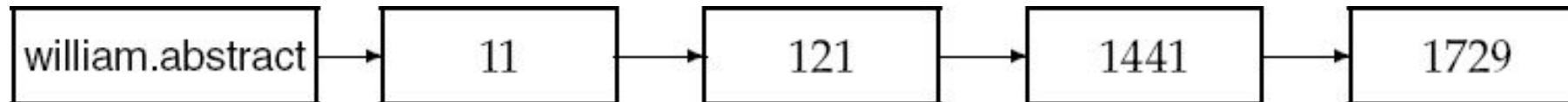
---

- We sometimes wish to search by these metadata
  - Example: *find docs authored by William Shakespeare in the year 1601, containing alas poor Yorick*
  - **Year = 1601 is an example of a field**
  - Also, author **last name** = shakespeare etc.
  - **Field or parametric index:** postings for each field value
  - **Field query** typically **treated as conjunction**
    - (*doc must be authored by shakespeare*)

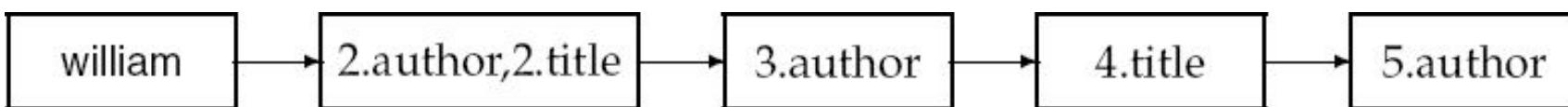
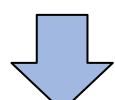
- A zone is a region of the doc that can contain an arbitrary amount of text, e.g.,
  - Title
  - Abstract
  - References ...
- Build inverted indexes on zones as well to permit querying
- E.g., “find docs with *merchant* in the **title zone** and matching the query *gentle rain*”

## Encode Zones in Dictionary vs. Postings

single index but zone specific terms in dictionary



single term but posting contains only the doc id but also the zone information



## Tiered Index : High and Low List

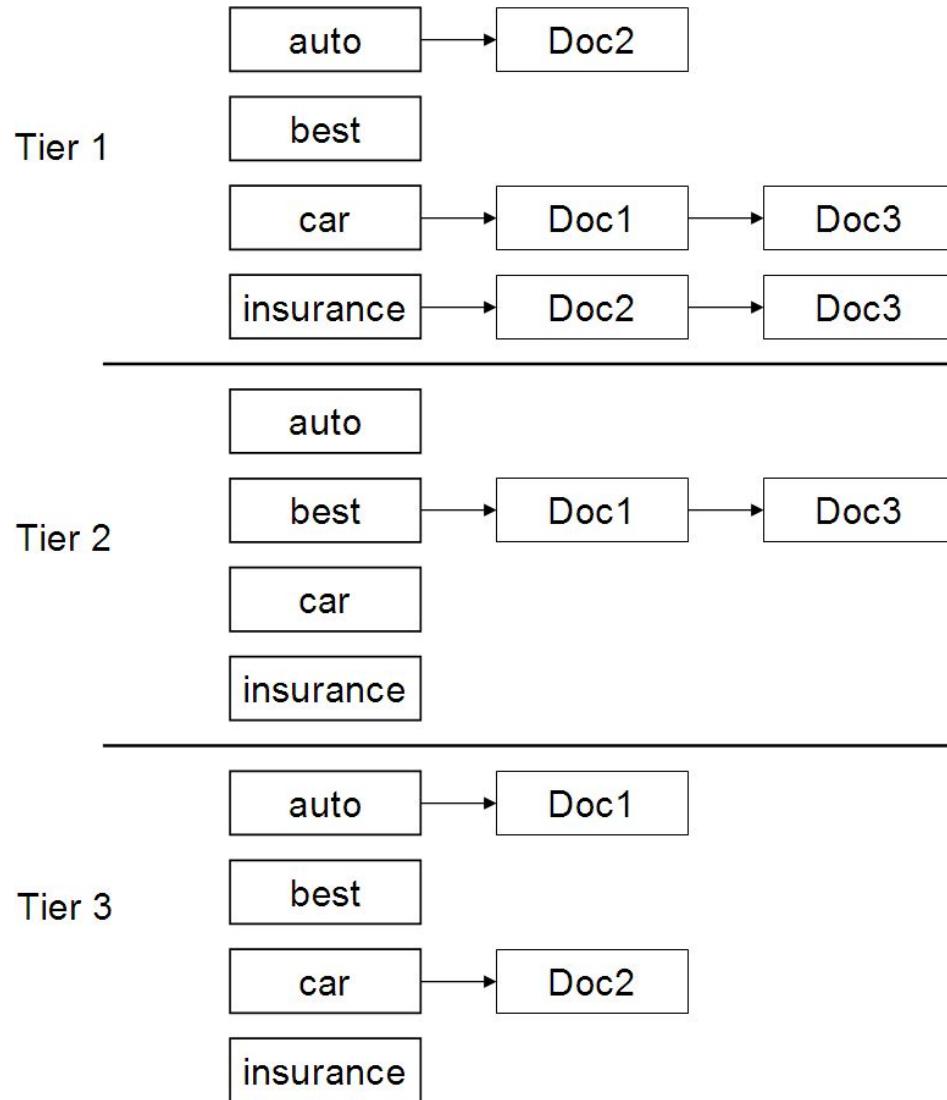
---

- Break postings up into a hierarchy of lists
  - Most important
  - ...
  - Least important
- **Can be done by  $g(d)$  or another measure**
- Inverted index thus broken up into tiers of decreasing importance
- **At query time use top tier unless it fails to yield  $K$  docs**
  - If so drop to lower tiers

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT

WEB

## Example : Tiered Index



## Query Term Proximity

---

- **Free text queries:** just a set of terms typed into the query box
  - common on the web
- *Users prefer docs in which **query terms occur within close proximity of each other***
- Let **w be the smallest window** in a doc containing all query terms, e.g.,
- For the query **strained mercy** the smallest window in the doc ***The quality of mercy is not strained* is 4 (words)**
- Clearly we need a **positional index** to implement a **constraint like query term proximity**

## Why Query Parsers ?

---

- Free text query from user **may in fact spawn one or more queries to the indexes**, e.g., query *rising interest rates*

1. Run the query as a phrase query
2. If  $<K$  docs are returned containing the phrase *rising interest rates*, run the two phrase queries *rising interest* and *interest rates*
3. If we still have  $<K$  docs, run the vector space query *rising interest rates*
4. Merge the three sets
5. Rank matching docs by vector space scoring

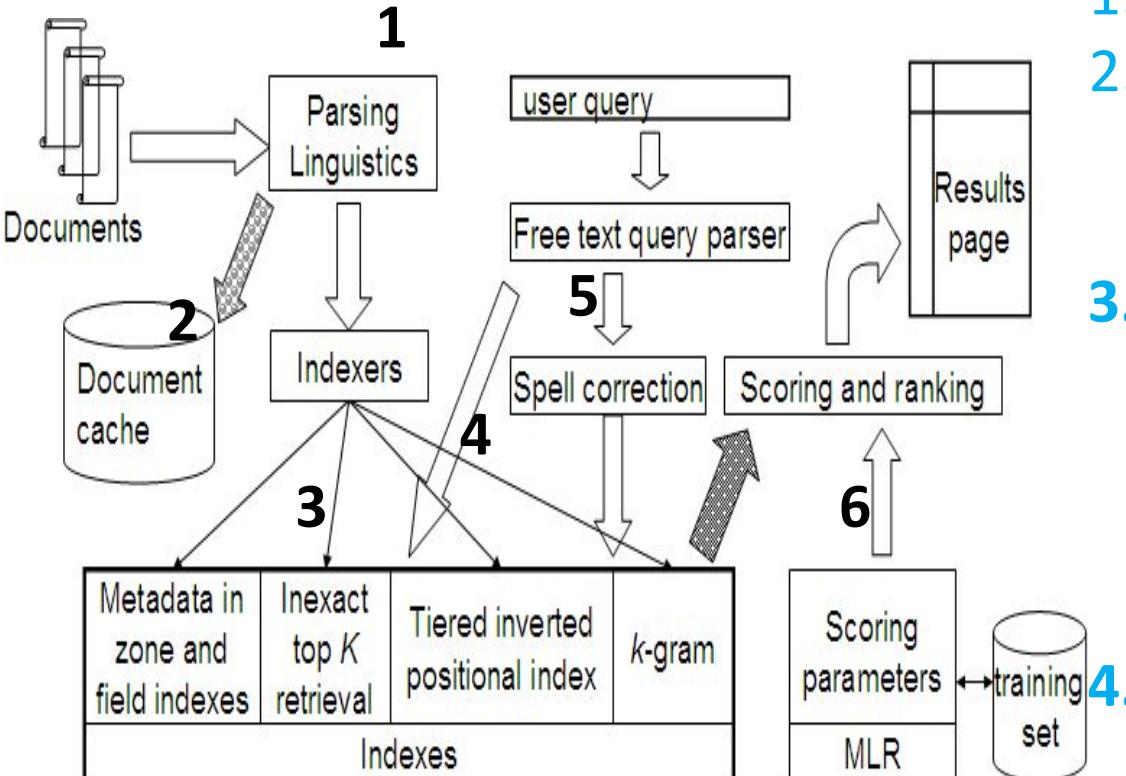
This sequence is issued by a query parser

## Aggregate Scores

---

- Score functions can combine multiple scores together cosine, static quality, proximity, etc.
- How do we know the best combination?
  - Applications like enterprise search ( static corpus)
    - expert-tuned
  - Increasingly common
    - Supervised machine-learning based

## Components of IR System



1. Tokenization and normalization
2. Processed output of step 1 in cache for **generating snippets** used in retrieval results
3. **Multiple indexes** are built (can be more than Inexact ~~fօəns~~ optimized and k-gram index is useful for spelling correction) in parallel
4. **Query parser** submits **query to the indexes**
5. Query parser also submits **query after spelling correction**
6. **Final score is aggregated using ML**



**PES**  
**UNIVERSITY**

**THANK  
YOU**

---

**Bhaskarjyoti Das**  
Department of Computer Science Engineering



# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

## Evaluation And Evaluation Benchmark

---

**Bhaskarjyoti Das**  
Department of Computer Science  
Engineering

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

## Evaluation in Information Retrieval

**Bhaskarjyoti Das**  
Department of Computer Science Engineering

## Quantitative Measures for a search engine

---

- How **fast** does it **index**
  - number of bytes per hour
  - Distributed indexing will do better ?
- How **fast** does it **search**
  - latency as a function of queries per second
  - Query Parser submits to normalindex as well as spelling correction mechanisms
    - one after another (saves capacity at cost of time) or together (saves time at cost of capacity)
- What is the **cost per query**?
  - in dollars (Engineering cost vs. Revenue of search)

## Qualitative Measures for a Search Engine is User Happiness

---

- However, the key measure for a search engine is **user happiness**
- What is **user happiness**?
- **Factors** include:
  - Speed of response (function of size of index)
  - Uncluttered UI
  - Most important: **relevance** ( we will focus on this )

## Need a Way to Quantify User Happiness

---

- Why focus on **qualitative measure** ?
- Even if a search is “free” and “blindingly fast”, it will not deliver user happiness if **useless answers** are served !
- How can we quantify user happiness?

## Who is the User in Web Search ?

---

- Who is the **user** we are trying to make happy?
- Web search engine: **searcher**.
- Success: Searcher finds what she was looking for. Measure: rate of return to this search engine
- Web search engine: **advertiser**.
- Success: Searcher clicks on ad. Measure: clickthrough rate

If the searcher (**user**) return rate is low, advertiser is also unhappy !

## Who is the User in Enterprise Search ?

---

- Ecommerce: **buyer**. Success: Buyer buys  
Measures: time to purchase, fraction something.  
“conversions” of searchers to buyers
- Ecommerce: **seller**. Success: Seller sells something.  
Measure: profit per item sold
- Enterprise: **CEO**. Success: Employees are more productive (because of effective search).  
■ Measure: profit of the company

If the **buyer** does not convert recommendations to buy decision, both seller and CEO will be unhappy

## Most common definition of user happiness: Relevance

---

- User happiness is equated with the relevance of search results to the query.
- But how do you measure relevance?
- Standard methodology in information retrieval consists of three elements.
  - A benchmark document collection
  - A benchmark suite of queries
  - An assessment of the relevance of each query-document pair

## Relevance: query vs. information need

---

- Relevance to **what?**
- First take: relevance to the query
- “Relevance to the query” is very **problematic**.
- **Information need** : “I am looking for information on whether drinking red wine is more effective at reducing your risk of heart attacks than white wine.”

## Relevance: query vs. information need

---

- This is an **information need, not a query.**
- Query  $q$ : [red wine white wine heart attack]
- Consider document  $d'$ : *At heart of his speech was an attack on the wine industry lobby for downplaying the role of red and white wine in drunk driving.*
- $d'$  is an excellent match for query  $q$ !
- $d'$  is **not** relevant to the information need  $i$ .

## Relevance: query vs. information need

---

- User happiness can only be measured by relevance to an information need, not by relevance to queries.

- Our terminology is sloppy in these slides and in IIR !

We talk about **query-document relevance judgments** even though we mean **information-need-document relevance judgments**.



THANK  
YOU

**Bhaskarjyoti Das**  
Department of Computer Science Engineering



**PES**  
**UNIVERSITY**

# **ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB**

## **Evaluation And Evaluation Benchmark**

---

**Bhaskarjyoti Das**  
Department of Computer Science  
Engineering

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

## Evaluation Benchmark

**Bhaskarjyoti Das**

Department of Computer Science  
Engineering

## Relevance

---

With respect to a user information need, a document in the test collection is given a **binary classification** as either **relevant** or **nonrelevant**.



## What we need for a benchmark

---

1.A benchmark document collection

2.A benchmark test suite of information needs, expressed as queries

3.A set of relevance judgments, standardly a binary assessment of either relevant or nonrelevant for each query- document pair.

-

## What we need for a benchmark

---

- Human relevance assessments

- We need to hire/pay “judges” or assessors to do this.
- Expensive, time-consuming
- Judges must be representative of the users we expect to see in reality.

## Gold Standard Ground Truth

---

- The test document collection and suite of information needs have to be of a **reasonable size**
- Need to **average performance over fairly large test sets** as results are **highly variable** over different documents and information needs.
- As a rule of thumb, **50 information needs** has usually been found to be a **sufficient minimum**.

Many systems contain various **weights** that can be **adjusted** to tune system performance.

Wrong to report results on a test collection which were obtained by tuning parameters to maximize performance on that collection.

That is because such tuning overstates the expected performance of the system, because the **weights** will be **set** to maximize performance on one particular set of queries rather than for a random sample of queries.

## Development Test Collection

---

The correct procedure is to have one or more development test collections, and to tune the parameters on the development test collection

The tester then runs the system with those weights on the test collection and reports the results on that collection as an unbiased estimate of performance.

## Standard relevance benchmark: Cranfield

---

### Cranfield experiments

- Pioneer work and foundation in IR evaluation
- Basic hypothesis
  - Retrieved documents' relevance is a good proxy of a system's utility in satisfying users' information need
- Procedure
  - 1,398 abstracts of aerodynamics journal articles
  - 225 queries
  - Exhaustive relevance judgments of all (query, document) pairs
  - Compare different indexing system over such collection



## Standard relevance benchmarks: TREC

---



- TREC = Text Retrieval Conference (TREC)
- Organized by the U.S. National Institute of Standards and Technology (NIST)
- TREC is actually a set of several different relevance benchmarks.
- Best known: TREC Ad Hoc, used for first 8 TREC evaluations between 1992 and 1999

## Standard relevance benchmarks: TREC

---



- 1.89 million documents, mainly newswire articles, 450 information needs
- No exhaustive relevance judgments – too expensive
- Rather, NIST assessors' relevance judgments are available only for the documents that were among the top k returned for some system which was entered in the TREC evaluation for which the information need was developed.

## Standard relevance benchmarks: Others

---

- **GOV2**

- Another TREC/NIST collection
- 25 million [web pages](#)
- Used to be largest collection that is easily available
- But still [3 orders of magnitude smaller than what Google/Yahoo/MSN index](#)

## Standard relevance benchmarks: Others

---

- **NTCIR**
  - East Asian language and cross-language retrieval information
- **Cross Language Evaluation Forum (CLEF)**
  - This evaluation series has concentrated on European languages and **cross-language information retrieval**.
- **Many others**



THANK  
YOU

---

**Bhaskarjyoti Das**  
Department of Computer Science  
Engineering



# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

## Unranked Evaluation

---

**Bhaskarjyoti Das**  
Department of Computer Science  
Engineering

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

## Unranked Evaluation in IR – Part 1

**Bhaskarjyoti Das**

Department of Computer Science Engineering

## Why we Need Evaluation Metric ?

---



- To answer such questions
  - Is Google better than Bing?
  - Which smoothing method is most effective?
  - Is BM25 better than language models?
  - Shall we perform stemming or stopword removal?
- We need a quantifiable metric, by which we can compare different IR systems
  - As unranked retrieval sets
  - As ranked retrieval results

## Evaluation of Unranked Retrieval

---

In a Boolean retrieval system

**Precision:** fraction of retrieved documents that are relevant,  
i.e.,  $p(\text{relevant} | \text{retrieved})$

**Recall:** fraction of relevant documents that are retrieved, i.e.,  
 $p(\text{retrieved} | \text{relevant})$

**Positive :** based on relevance

	relevant	nonrelevant
retrieved	true positive (TP)	false positive (FP)
not retrieved	false negative (FN)	true negative (TN)

Precision:

$$P = \frac{TP}{TP + FP}$$

## Precision /Recall tradeoff

---

- Precision and recall **trade off** against each other
  - Precision decreases as the **number of retrieved documents increases** (unless in perfect ranking), while **recall keeps increasing** (**worst case remains same**)
  - These two metrics **emphasize different perspectives** of an IR system
    - Precision: prefers systems **retrieving fewer documents, but highly relevant**
    - Recall: prefers systems **retrieving more documents**

## Precision /Recall tradeoff

---

- Recall is a **non-decreasing function** of the number of docs retrieved.
- A system **that returns all docs** has **100% recall!** You can **increase recall** by **returning more docs**
- The converse is also true (usually): It's **easy to get high precision** for **very low recall**.

## Why We Need a Combined Score ?

---

- How to compare two search engines – one high on precision and the other high on recall ?
- Both precision and recall are important. Can we combine both ?

## A combined measure: F

- Summarizing precision and recall to a single value
  - In order to compare different systems
  - F-measure: weighted harmonic mean of precision and recall,  $\alpha$  balances the trade-off

$$F = \frac{1}{\alpha \frac{1}{P} + (1 - \alpha) \frac{1}{R}}$$

Why harmonic mean?

System1: P:0.53, R:0.36

System2: P:0.01, R:0.99

$$\left( F_1 = \frac{2}{\frac{1}{P} + \frac{1}{R}} \right)$$

H	A
0.429	0.445
0.019	0.500

*Equal weight  
between precision  
and recall*

## F: Example with a Contingency Table

	relevant	not relevant	
retrieved	20	40	60
not retrieved	60	1,000,000	1,000,060
	80	1,000,040	1,000,120

- $P = 20/(20 + 40) = 1/3$

- $R = 20/(20 + 60) = 1/4$

- $F_1 = 2 \frac{\frac{1}{3}}{\frac{1}{3} + \frac{1}{4}} = 2/7$

## Accuracy

---

- Why do we use complex measures like precision, recall, and  $F$ ?
  - Why not something simple like accuracy?
- Accuracy is the fraction of decisions (relevant/nonrelevant) that are correct.
  - In terms of the contingency table,  
$$\text{accuracy} = (TP + TN) / (TP + FP + FN + TN)$$



**PES**  
**UNIVERSITY**

**THANK YOU**

---

**Bhaskarjyoti Das**  
Department of Computer Science  
Engineering



# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

## Unranked Evaluation Part 2

---

**Bhaskarjyoti Das**  
Department of Computer Science  
Engineering

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

## Unranked Evaluation in IR – part 2

**Bhaskarjyoti Das**  
Department of Computer Science Engineering

## Why is Accuracy not a Useful Measure ?

---

- Compute precision, recall and  $F_1$  for this result set:

	relevant	not relevant
retrieved	18	2
not retrieved	82	1,000,000,000



- What is the approximate accuracy here ?
- The snoogle search engine **below always returns 0 results ("0 matching results found")**, regardless of the query and gives **high accuracy !**
- If you do the complementary act of **returning every document to all query**, then you get **high recall !**

## Why accuracy is a useless measure in IR

---

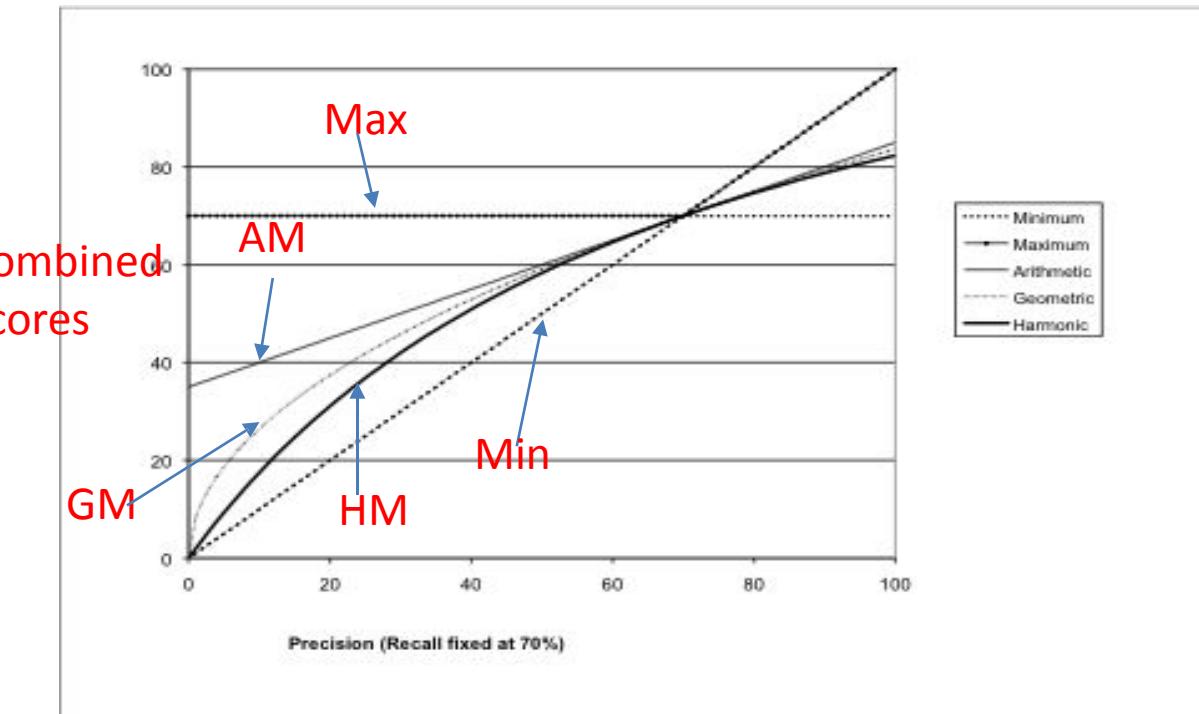
- Simple trick to maximize accuracy in web IR: always say no and return nothing. You then get 99.99% accuracy on most queries.
- Searchers on the web (and in IR in general) **want to find something** and have a certain tolerance for junk.
- It's better to return some bad hits as long as you return something.
- →**We use precision, recall, and  $F$  for evaluation, not accuracy.**

## F: Why harmonic mean?

---

- Why don't we use a **different mean of  $P$  and  $R$**  as a measure
  - the arithmetic mean ( $P+R)/2$  or Geometric Mean  $\sqrt{PR}$ )
- The simple (arithmetic) mean is 50% for “return-everything” search engine, which is too high.
- Desideratum: Punish really bad performance on either **precision or recall**.
- Taking the minimum achieves this but minimum is not smooth and **not using both  $P$  and  $R$** .
- It can be easily proved that  $2PR/(P+R) \leq \sqrt{PR} \leq (P+R)/2$
- $F$  (harmonic mean) is a kind of **smooth minimum**.

## F1 and other averages



- **AM** :  $(P+0.7)/2$  i.e. straight line
- **Max** : As long as  $P < 0.7$ ,  $R = 0.7$  is picked up and then  $P$  picked up
- **Min** : If  $P < 0.7$ ,  $P$  is picked up, otherwise  $R$  is picked up
- **GM vs HM** : For low precision and high precision, effect on HM is more moderate !

## Difficulties in using precision, recall and F

---

- Should average over large document collection/query ensembles
- Need human relevance assessments
  - People aren't reliable assessors
- Assessments have to be binary
  - Nuanced assessments?
- Heavily skewed by collection/authorship
  - Results may not translate from one domain to another



**PES**  
UNIVERSITY

**THANK YOU**

---

**Bhaskarjyoti Das**  
Department of Computer Science  
Engineering



**PES**  
UNIVERSITY

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

## Ranked Evaluation

---

**Bhaskarjyoti Das**  
Department of Computer Science  
Engineering

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

## Ranked Evaluation Part 1

**Bhaskarjyoti Das**  
Department of Computer Science Engineering

## Recap

---

Combined measure that assesses precision/recall tradeoff is **F measure** (weighted harmonic mean):

$$F = \frac{1}{\alpha \frac{1}{P} + R(1-\alpha)} = \frac{(\beta^2 + 1)PR}{\beta^2 P + R}$$

- People usually use balanced  $F_1$  measure  
i.e., with  $\beta = 1$  or  $\alpha = \frac{1}{2}$
- $\alpha = [0,1]$  and  $\beta = [0, \infty]$  as  $\beta = (1 - \alpha) / \alpha$
- Values of  $\beta < 1$  emphasize precision and  $\beta > 1$  emphasize recall

## Recap

---

- Values of  $\beta < 1$  emphasize precision and  $\beta > 1$  emphasize recall
- Is Precision always important ? It depends ;
  - For General Searcher with limited patience, YES
  - For Searcher looking for Legal Information or a Researcher looking for all related areas of research, Recall is more important !

## Variance of measure like precision/recall

---

- For a test collection, it is usual that a system does badly on some information needs (e.g.,  $P = 0.2$  at  $R = 0.1$ ) and really well on others (e.g.,  $P = 0.95$  at  $R = 0.1$ ).
- Indeed, it is usually the case that the variance of the same system across queries is much greater than the variance of different systems on the same query.
- That is, there are easy information needs and hard ones.

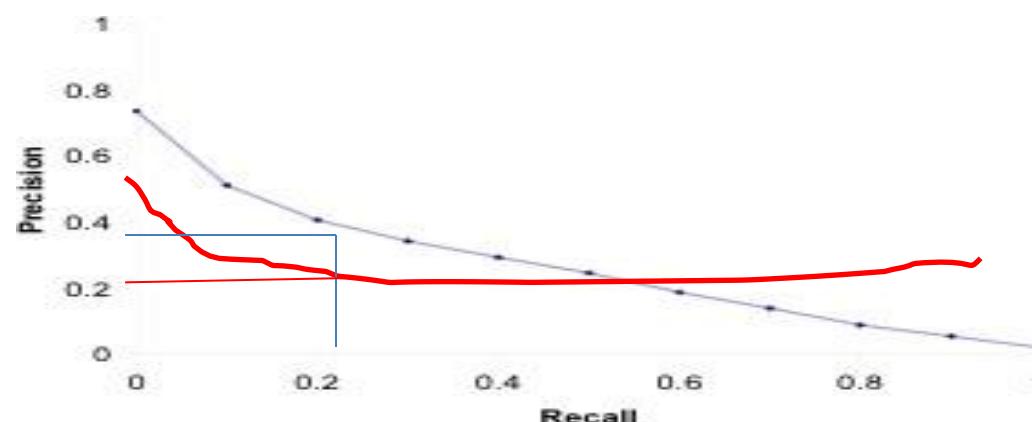
## Precision-recall curve

---

- Precision/recall/F are measures for unranked sets.
- We can easily turn set measures into measures of **ranked lists**.
- Just compute the set measure for each of : the top 1, top 2, top 3, top 4 etc results
- Doing this for precision and recall gives you a **precision- recall curve**.

## Can we Compare Two Search Engines then ?

- The system can return any number of results
- By taking various numbers of the top (ranked) returned documents (at various levels of recall), the evaluator can produce a *precision-recall curve*
- *But how would you compare a search engine with another search engine ? At fixed recall points ( you can't vary both as they are related in some way ), get precision for many queries ? Then do some average ?*

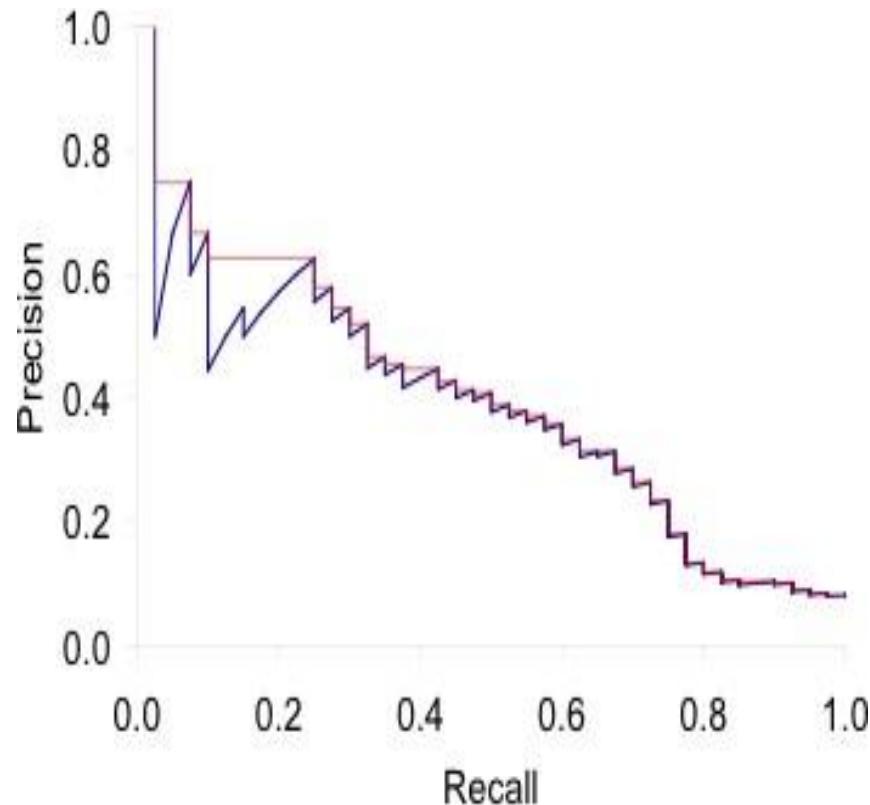


## Why should We Look at Precision and not Recall ?

---

- In precision, we compute (total no relevant doc / K retrieved ).  
Denominator K is same for both search engines !
- In recall, we compute (total no relevant doc / no of relevant doc in corpus). Denominator same for both search engines being compared and is a large number !
- So, we can just look at (same) numerator in both cases for comparing search engines.
- And, we might as well look at only precision ! That is more prevalent need anyway.

## A Precision-recall curve is a Saw Tooth



Case 1 : next 1 document considered is irrelevant. So, recall remains same and precision drops !

Case 2 : next 1 document considered are relevant. So recall anyway increases and so is precision . So, we have right shift !

Calculation above is cumulative. Recall always increases but precision may not

So we need to smoothen this curve so that we can get value at any point.

## Why Averaging over Queries ?

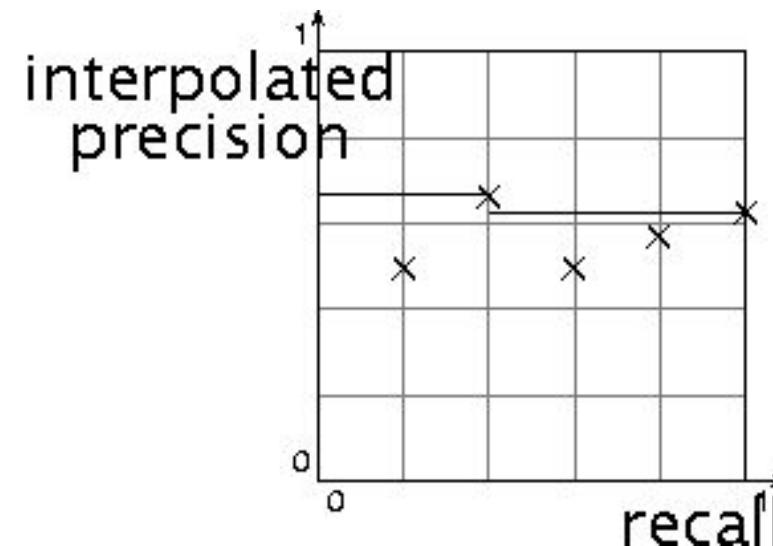
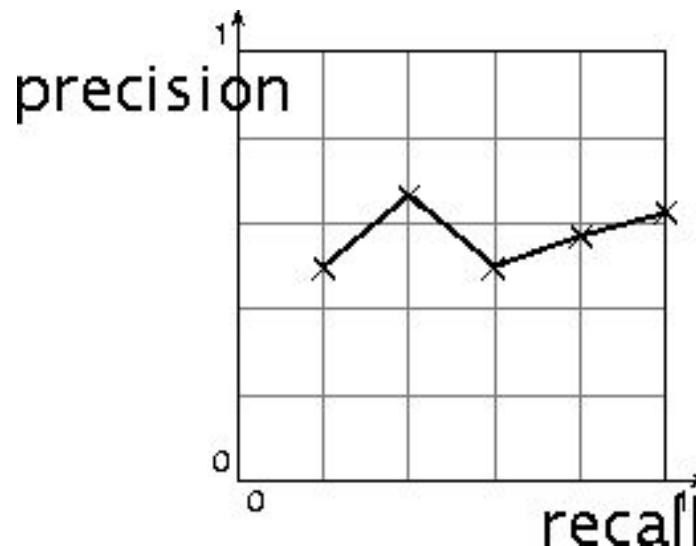
---

- A precision-recall graph for one query is not a very sensible thing to look at
- You need to average performance over a whole bunch of queries.
- Also there's a technical issue:
  - Precision-recall calculations place some points on the graph
  - How do you determine a value (interpolate) between the points?

## Interpolated Precision

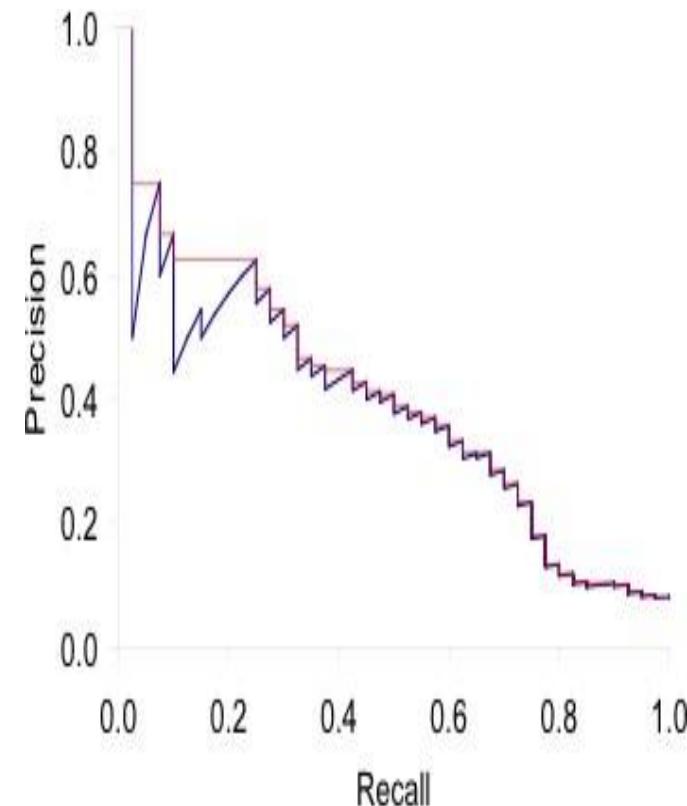
Idea: If locally precision increases with increasing recall, then you should get to count that... So you take the max of precisions to right of value

$$P_{\text{interp}}(r) = \max(r' \geq r) P(r')$$



## A Precision-recall curve

- Each point corresponds to a result for the top  $k$  ranked hits ( $k = 1, 2, 3, 4, \dots$ ).
- Interpolation (in red): Take maximum of all future points
- Rationale for interpolation: The user is willing to look at more stuff if both precision and recall get better.

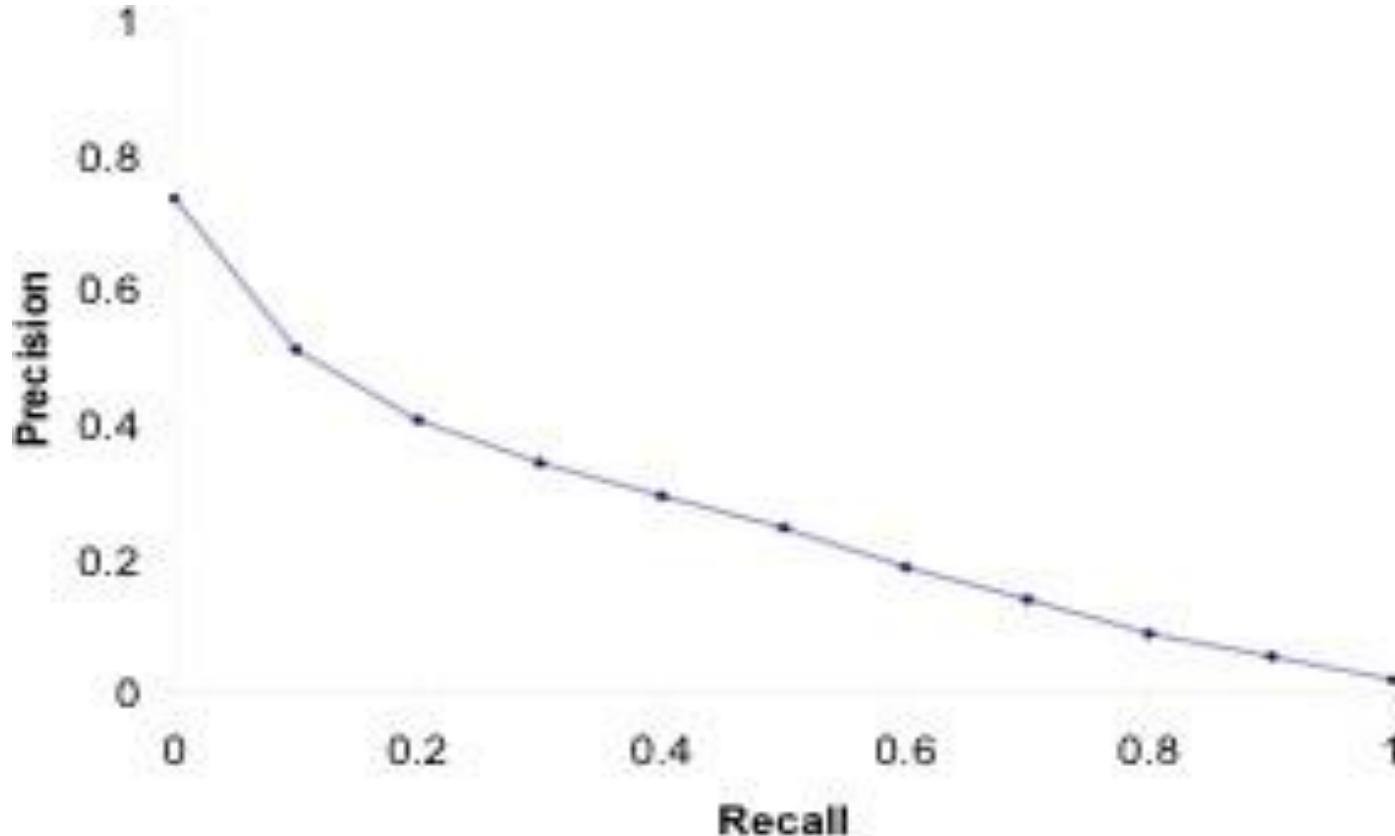


## Averaged 11-point precision/recall graph

---

- For each information need, compute interpolated precision at recall levels 0.0, 0.1, 0.2.. (What is the precision at Recall 0 ? )
- Do this for each of the queries (=information need) in the evaluation benchmark
- Average over queries but still it is a 11 point measure
- The curve is typical of performance levels at TREC 8

## First TREC Ad Hoc Evaluation : 11-point Interpolated Average Precision



## Mean Average Precision (MAP)

- Graphs are good, but people want summary measures!
- MAP ( mean average Precision) offers such a measure
- For single information need, average precision is the average of the precision value obtained for the set of top K documents existing after each relevant document is retrieved
- Perhaps appropriate for most of web search: all people want are good matches on the first one or two results pages
- This value is then averaged over information need ( query representing that )
- Using MAP , fixed recall levels are NOT chosen and there is no interpolation

## Mean Average Precision (MAP)

---

- Consider rank position of each relevant doc
  - E.g.,  $K_1, K_2, \dots, K_R$
- Compute P@K for each  $K_1, K_2, \dots, K_R$
- Average precision = average of those P@K

E.g.,



$$AvgPrec = \left( \frac{1}{1} + \frac{2}{3} + \frac{3}{5} \right) / 3$$

MAP is mean of Average Precision across multiple queries/rankings

## Average Precision about One Query



= the relevant documents

Ranking #1



Recall	0.17	0.17	0.33	0.5	0.67	0.83	0.83	0.83	1.0
--------	------	------	------	-----	------	------	------	------	-----

Precision	1.0	0.5	0.67	0.75	0.8	0.83	0.71	0.63	0.56	0.6
-----------	-----	-----	------	------	-----	------	------	------	------	-----

Ranking #2



Recall	0.0	0.17	0.17	0.17	0.33	0.5	0.67	0.67	0.83	1.0
--------	-----	------	------	------	------	-----	------	------	------	-----

Precision	0.0	0.5	0.33	0.25	0.4	0.5	0.57	0.5	0.56	0.6
-----------	-----	-----	------	------	-----	-----	------	-----	------	-----

AvgPrec of the two rankings

$$\text{Ranking } \#1: (1.0 + 0.67 + 0.75 + 0.8 + 0.83 + 0.6) / 6 = 0.78$$

$$\text{Ranking } \#2: (0.5 + 0.4 + 0.5 + 0.57 + 0.56 + 0.6) / 6 = 0.52$$

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

MAP is about a System



Ranking #1



	Recall	0.2	0.2	0.4	0.4	0.4	0.6	0.6	0.6	0.8	1.0
	Precision	1.0	0.5	0.67	0.5	0.4	0.5	0.43	0.38	0.44	0.5



Ranking #2



	Recall	0.0	0.33	0.33	0.33	0.67	0.67	1.0	1.0	1.0	1.0
	Precision	0.0	0.5	0.33	0.25	0.4	0.33	0.43	0.38	0.33	0.3

Query 1,

$$\text{AvgPrec} = (1.0 + 0.67 + 0.5 + 0.44 + 0.5) / 5 = 0.62$$

Query 2,

$$\text{AvgPrec} = (0.5 + 0.4 + 0.43) / 3 = 0.4$$

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB MAP

---



- Precision at fixed retrieval level
  - Precision-at- $k$ : Precision of top  $k$  results
  - But: averages badly and has an arbitrary parameter of  $k$
- $\text{MAP}(Q) = \frac{1}{|Q|} \sum_{k=1}^{|Q|} \text{Precision}(R_{jk})$
- Information need  $q_j \in Q$  is  $\{d_{j1}, d_{j2}, \dots, d_{jk}\}$   
 $R_j$  is the set of ranked retrieval from top results using the  $j$  queries till you get to document  $d_{jk}$
- Since averaging over all queries, each information need is weighted equally even many documents are relevant over few queries and very few to many queries !

- More agreement in MAP for an individual information need across systems ( than for different information needs)
- It means set of info. Needs should be diverse for MAP to be an effective measure across systems
- MAP gives precision not at fixed recall levels
- If no relevant document is retrieved for a query, the precision value of the Precision( $R_{jk}$ ) is zero
- For a single query, average precision approximates the area under the precision recall curve
- MAP roughly equates the average area under the P-R curve for a set of queries
- MAP factors precision at all recall level ( P-R curve) but for web search, recall is not important – only thing that matters is top K retrieval ( precision)

- MAP is macro-averaging: each query counts equally
- MAP assumes users are interested in finding many relevant documents for each query
- MAP requires many relevance judgments in text collection

- More agreement in MAP for an individual information need across systems ( than for different information needs)
- It means set of info. Needs should be diverse for MAP to be an effective measure across systems
- MAP gives precision not at fixed recall levels
- If no relevant document is retrieved for a query, the precision value of the Precision( $R_{jk}$ ) is zero
- For a single query, average precision approximates the area under the precision recall curve
- MAP roughly equates the average area under the P-R curve for a set of queries
- MAP factors precision at all recall level ( P-R curve) but for web search, recall is not important – only thing that matters is top K retrieval ( precision)



THANK  
YOU

---

**Bhaskarjyoti Das**  
Department of Computer Science  
Engineering



**PES**  
UNIVERSITY

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

## Ranked Evaluation

---

**Bhaskarjyoti Das**  
Department of Computer Science  
Engineering

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

## Ranked Evaluation Part 1

**Bhaskarjyoti Das**  
Department of Computer Science Engineering

## Precision at K

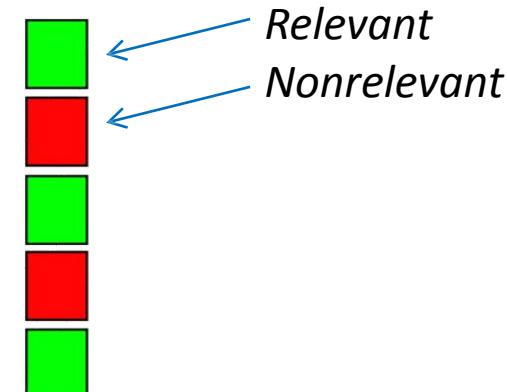
---

- MAP factors precision at all recall level ( P-R curve) but for web search, recall is not important – only thing that matters is top K retrieval ( precision)
- What matters is : how many relevant documents are there in the first 1-3 pages !!
- This leads to measuring precision at fixed low levels of retrieved results
- Advantage : no estimate of corpus needed
- Disadvantage : unstable measure – does not average well ! Total number of relevant documents for a query has a strong effect on this

## Precision at K

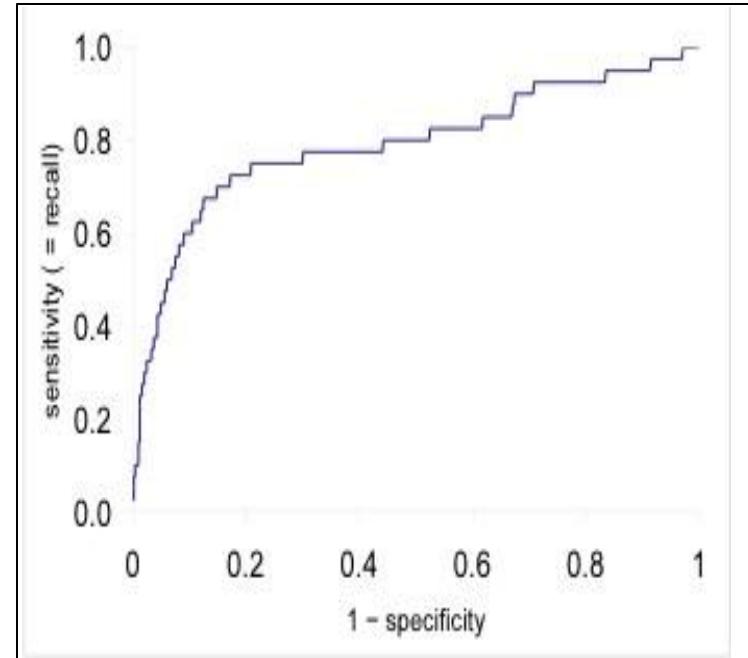
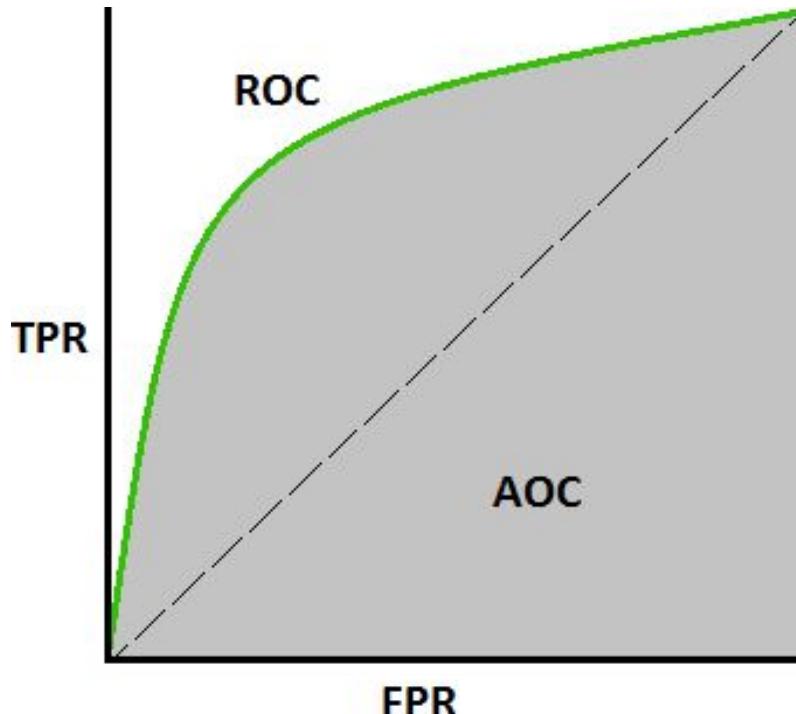
---

- Set a ranking position threshold K
- Ignores all documents ranked lower than K
- Compute precision in these top K retrieved documents
  - Example-  
 $P@3$  of 2/3  
 $P@4$  of 2/4  
 $P@5$  of 3/5
- In a similar fashion we have Recall@K



- R –precision attempts to address the issue in precision at K
- Given : a set of known relevant document Rel in a corpus
- Measure : measure top  $|Rel|$  results of the system. If there are r relevant documents, then  $r/|Rel|$  is the precision.
- What is the recall ? The recall is also  $r/|Rel|$
- So, R –Precision is a break-even point in the precision-recall curve (represents only one point).
- Why should you not be interested in this break even point ?

## ROC curve vs. Precision Recall Curve



$$\text{TPR/Recall/Sensitivity} = \frac{TP}{TP+FN}$$

$$\text{FPR} = 1 - \text{Specificity}$$

$$\text{Specificity} = \frac{TN}{TP+FN}$$

## ROC curve vs. Precision Recall Curve

---

- For a typical corpus ( web ?), set of negative is very large and the value of specificity is almost 1 and FPR is almost 0
- But we are only interested in the small area in the lower left corner precision-recall graph ( $0 < \text{Recall} < 0.4$ )
- Precision-recall graph “blows up” this area
- Precision-recall graph loosely referred to as ROC curve but not accurate

## Beyond Binary Relevance

---

- The level of documents' relevance quality with respect to a given query varies
  - Highly relevant documents are more useful than marginally relevant documents
  - The lower the ranked position of a relevant document is, the less useful it is for the user, since it is less likely to be examined
  - ***Discounted Cumulative Gain***

## Discounted Cumulative Relevance

---

- Uses graded relevance as a measure of usefulness, or gain, from examining a document
- Gain is accumulated starting at the top of the ranking and discounted at lower ranks
- Typical discount is  $1/\log(\text{rank})$ 
  - With base 2, the discount at rank 4 is  $1/2$ , and at rank 8 it is  $1/3$

## Discounted Cumulative Relevance

DCG is the total gain accumulated at a particular rank position p:

Alternative formulation

$$DCG_p = rel_1 + \sum_{i=2}^p \frac{rel_i}{\log_2 i}$$

Relevance label at position  $i$

Standard metric in some web search companies  
Emphasize on retrieving highly relevant documents

$$DCG_p = \sum_{i=1}^p \frac{2^{rel_i} - 1}{\log(1 + i)}$$

## Discounted Cumulative Relevance

DCG is the total gain accumulated at a particular rank position p:

Alternative formulation

$$DCG_p = rel_1 + \sum_{i=2}^p \frac{rel_i}{\log_2 i}$$

Relevance label at position  $i$

Standard metric in some web search companies  
Emphasize on retrieving highly relevant documents

$$DCG_p = \sum_{i=1}^p \frac{2^{rel_i} - 1}{\log(1 + i)}$$

## Normalized Discounted Cumulative Relevance

---

- Normalization is useful for contrasting queries with varying numbers of relevant results
- Normalize DCG at rank n by the DCG value at rank n of the ideal ranking
  - The ideal ranking is achieved via ranking documents with their relevance labels

## Normalized Discounted Cumulative Relevance :

### Example

i	5 documents: d <sub>1</sub> , d <sub>2</sub> , d <sub>3</sub> , d <sub>4</sub> , d <sub>5</sub>		Ranking Function <sub>1</sub>		Ranking Function <sub>2</sub>	
	Document Order	rel <sub>i</sub>	Document Order	rel <sub>i</sub>	Document Order	rel <sub>i</sub>
1	d <sub>5</sub>	4	d <sub>3</sub>	2	d <sub>5</sub>	4
2	d <sub>4</sub>	3	d <sub>4</sub>	3	d <sub>3</sub>	2
3	d <sub>3</sub>	2	d <sub>2</sub>	1	d <sub>4</sub>	3
4	d <sub>2</sub>	1	d <sub>5</sub>	4	d <sub>1</sub>	0
5	d <sub>1</sub>	0	d <sub>1</sub>	0	d <sub>2</sub>	1
	NDCG <sub>GT</sub> =1.00		NDCG <sub>RF1</sub> =0.67		NDCG <sub>RF2</sub> =0.97	

$$DCG_{\bar{GT}} = \frac{2^4 - 1}{\log_2 2} + \frac{2^3 - 1}{\log_2 3} + \frac{2^2 - 1}{\log_2 4} + \frac{2^1 - 1}{\log_2 5} + \frac{2^0 - 1}{\log_2 6} = 21.35$$

$$DCG_{\bar{RF1}} = \frac{2^2 - 1}{\log_2 2} + \frac{2^3 - 1}{\log_2 3} + \frac{2^1 - 1}{\log_2 4} + \frac{2^4 - 1}{\log_2 5} + \frac{2^0 - 1}{\log_2 6} = 14.38$$

$$DCG_{\bar{RF2}} = \frac{2^4 - 1}{\log_2 2} + \frac{2^2 - 1}{\log_2 3} + \frac{2^3 - 1}{\log_2 4} + \frac{2^0 - 1}{\log_2 5} + \frac{2^1 - 1}{\log_2 6} = 20.78$$



THANK  
YOU

---

**Bhaskarjyoti Das**  
Department of Computer Science  
Engineering



**PES**  
UNIVERSITY

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

## Assessing Relevance

---

**Bhaskarjyoti Das**  
Department of Computer Science  
Engineering

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

## Assessing Relevance

**Bhaskarjyoti Das**

Department of Computer Science  
Engineering

## Assessing relevance – Why ?

---

- To properly evaluate a system, your test information needs must be germane to the documents in the test document collection, and appropriate for predicted usage of the system.
- Using random combinations of query terms as an information need is generally not a good idea because typically they will not resemble the actual distribution of information needs.

- Adhoc retrieval to refer to regular retrieval without relevance feedback.
- Human annotation is expensive and time consuming Cannot afford exhaustive annotation of large corpus.
- Pooling : For large collections, assess relevance only for a subset of the documents for each query that is formed from top K returned by a number of different IR systems

## Validity of relevance assessments

---

- Relevance assessments are only usable if they are consistent.
- If they are not consistent, then there is no “truth” and experiments are not repeatable.
- How can we measure this consistency or agreement among judges?
  - Kappa measure

- Kappa is measure of how much judges agree or disagree.
- Designed for categorical judgments
- Corrects for chance agreement

*kappa* statistic

A measure of agreement between judges

$$\kappa = \frac{P(A) - P(E)}{1 - P(E)}$$

$P(A)$  is the proportion of the times judges agreed

$P(E)$  is the proportion of times they would be expected to agree by chance

$\kappa = 1$  if two judges always agree

$\kappa = 0$  if two judges agree by chance

$\kappa < 0$  if two judges always disagree

- Values of  $k$  in the interval  $[2/3, 1.0]$  are seen as acceptable.
- With smaller values: need to redesign relevance assessment methodology used etc.

## Calculating the Kappa statistic

		Judge 2 Relevance		
		Yes	No	Total
Judge 1 Relevance	Yes	300	20	320
	No	10	70	80
	Total	310	90	400

Observed proportion of the times the judges agreed

$$P(A) = (300 + 70)/400 = 370/400 = 0.925$$

Pooled marginals

$$P(\text{nonrelevant}) = (80 + 90)/(400 + 400) = 170/800 = 0.2125$$

$$P(\text{relevant}) = (320 + 310)/(400 + 400) = 630/800 = 0.7878$$

Probability that the two judges agreed by chance  $P(E)$  =

$$P(\text{nonrelevant})^2 + P(\text{relevant})^2 = 0.2125^2 + 0.7878^2 = 0.665$$

Kappa statistic  $\kappa = (P(A) - P(E))/(1 - P(E)) =$

$$(0.925 - 0.665)/(1 - 0.665) = 0.776 \text{ (still in acceptable range)}$$

Information need	number of docs judged	disagreements
51	211	6
62	400	157
67	400	68
95	400	110
127	400	106



**PES**  
UNIVERSITY

THANK  
~~YOU~~  
\_\_\_\_\_

**Bhaskarjyoti Das**  
Department of Computer Science  
Engineering



**PES**  
**UNIVERSITY**

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

## Assessing Relevance

---

**Bhaskarjyoti Das**  
Department of Computer Science  
Engineering

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

## Critiques and Justifications

**Bhaskarjyoti Das**

Department of Computer Science  
Engineering

## Impact of Interjudge disagreement

---

- Judges disagree a lot. Does that mean that the results of information retrieval experiments are meaningless?
- No.
- Large impact on absolute performance numbers
- Virtually no impact on ranking of systems

## Assessor Consistency

---

- Is inconsistency of assessors a concern?
  - Human **annotators** are idiosyncratic and variable
  - **Relevance judgments** are subjective
- Studies mostly concluded that the **inconsistency didn't affect relative comparison** of systems
  - Success of an IR system depends on how good it is at **satisfying the needs of these idiosyncratic humans**
  - Lesk & Salton (1968): assessors mostly disagree on documents at **lower ranks**, but **measures are more affected by top-ranked documents**

## Impact of Interjudge disagreement

---

- Suppose we want to know if algorithm A is better than algorithm B
- An information retrieval experiment will give us a reliable answer to this question ...
- . . . even if there is a lot of disagreement between judges.

- Goal of any IR system
  - Satisfying users' information need
- Core quality measure criterion
  - "*how well a system meets the information needs of its users.*" – wiki

## What has been Considered

---

- The ability of the system **to present all relevant documents**
  - Recall-driven measures
- The ability of the **system to withhold non-relevant documents**
  - Precision-driven measures

## Advantages of System Evaluation

---

- We have a **fixed setting** in which we can **vary IR system and system parameters** for **comparative experiments**
- Such **formal testing** is **much less expensive**
- Allows **cleaner diagnosis** of the **effect of changing system parameters** than **user studies for IR**
- Once we have **confidence in formal measures**, we can **proceed to optimize effectiveness** by **ML methods.**

- Relevance of one document is assumed to be independent of that of the other documents ( documents are scored against queries but not against each other even though we are ranking them !)
- Assessments are binary
- Relevance is subjective
- User's information needs do not change as they start looking at retrieved results
- Results based on one collection heavily skewed by the choice of collection, queries and relevance judgement

## Marginal relevance

---

- We've defined relevance for an isolated query-document pair.
- Alternative definition: marginal relevance
- The marginal relevance of a document at position  $k$  in the result list is the additional information it contributes over and above the information that was contained in documents  $d_1 \dots d_{k-1}$ .

## Marginal relevance

---

- When marginal relevance is needed ?
- Even if a document is highly relevant, its information can be completely irrelevant due to other documents that are already retrieved !
  - Example – duplicated on www
  - Example – different documents provide similar precis
- Maximizing marginal relevance requires returning documents that exhibit diversity and novelty

## Evaluation at large search engine

---

- Recall is difficult to measure on the web
- Search engines often use precision at top  $k$ , e.g.,  $k = 10 \dots$
- $\dots$  or use measures that reward you more for getting rank 1 right than for getting rank 10 right.
- Search engines also use non-relevance-based measures.

- Example 1: click through on first result
- Not very reliable if you look at a single click through (you may realize after clicking that the summary was misleading and the document is nonrelevant) . . .
- . . . but pretty reliable in the aggregate.
- Example 2: Ongoing studies of user behavior in the lab
- Example 3: A/B testing

- Purpose: Test a single innovation
- Prerequisite: You have a large search engine up and running.
  - Have most users use old system
  - Divert a small proportion of traffic (e.g., 1%) to the new system that includes the innovation
  - Evaluate with an “automatic” measure like click through on first result
    - Now we can directly see if the innovation does improve user happiness.
  - Probably the evaluation methodology that large search engines trust most



THANK  
YOU

---

**Bhaskarjyoti Das**  
Department of Computer Science  
Engineering



**PES**  
UNIVERSITY

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

## Relevance Feedback

---

**Bhaskarjyoti Das**  
Department of Computer Science  
Engineering

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

## Why Relevance Feedback and Query Expansion ?

**Bhaskarjyoti Das**  
Department of Computer Science Engineering

## Relevance Feedback and Query Expansion – Why

---

### Synonymy

- Has an **impact** on the **recall** of most Information Retrieval (IR) systems
- You want for aircraft to match real plane and *not toy plane* but searching for *aircraft* doesn't match with *plane*
- Search for thermodynamic* does not match with discussions on *heat*

## How to Handle Linguistic Ambiguity ?

---

- Linguistic Ambiguity – pick the right set of synonyms ?
- Possible approaches
  - Thesaurus / WordNet*
  - Statistically related terms*
  - Relevance Feedback*
  - Pseudo-relevance Feedback*
  - Dimensionality reduction*
    - Latent Semantic Indexing ( LSI )*
    - Topic Model ( Latent Dirichlet Allocation )*

## Options for Improving Recall Results

---

- ❑ Options for improving recall results...
  - ❑ Global methods
    - ❑ Query expansion/reformulation with
      - ❑ Techniques like spelling correction
      - ❑ Automatic thesaurus generation
  - ❑ Local methods
    - ❑ Relevance feedback
    - ❑ Pseudo or blind relevance feedback

## Relevance Feedback- Basic Idea

---

- ❑ Relevance feedback: user feedback on relevance of docs in initial set of results
  - ❑ User issues a (short, simple) query
  - ❑ The user marks some results as relevant or non-relevant.
  - ❑ The system computes a better representation of the information need based on feedback.
  - ❑ Relevance feedback can go through one or more iterations.
- ❑ Idea: it may be difficult to formulate a good query when you don't know the collection well, so iterate

- We use the term **ad hoc retrieval** to refer to regular retrieval without relevance feedback.
- We will now look at three different examples of relevance feedback that highlight different aspects of the process.

## Relevance Feedback : Image Search Example



PES  
UNIVERSITY

New Page 1 - Netscape

File Edit View Go Bookmarks Tools Window Help

http://nayana.ece.ucsb.edu/ji

Home Browsing and ...

Shopping related 607,000 images are indexed and classified in the database  
Only One keyword is allowed!!!

bike

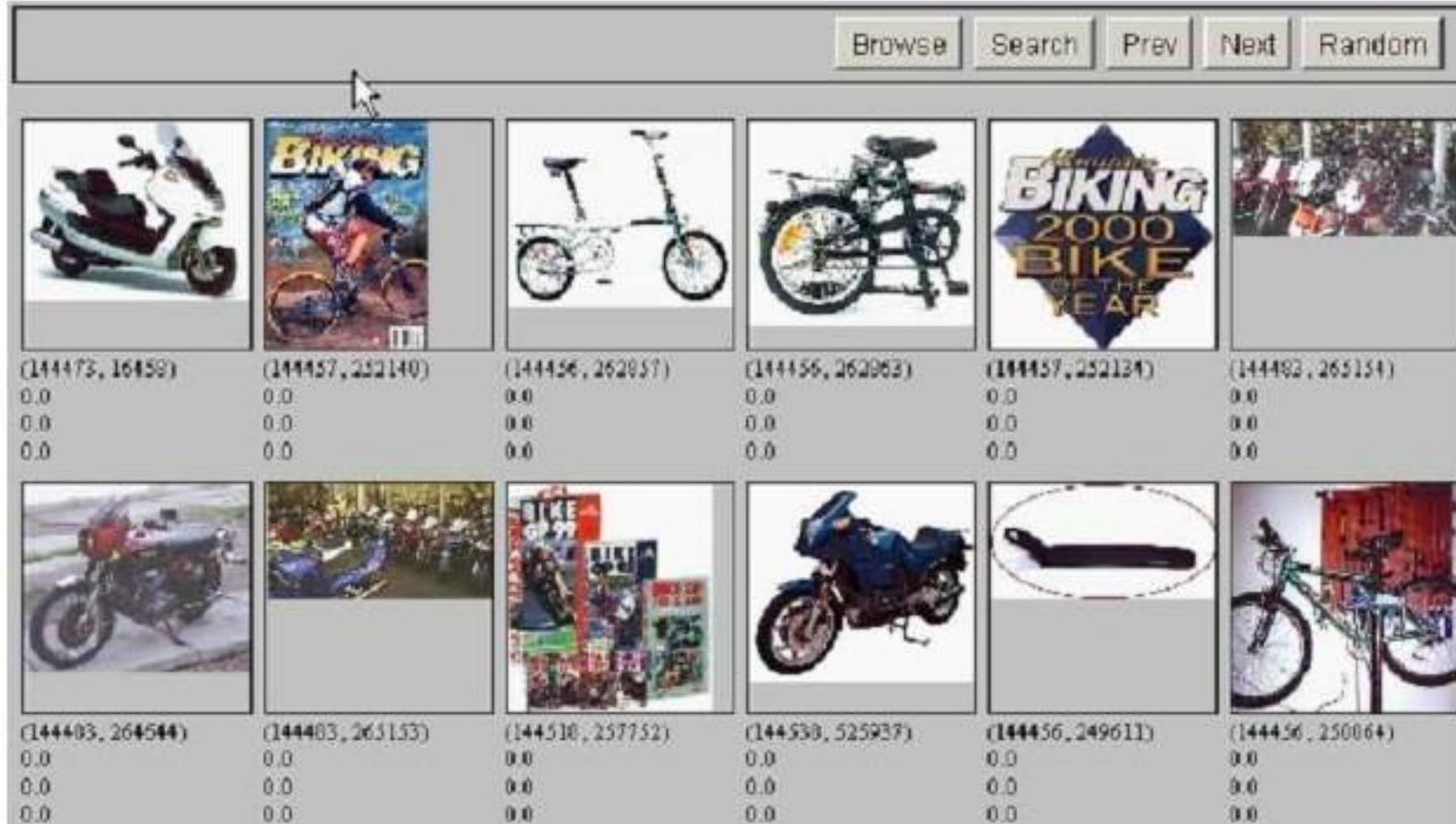
Search

Designed by [Baris Sumengen](#) and [Shawn Newsam](#)

Powered by JLAMP2000 (Java, Linux, Apache, Mysql, Perl, Windows2000)

Hard for user to write exactly what he wants ..

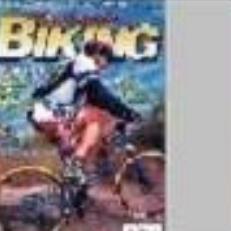
## Results for initial query



All images are not relevant ..

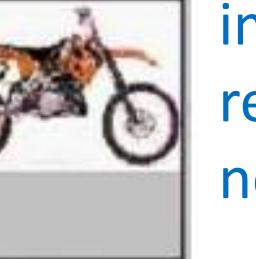
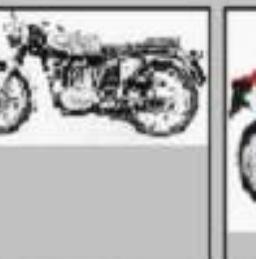
## User feedback : Select what is relevant

Browse Search Prev Next Random

					
(144473, 16458) 0.0 0.0 0.0	(144457, 252140) 0.0 0.0 0.0	(144456, 262957) 0.0 0.0 0.0	(144456, 262963) 0.0 0.0 0.0	(144457, 252134) 0.0 0.0 0.0	(144493, 265154) 0.0 0.0 0.0
					
(144483, 264644) 0.0 0.0 0.0	(144483, 265153) 0.0 0.0 0.0	(144518, 257752) 0.0 0.0 0.0	(144539, 525037) 0.0 0.0 0.0	(144456, 240611) 0.0 0.0 0.0	(144456, 250061) 0.0 0.0 0.0

Selects only what  
is relevant ..

## Results after relevant feedback

						Browse	Search	Prev	Next	Random
										
(144538, 523493)	(144538, 523835)	(144538, 523529)	(14456, 253569)	(14456, 253568)	(144538, 523799)					
0.54182	0.56319296	0.584279	0.64501	0.650275	0.66709197					
0.231944	0.267304	0.280881	0.351395	0.411743	0.358033					
0.309876	0.395889	0.303398	0.293615	0.23855	0.309059					
										
(144473, 16249)	(144456, 249634)	(144456, 253693)	(144473, 16328)	(144483, 265264)	(144478, 512410)					
0.6721	0.675018	0.576901	0.700339	0.70170296	0.70297					
0.393922	0.4639	0.47645	0.309002	0.36176	0.469111					
0.278178	0.211118	0.200451	0.391337	0.339948	0.233859					

All retrieved images are relevant now ..

## Example 2: A text (non-image)example

Initial query:

[new space satellite applications] Results for initial query: ( $r = \text{rank}$ )

$r$		
+ 1	0.539	NASA Hasn't Scrapped Imaging Spectrometer
+ 2	0.533	NASA Scratches Environment Gear From Satellite Plan
3	0.528	Science Panel Backs NASA Satellite Plan, But Urges Launches of Smaller Probes
4	0.526	A NASA Satellite Project Accomplishes Incredible Feat: Staying Within Budget
5	0.525	Scientist Who Exposed Global Warming Proposes Satellites for Climate Research
6	0.524	Report Provides Support for the Critics Of Using Big Satellites to Study Climate
7	0.516	Arianespace Receives Satellite Launch Pact From Telesat Canada
+ 8	0.509	Telecommunications Tale of Two Companies

User then marks relevant documents with

“+”

## Expanded query after relevance feedback

2.074	new	15.106	space	Query expanded by 18 terms (based on some weights)
30.816	satellite	5.660	application	
5.991	nasa	5.196	eos	
4.196	launch	3.972	aster	
3.516	instrument	3.446	arianespace	
3.004	bundespost	2.806	ss	
2.790	rocket	2.053	scientist	
2.003	broadcast	1.172	earth	
0.836	oil	0.646	measure	

query: [new space satellite applications]

## Results for expanded query

- 
- \* 1 0.513 NASA Scratches Environment Gear From Satellite Plan
  - \* 2 0.500 NASA Hasn't Scrapped Imaging Spectrometer
  - 3 0.493 When the Pentagon Launches a Secret Satellite, Space Sleuths Do Some Spy Work of Their Own
  - 4 0.493 NASA Uses 'Warm' Superconductors For Fast Circuit
  - \* 5 0.492 Telecommunications Tale of Two Companies
  - 6 0.491 Soviets May Adapt Parts of SS-20 Missile For Commercial Use
  - 7 0.490 Gaping Gap: Pentagon Lags in Race To Match the Soviets In Rocket Launchers
  - 8 0.490 Rescue of Satellite By Space Agency To Cost \$90 Million

query: [new space satellite applications]

Revised results shown; \* marks the relevant documents in the relevance feedback phase

## Recap for Vector Space Model

---

- Represent both document , query as a vector of n-dimension when n= vocabulary size
- Term Weighting : how important the term is in document
  - Tf : How frequent the term is in a document
  - Normalization : Is the term unusually frequent ?
  - IDF : Is it a content term ?
- Vector Similarity : Is the document **D** near the query **Q** ?
  - Project to unit radius hypersphere **D.Q**

## Main Idea for Relevance Feedback

---

- When the user provides feedback that he wants to see some documents higher in the ranking, how does the Search Engine interpret that ?
  - User Label → new Query
  - Should match more relevant documents
  - Should match fewer irrelevant documents



THANK  
YOU

---

**Bhaskarjyoti Das**  
Department of Computer Science  
Engineering



**PES**  
UNIVERSITY

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

## Relevance Feedback

---

**Bhaskarjyoti Das**  
Department of Computer Science  
Engineering

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

## Rocchio Algorithm

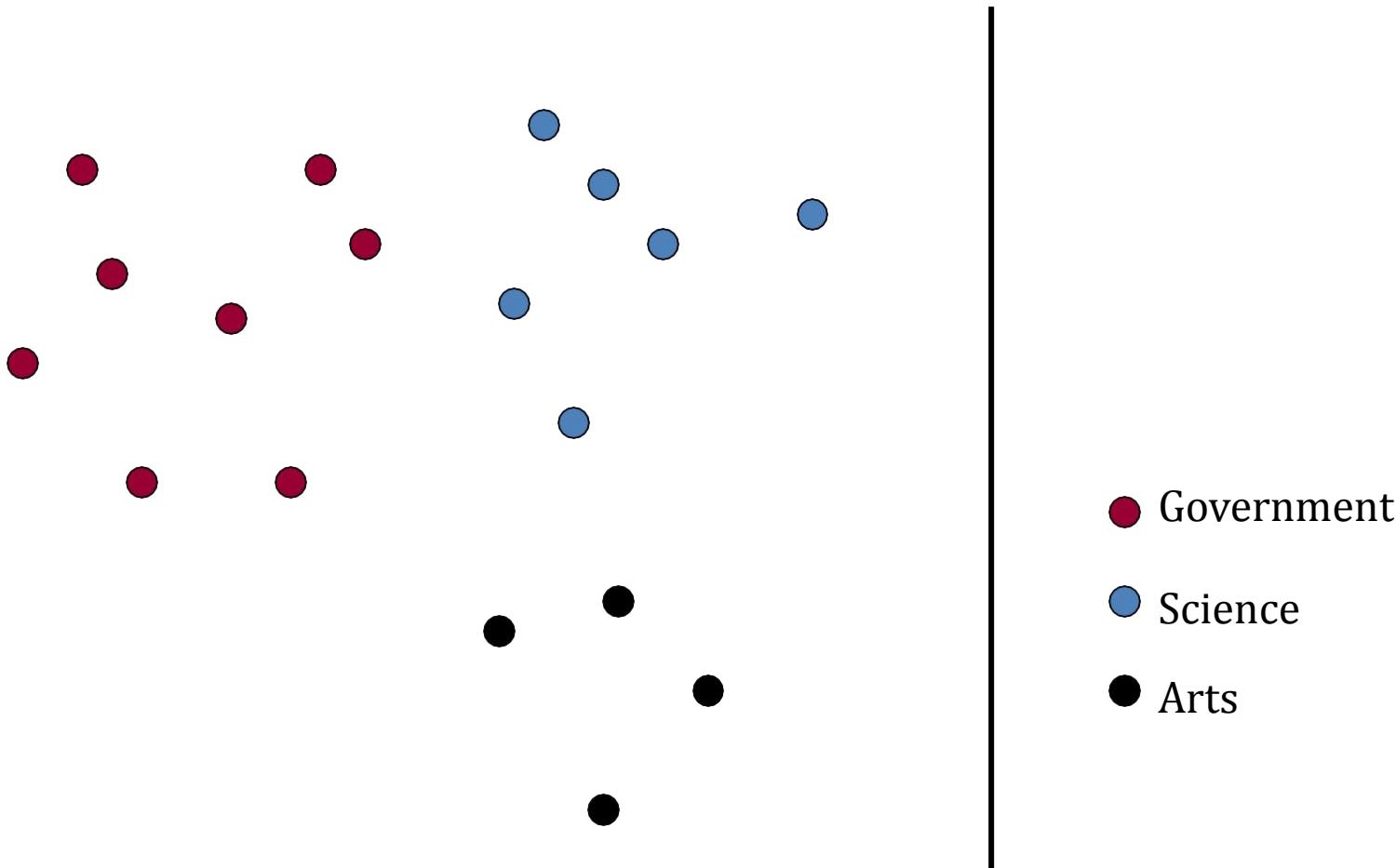
Bhaskarjyoti Das

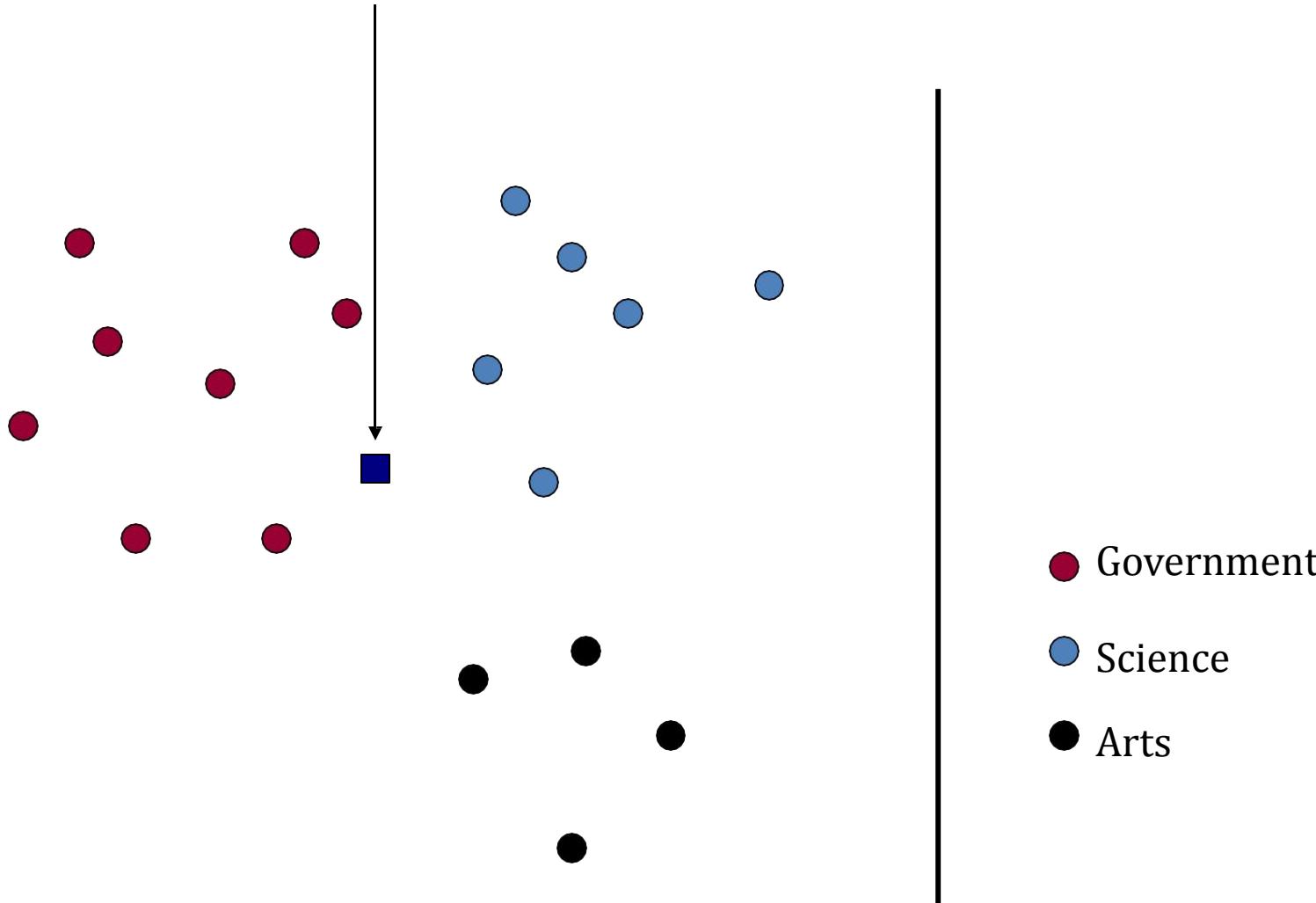
Department of Computer Science  
Engineering

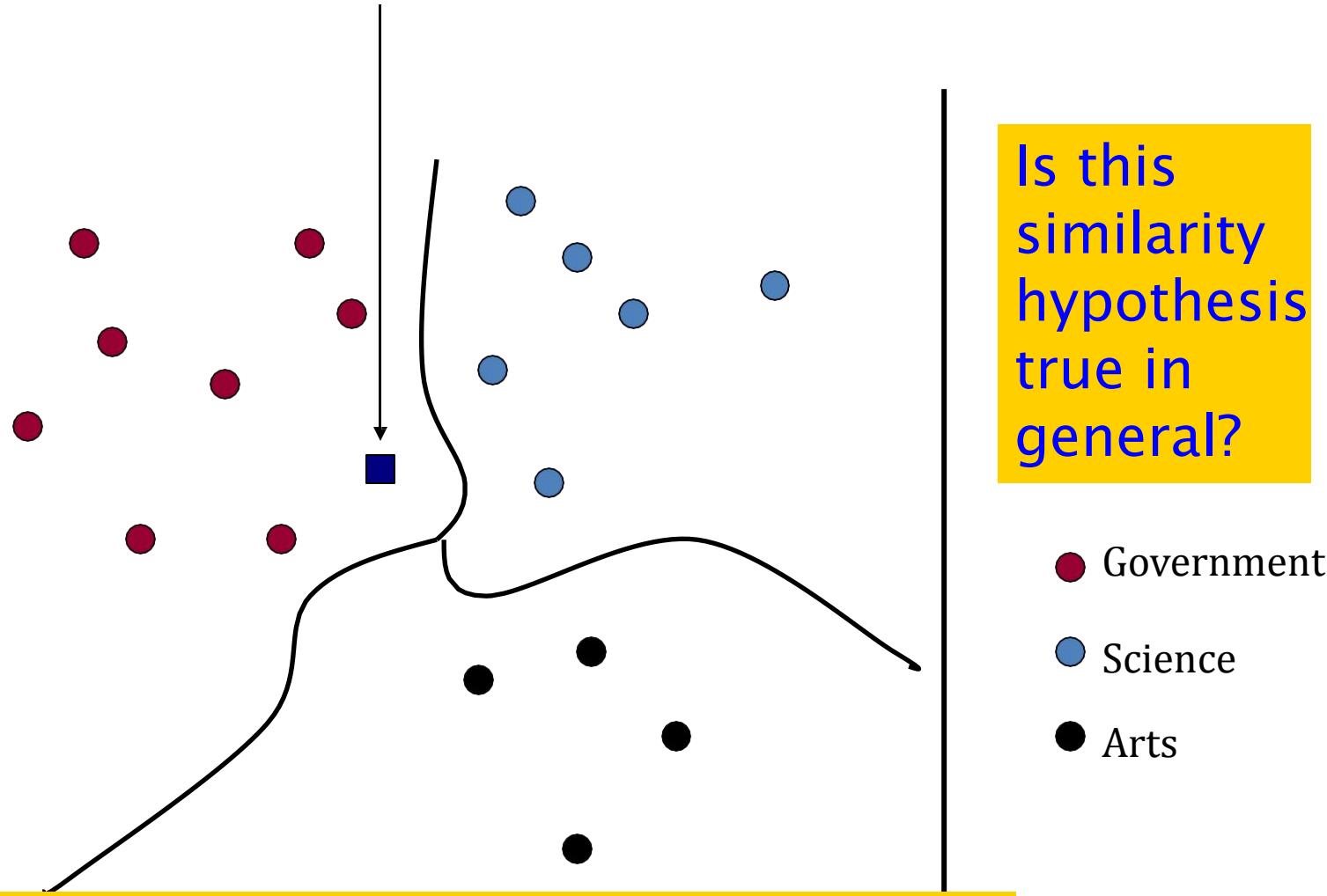
## Classification using Vector Space

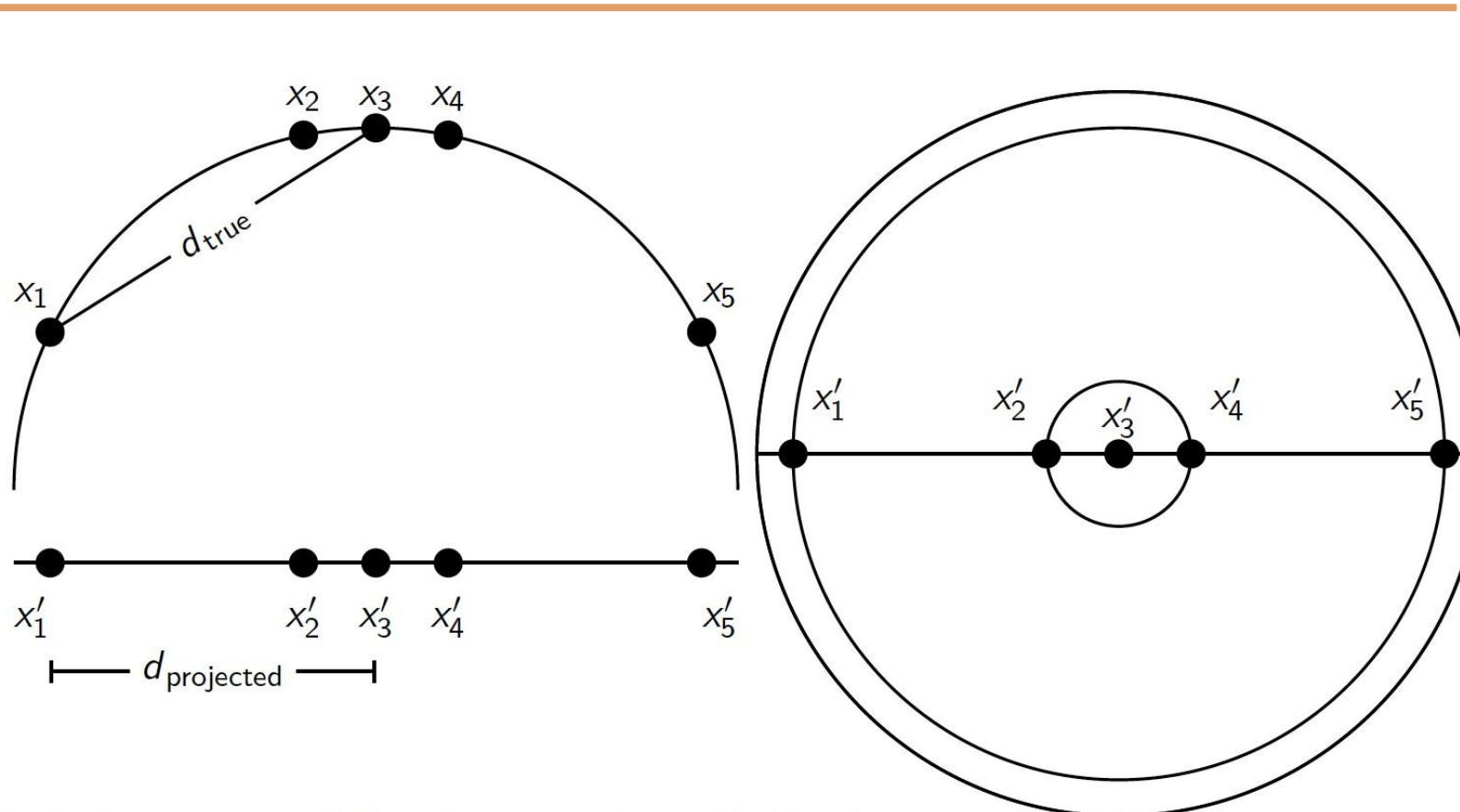
---

- As before, the training set is a set of documents, each labeled with its class (e.g., topic)
- In vector space classification, this set corresponds to a labeled set of points (or, equivalently, vectors) in the vector space
- Premise 1: Documents in the same class form a contiguous region of space
- Premise 2: Documents from different classes don't overlap (much)
- We define surfaces to delineate classes in the space

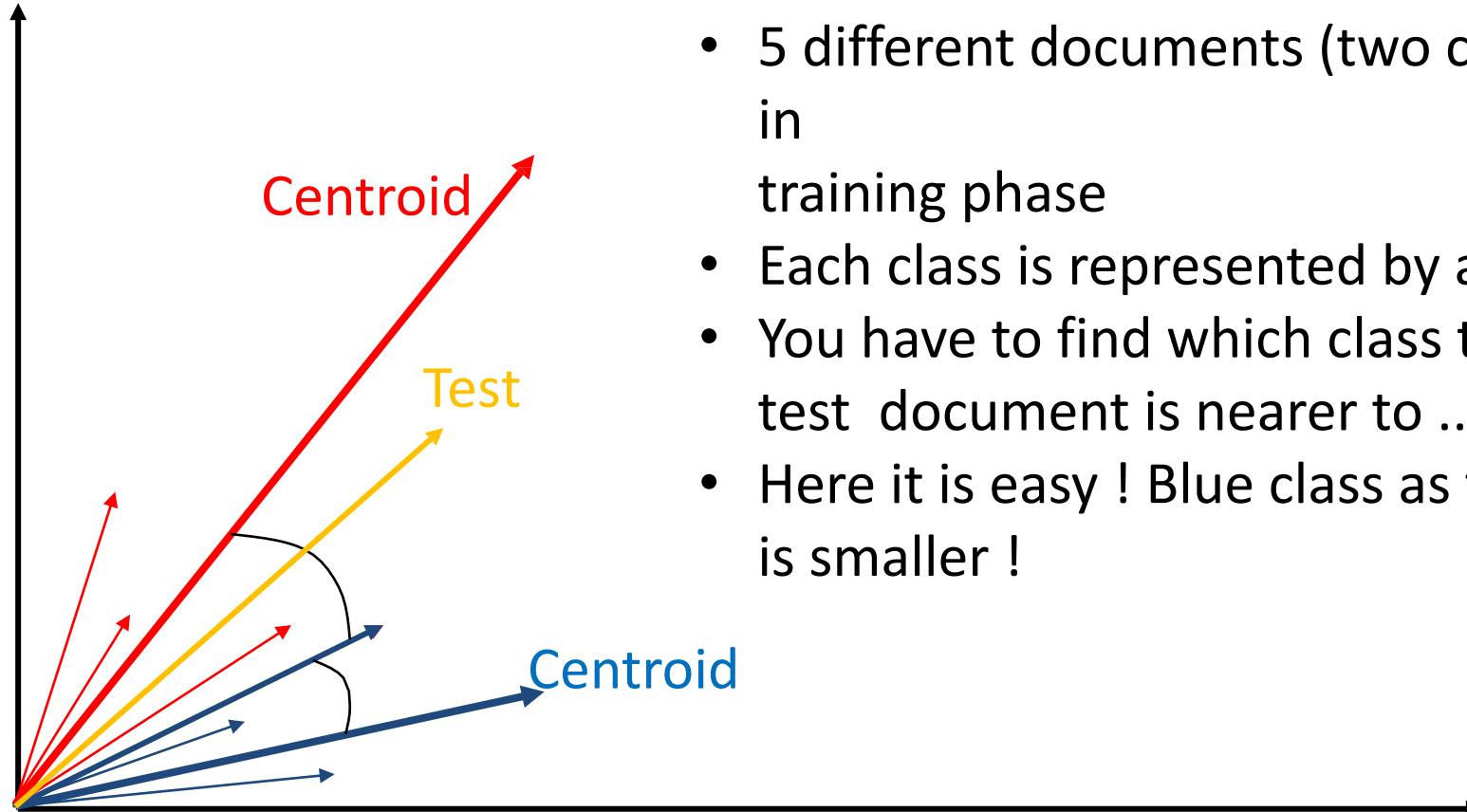








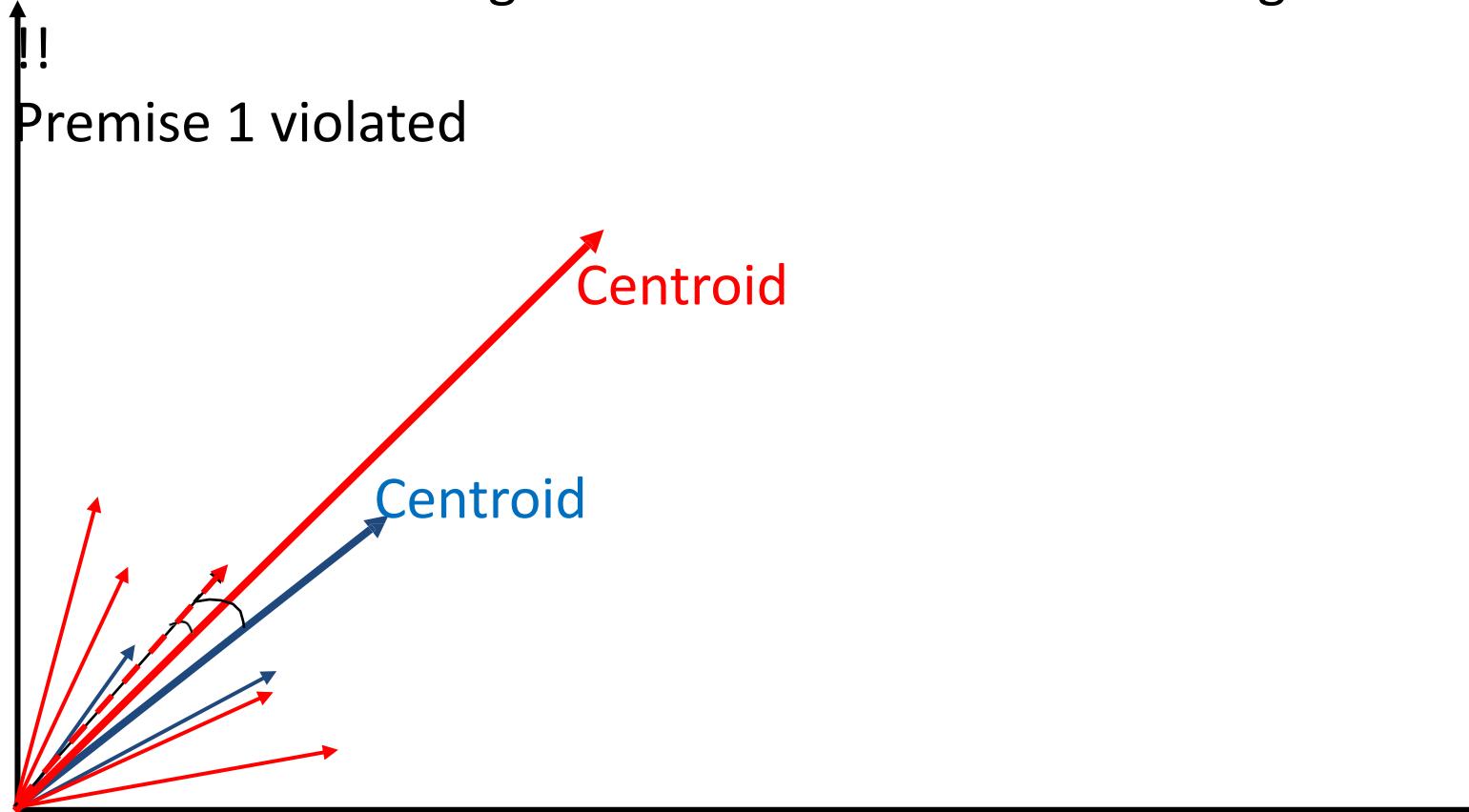
Left: A projection of the 2D semicircle to 1D. For the points  $x_1, x_2, x_3, x_4, x_5$  at  $x$  coordinates  $-0.9, -0.2, 0, 0.2, 0.9$  the distance  $|x_2x_3| \approx 0.201$  only differs by 0.5% from  $|x'_2x'_3| = 0.2$ ; but  $|x_1x_3|/|x'_1x'_3| = d_{\text{true}}/d_{\text{projected}} \approx 1.06/0.9 \approx 1.18$  is an example of a large distortion (18%) when projecting a large area. Right: The corresponding projection of the 3D hemisphere to 2D.



- 5 different documents (two classes) in training phase
- Each class is represented by a centroid
- You have to find which class the test document is nearer to ..
- Here it is easy ! Blue class as the angle is smaller !

## Rocchio Anomaly

- Prototype models have problems with polymorphic (disjunctive) categories.
- Red centroid falls right in the middle of the blue region
- Premise 1 violated



# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

**Rocchio Algorithm  
for Relevance  
Feedback**

**Bhaskarjyoti Das**  
Department of Computer Science Engineering

## Key concept for relevance feedback: Centroid

---

- The centroid is the center of mass of a set of points.
- Recall that we represent documents as vectors (tips as points) in a high-dimensional space.
- Thus: we can compute centroids of documents.

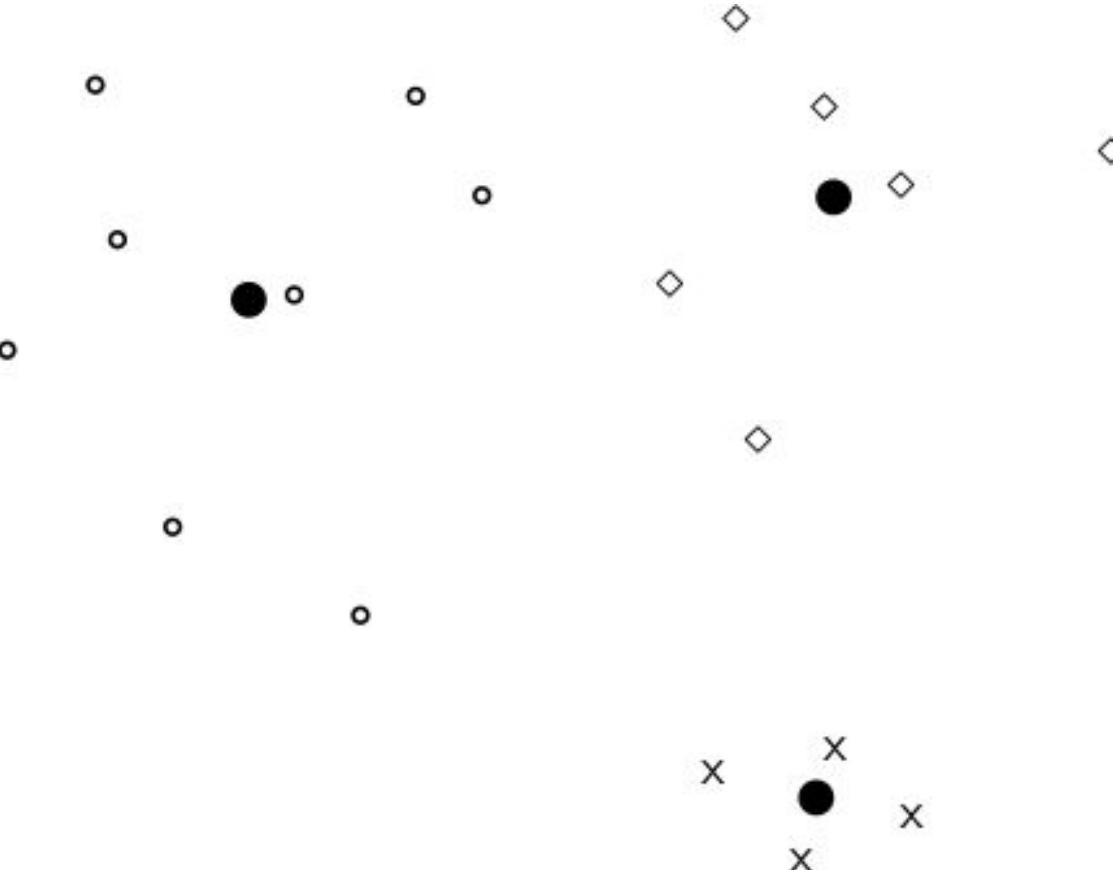
- Definition:

$$\vec{\mu}(D) = \frac{1}{|D|} \sum_{d \in D} \vec{v}(d)$$

where  $D$  is a set of documents and  $\vec{v}(d)$  is the vector we use to represent document  $d$ .

*Note that centroid will in general not be a unit vector even when the inputs are unit vectors.*

## Centroid: Example



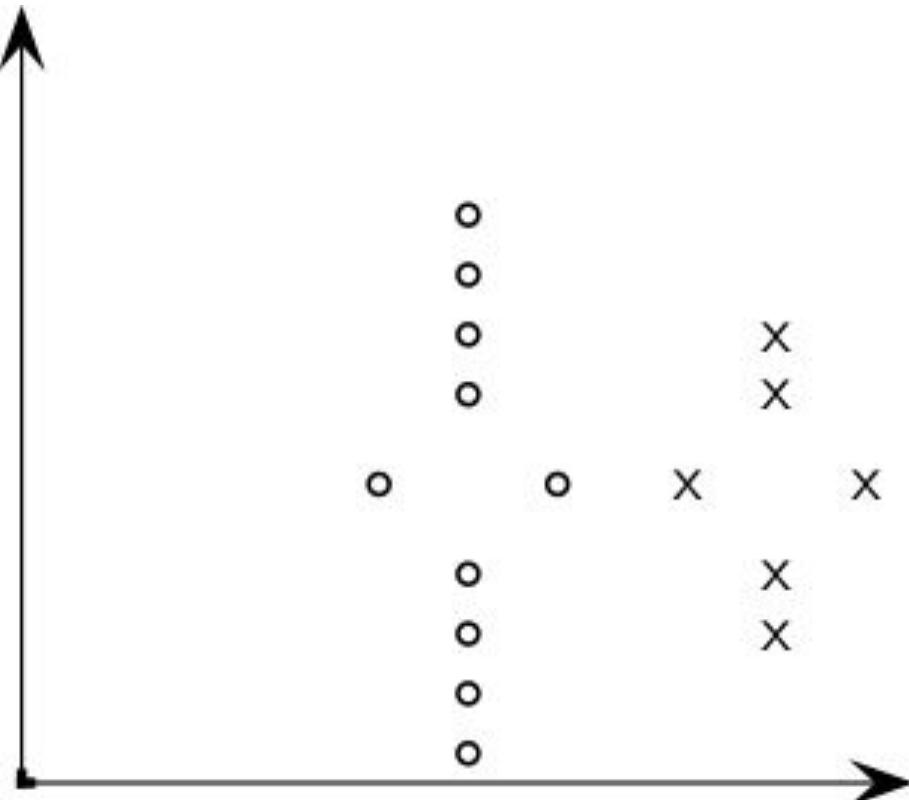
## Rocchio algorithm for Relevance Feedback

- The Rocchio algorithm can be used to implement relevance feedback in the vector space model.
- $D_r$  is the set of documents user marked as relevant and  $D_{nr}$  is the set that user marked as irrelevant
- The idea : select a modified query that will move towards the centroid of  $D_r$  and further away from the centroid of  $D_{nr}$
- Rocchio' chooses the query  $\vec{q}_{opt}$  at maximizes

$$\vec{q}_{opt} = \arg \max_{\vec{q}} [\text{sim}(\vec{q}, \mu(D_r)) - \text{sim}(\vec{q}, \mu(D_{nr}))]$$

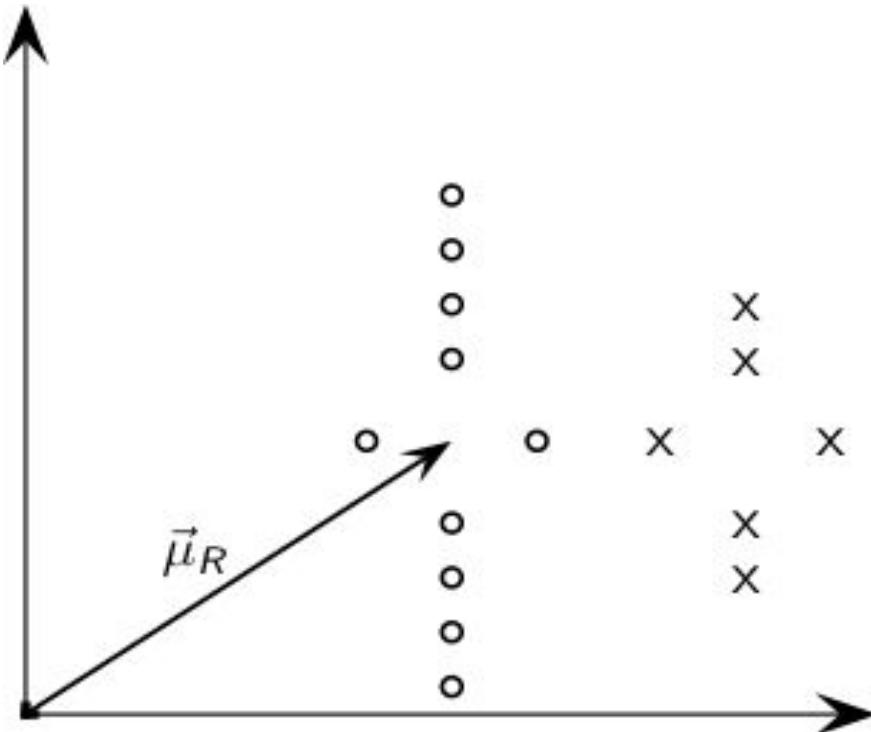
## Exercise : Compute Rocchio' vector

---



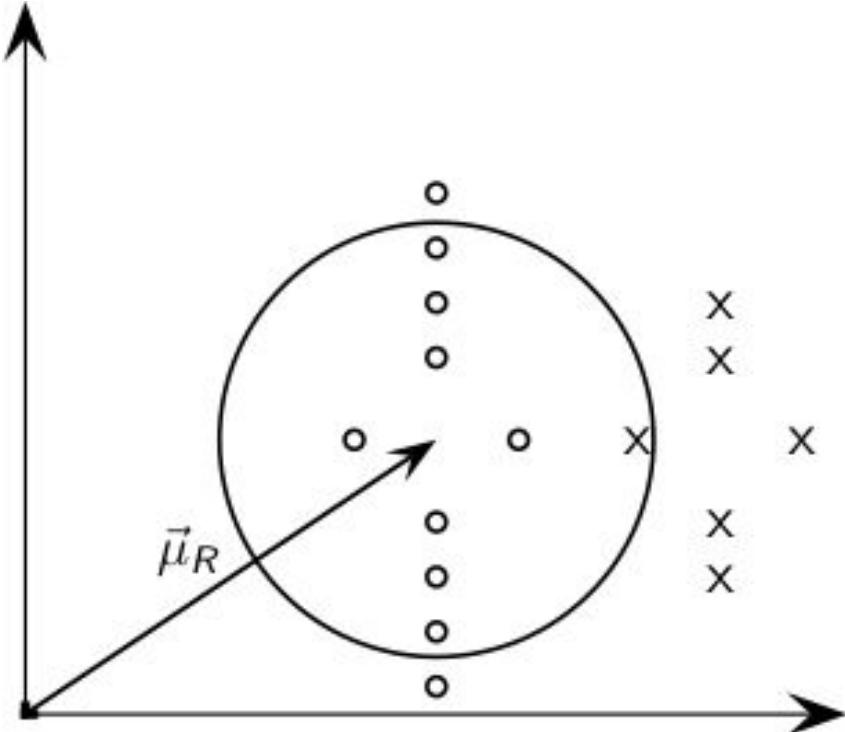
circles: relevant documents, Xs: nonrelevant documents

## Rocchio' illustrated



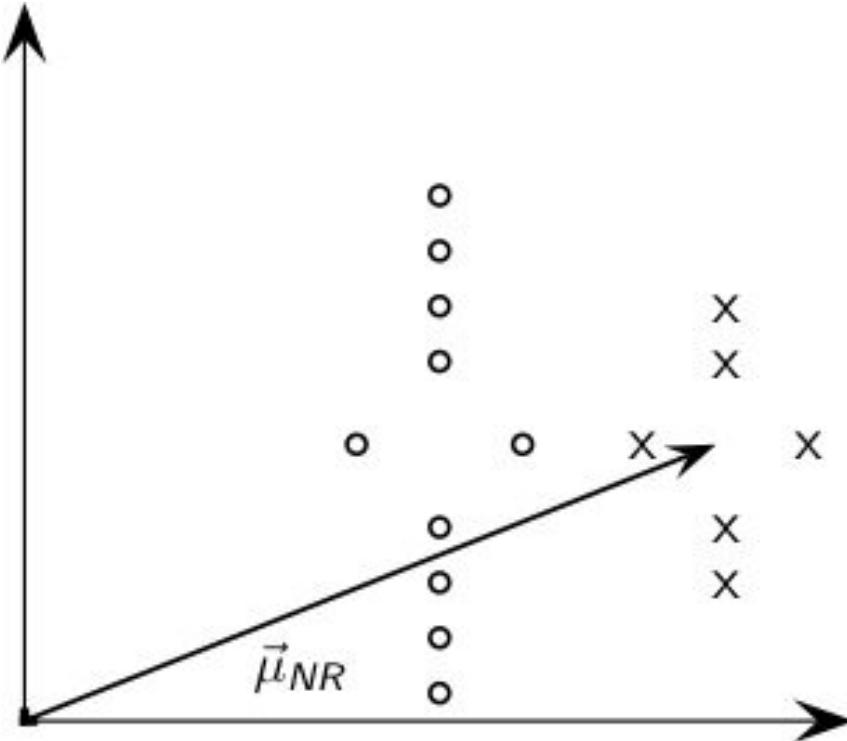
$\vec{\mu}_R$  : centroid of relevant documents

## Rocchio' illustrated



$\vec{\mu}_R$  does not separate relevant / nonrelevant.

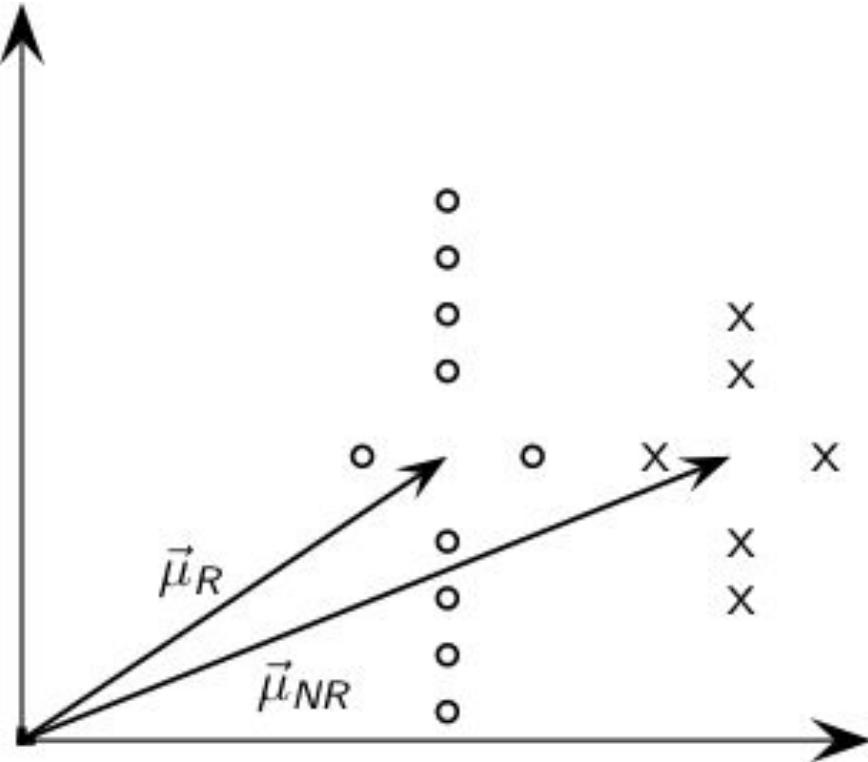
## Rocchio' illustrated



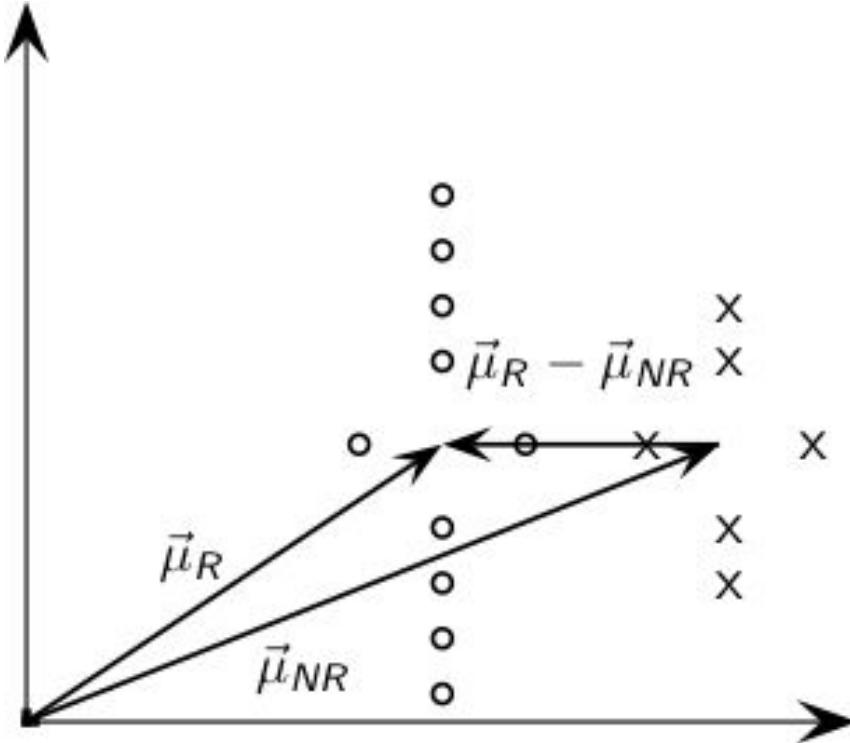
$\vec{\mu}_{NR}$ : centroid of nonrelevant documents.

## Rocchio' illustrated

---

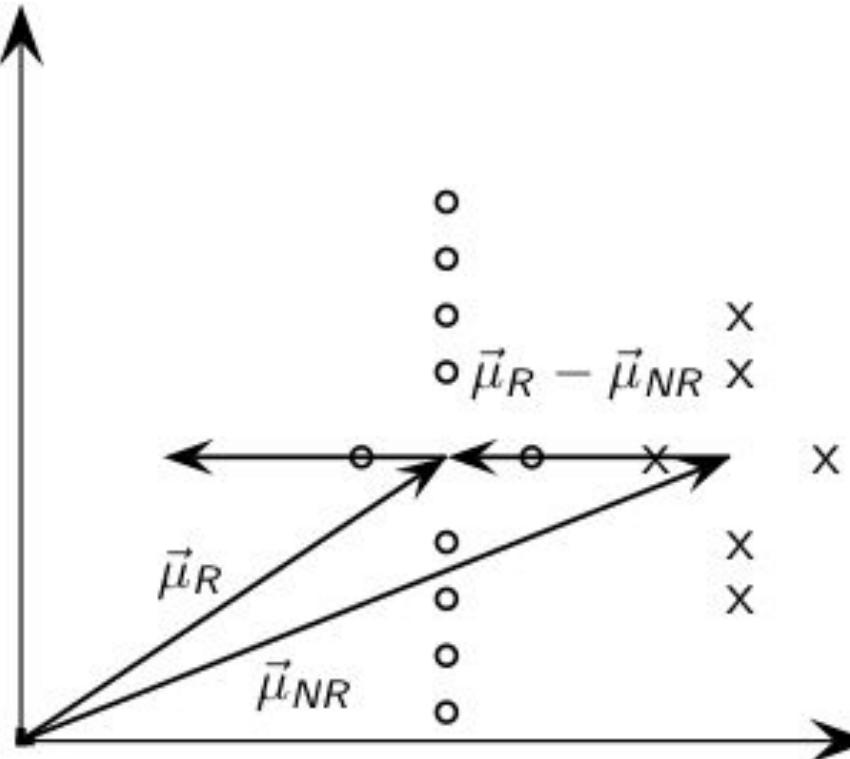


## Rocchio' illustrated



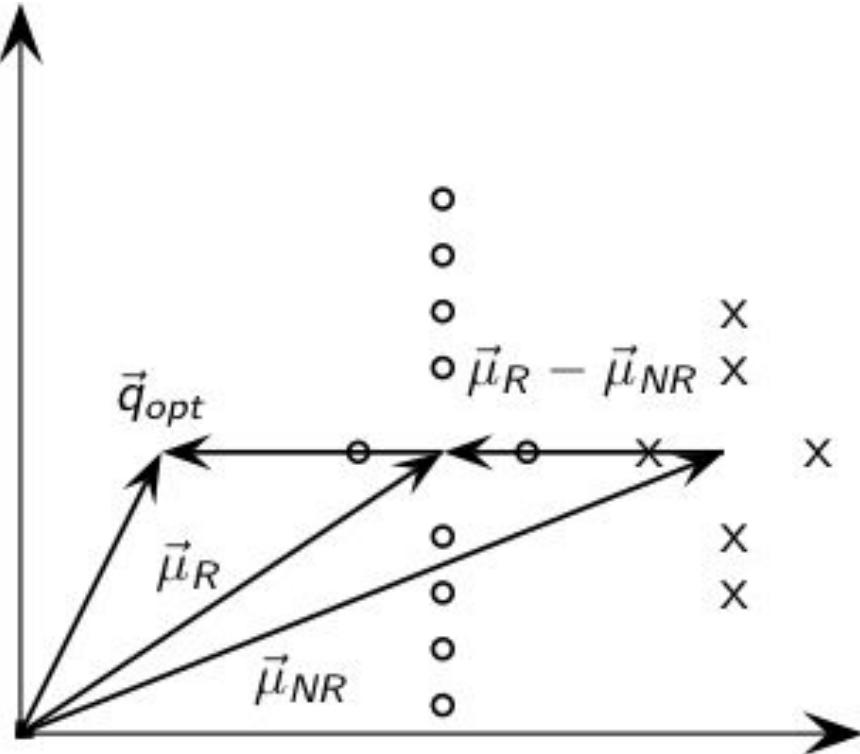
$\vec{\mu}_R - \vec{\mu}_{NR}$ : difference vector

## Rocchio' illustrated



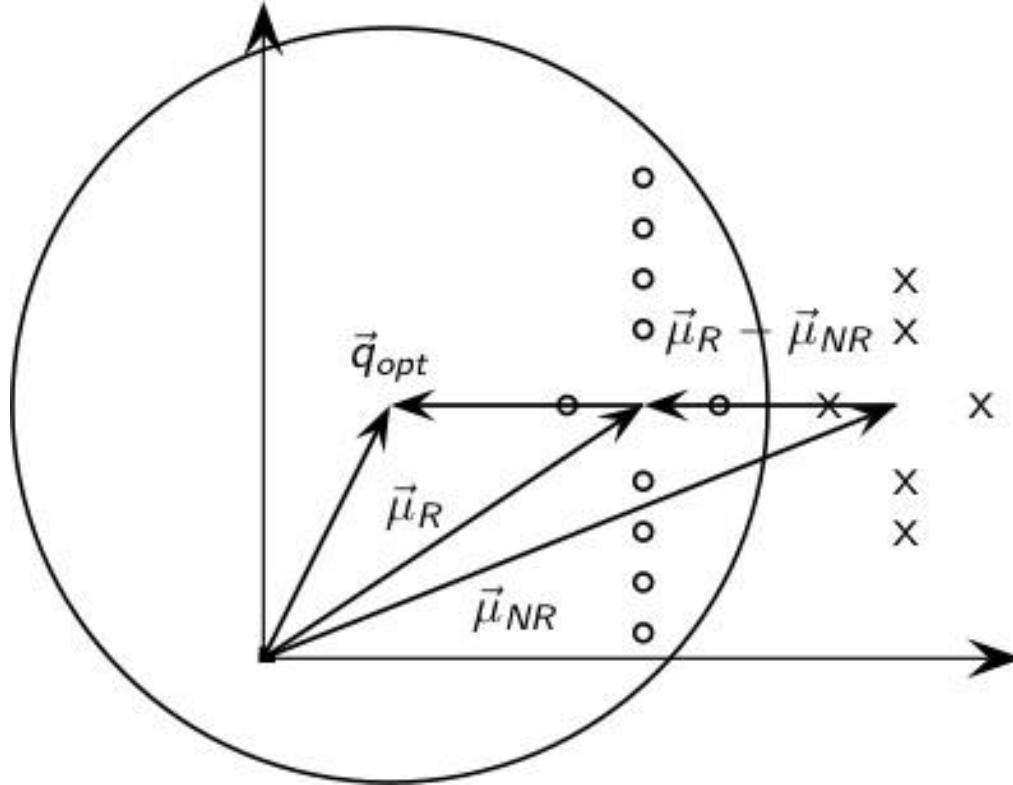
Add difference vector to  $\vec{\mu}_R$  ...

## Rocchio' illustrated



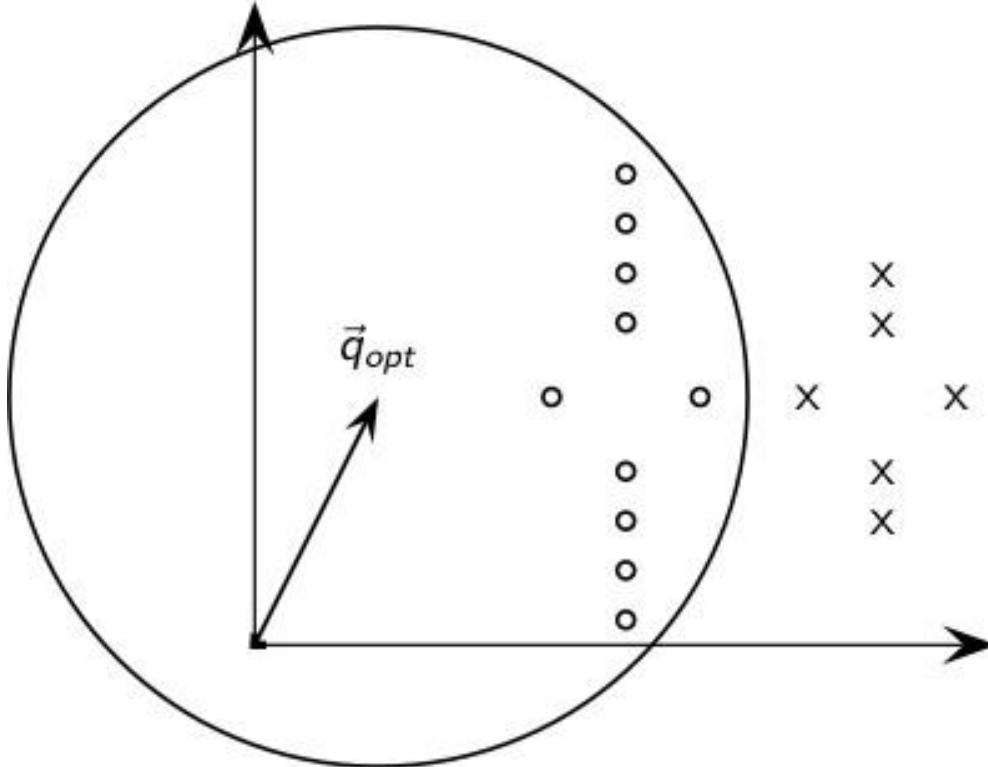
... to       $\vec{q}_{opt}$   
get

## Rocchio' illustrated



$\vec{q}_{opt}$  separates relevant / nonrelevant perfectly.

## Rocchio' illustrated



$\vec{q}_{opt}$  separates relevant / nonrelevant perfectly.



THANK  
YOU

---

**Bhaskarjyoti Das**  
Department of Computer Science  
Engineering



**PES**  
**UNIVERSITY**

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

## Relevance Feedback

---

**Bhaskarjyoti Das**  
Department of Computer Science  
Engineering

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

**Rocchio' and Pseudo  
Relevance  
Feedback**

**Bhaskarjyoti Das**  
Department of Computer Science Engineering

## Terminology

---

- We use the name ‘Rocchio’ for the theoretically better motivated original version of Rocchio.
- The implementation that is actually used in most cases is the SMART implementation – we use the name Rocchio (without prime) for that.

## Rocchio's algorithm

---

- The optimal query vector is:

$$\begin{aligned}\vec{q}_{opt} &= \mu(D_r) + [\mu(D_r) - \mu(D_{nr})] \\ &= \frac{1}{|D_r|} \sum_{\vec{d}_j \in D_r} \vec{d}_j + \left[ \frac{1}{|D_r|} \sum_{\vec{d}_j \in D_r} \vec{d}_j - \frac{1}{|D_{nr}|} \sum_{\vec{d}_j \in D_{nr}} \vec{d}_j \right]\end{aligned}$$

- We move the centroid of the relevant documents by the difference between the two centroids (relevant and irrelevant documents).

## Rocchio's 1971 algorithm(SMART by Salton)

---

Used in practice:

$$\begin{aligned}\vec{q}_m &= \alpha \vec{q}_0 + \beta \mu(D_r) - \gamma \mu(D_{nr}) \\ &= \alpha \vec{q}_0 + \beta \frac{1}{|D_r|} \sum_{\vec{d}_j \in D_r} \vec{d}_j - \gamma \frac{1}{|D_{nr}|} \sum_{\vec{d}_j \in D_{nr}} \vec{d}_j\end{aligned}$$

$\vec{q}_m$ : modified query vector;  $\vec{q}_0$ : original query vector

$D_r$  and  $D_{nr}$ : sets of known relevant and nonrelevant

documents respectively

$\alpha, \beta$ , and  $\gamma$ : weights

- New query moves towards relevant documents and away from nonrelevant documents.
- Adding a little bit of the centroid of the relevant and subtracting a little bit of the centroid of the irrelevant

## Rocchio's 1971 algorithm(SMART)

---

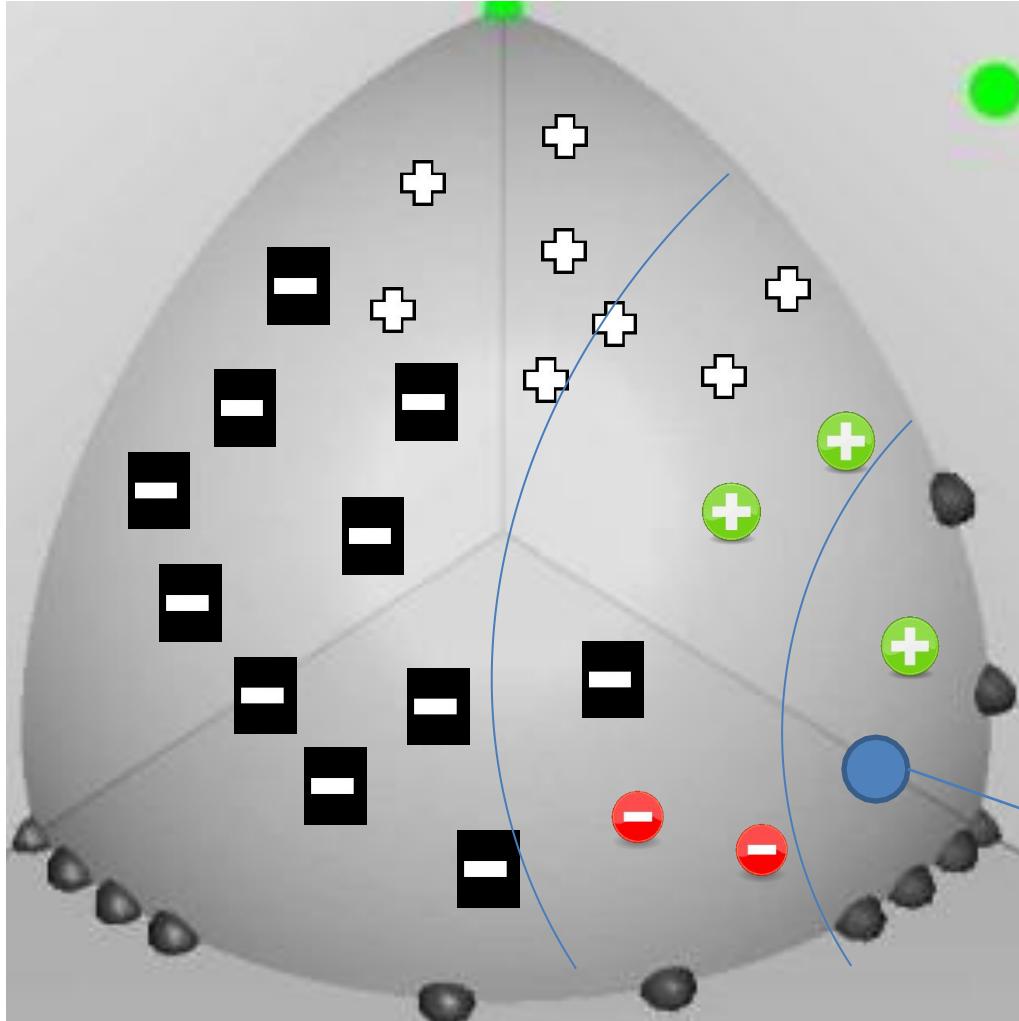
- Tradeoff  $\alpha$  vs.  $\beta/\gamma$ : If we have a lot of judged documents, we want a higher  $\beta/\gamma$ .
- Set negative term weights to 0.
  - “Negative weight” for a term doesn’t make sense in the vector space model.

## Rocchio's 1971 algorithm(SMART) - Example

---

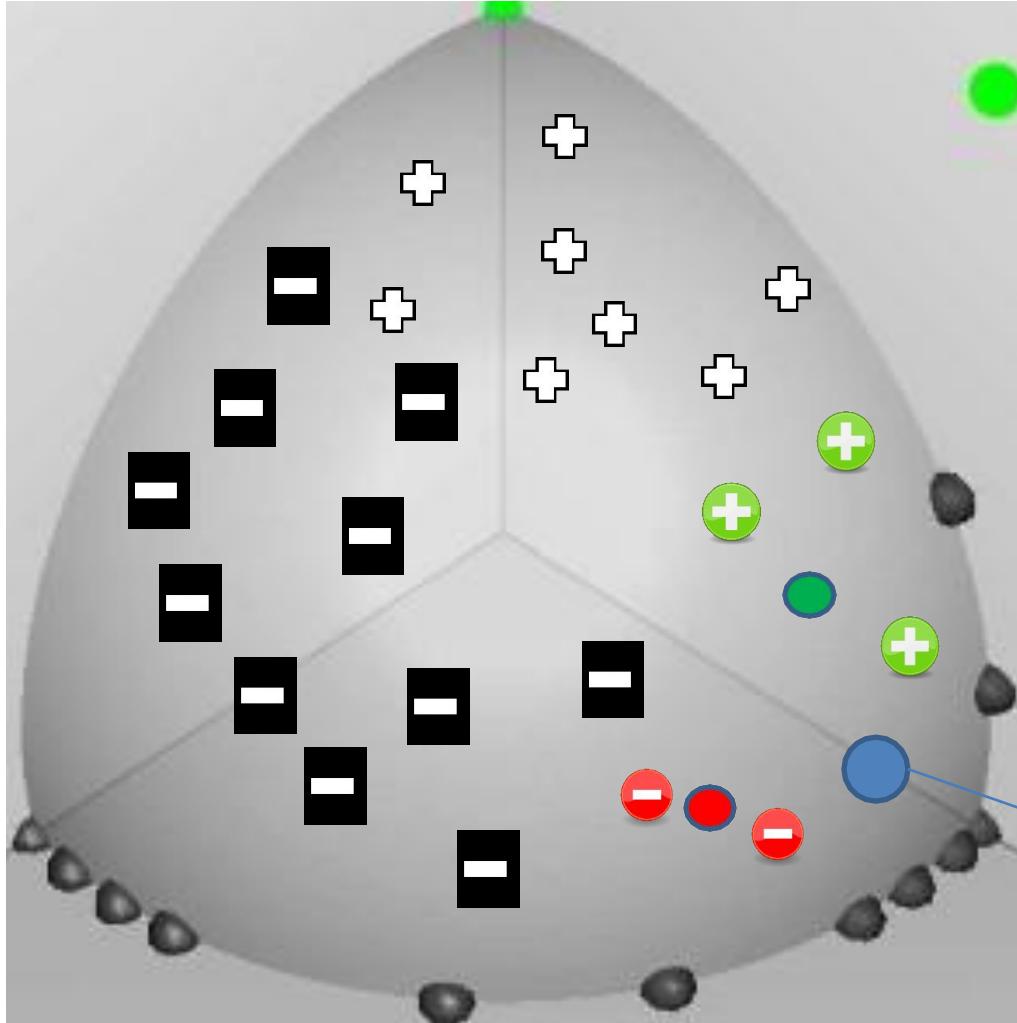
- Vocabulary  $\{t_1, t_2, t_3, t_4, t_5\}$
- Original Query  $q_0 = [1, 0, 1, 0, 0]$
- Relevant Document  $d_r = [2, 2, 1, 0, 0]$
- Non relevant document  $d_n = [2, 0, 1, 0, 3]$
- $\alpha = 1, \beta = 1, \gamma = 0.5$
- No of relevant documents = of nonrelevant documents = 1
  - $q_1 = q_0 + 1.0 * d_r - 0.5 * d_n$ 
    - $= [1, 0, 1, 0, 0] + [2, 2, 1, 0, 0] - 0.5 * [2, 0, 1, 0, 3]$
    - $= [2, 2, 1.5, 0, -1.5]$
    - $= [2, 2, 1.5, 0, 0]$
    - $= [2*t_1, 2*t_2, 1.5*t_3]$

## Vector space example: Having Relevant and Irrelevant Documents



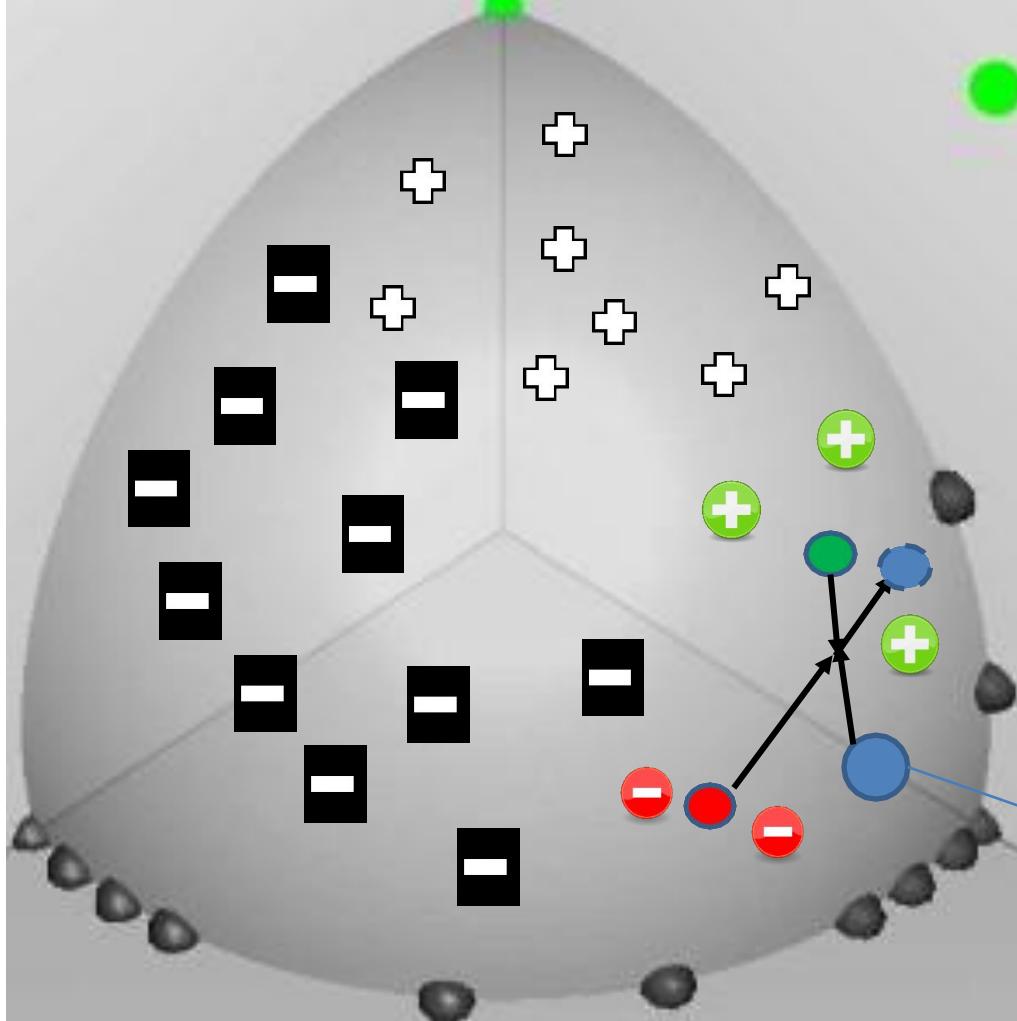
- Two classes of documents that are contiguous
- User has run one query  $Q_0$
- User marked two documents as relevant and two as irrelevant
- Blue iso lines (same magnitude)  
Query  $Q_0$

## Vector space example: Having Relevant and Irrelevant Documents



- Two Centroids are calculated

## Vector space example: Having Relevant and Irrelevant Documents



- The query is interpolated some extent to the positive centroid
- The interpolated position is then moved further away from the negative centroid

Query  $Q_0$

## Positive vs. Negative relevance feedback

---

- Positive feedback is more valuable than negative feedback.
- For example, set  $\beta = 0.75$ ,  $\gamma = 0.25$  to give higher weight to positive feedback.
- Many systems only allow positive feedback.

## Relevance feedback : Assumptions

---

- When can relevance feedback enhance recall?
- Assumption A1: The user knows the terms in the collection well enough for an initial query.
- Assumption A2: Relevant documents similar to ~~contain~~ (so I can “hop” from one relevant document to a different one when giving relevance feedback).

## Violation of A1

---



- **Assumption A1:** The user knows the terms in the collection well enough for an initial query.
- **Violation:** Mismatch of searcher's vocabulary and collection vocabulary
- Example: cosmonaut / astronaut

## Violation of A2

---

- **Assumption A2:** Relevant documents are similar.
  - Example : Contradictory Govt Policies
    - Subsidies for tobacco farmers vs. anti-smoking campaigns
    - Aid for developing countries vs. high tariffs on imports from developing countries
  - Relevance feedback on tobacco docs will not help with finding docs on developing countries.

## Relevance feedback : Evaluation

---

- Pick one of the evaluation measures e.g., precision in top 10:  $P@10$ 
  - Compute  $P@10$  for original query  $q_0$
  - Compute  $P@10$  for modified relevance feedback query  $q_1$
  - In most cases:  $q_1$  is spectacularly better than  $q_0$ !
- Is this a fair evaluation?

## Relevance feedback : Evaluation

---

- Fair evaluation must be on “residual” collection: docs not yet judged by user.
  - Studies have shown that relevance feedback is successful when evaluated this way.
  - Empirically, one round of relevance feedback is often very useful. Two rounds are marginally useful.

## Evaluation: Caveat

---

- True evaluation of usefulness must compare to other methods taking the same amount of time.
  - Alternative to relevance feedback: User revises and resubmits query.
  - Users may prefer revision/resubmission to having to judge relevance of documents.
- There is no clear evidence that relevance feedback is the “best use” of the user’s time.

## Relevance feedback: Problems

---

- Relevance feedback is expensive.
  - Relevance feedback creates long modified queries.
  - Long queries are expensive to process.
- Users are reluctant to provide explicit feedback
  - Users refuse to do anything that is essential for their work
  - If Search Engines force them to do that, the result may be worse ( users leave !)
- It's often hard to understand why a particular document was retrieved after applying relevance feedback.
- The search engine Excite had full relevance feedback at one point, but abandoned it later.

## Pseudo-relevance feedback

---

- Pseudo-relevance feedback automates the “manual” part of true relevance feedback
- Top ranked documents are usually relevant (observed on average)
  - In weakly supervised tasks, we bootstrap with few labelled examples
  - Same Trick in Pseudo Relevance : Run Rocchio with  $\gamma=0$
- Equivalent to massive query expansion with added words from top ranked documents to query
  - Though it may be a challenge to control the query drift, pseudo relevance is one of the most effective trick !

## Pseudo-relevance feedback

---

- Works very well on average
- But can go horribly wrong for some queries.
- Several iterations can cause [query drift](#).

## Pseudo-relevance feedback at TREC4

---

- Cornell SMART system
- Results show number of relevant documents out of top 100 for 50 queries (so total number of documents is 5000):

method	number of relevant documents
Inc.Itc	3210
Inc.Itc-PsRF	3634
Lnu.Itu	3709
Lnu.Itu-PsRF	4350



THANK  
YOU

---

**Bhaskarjyoti Das**  
Department of Computer Science  
Engineering



**PES**  
UNIVERSITY

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

## Query Expansion

---

**Bhaskarjyoti Das**  
Department of Computer Science  
Engineering

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

## Query Expansion

**Bhaskarjyoti Das**

Department of Computer Science Engineering

## Query Expansion

---

- Query expansion is another method for increasing recall.
- We use “global query expansion” to refer to “global methods for query reformulation”.
  - In global query expansion, the query is modified based on some global resource, i.e. a resource that is not query-dependent.

## Types of user feedback

---

- User gives feedback on documents.
- More common in relevance feedback
- User gives feedback on words or phrases.
- More common in query expansion

## Query Expansion

---

- Main information we use: **(near-)synonymy**
- A publication or database that collects (near-)synonyms is called a **thesaurus**.
- **types of thesauri:**
  - manually created
  - automatically created
  - Query log based

## Types of query expansion

---

- Manual thesaurus (maintained by editors, e.g., PubMed)
- Automatically derived thesaurus (e.g., based on co-occurrence statistics)
- Query-equivalence based on query log mining

## Thesaurus-based query expansion

---

- For each term  $t$  in the query, expand the query with words the thesaurus lists as semantically related with  $t$ .
  - Example from earlier: HOSPITAL → MEDICAL
- Generally increases recall
- May significantly decrease precision, particularly with ambiguous terms
  - INTEREST RATE → INTEREST RATE FASCINATE

## Thesaurus-based query expansion

---

- Widely used in specialized search engines for science and engineering
- It's very expensive to create a manual thesaurus and to maintain it over time.
- A manual thesaurus has an effect roughly equivalent to annotation with a controlled vocabulary.

## Example for manual thesaurus: PubMed



The screenshot shows the PubMed search interface. The top navigation bar includes links for PubMed, Nucleotide, Protein, Genome, Structure, PopSet, and Taxonomy. The search bar contains the text "Search PubMed for cancer". Below the search bar are buttons for Go, Clear, Limits, Preview/Index, History, Clipboard, and Details. On the left sidebar, there is a link to "About Entrez". The main content area displays the "PubMed Query:" as: ("neoplasms"[MeSH Terms] OR cancer[Text Word]). At the bottom of the search interface, there are "Search" and "URL" buttons.

## Automatic thesaurus generation

---

- Attempt to generate a thesaurus automatically by analyzing the distribution of words in documents
  - Fundamental notion: similarity between two words
- Definition 1: Two words are similar if they co-occur with similar words.
  - “car” ≈ “motorcycle” because both occur with “road”, “gas” and “license”, so they must be similar.
- Definition 2: Two words are similar if they occur in a given grammatical relation with the same words.
  - You can harvest, peel, eat, prepare, etc. apples and pears, so apples and pears must be similar.
- Co-occurrence is more robust, grammatical relations are more accurate.

## Co-occurrence-based thesaurus : Examples

Word	Nearest neighbors
absolutely	absurd whatsoever totally exactly nothing
bottomed	dip copper drops topped slide trimmed
captivating	shimmer stunningly superbly plucky witty
doghouse	dog porch crawling beside downstairs
makeup	repellent lotion glossy sunscreen skin gel
mediating	reconciliation negotiate case conciliation
keeping	hoping bring wiping could some would
lithographs	drawings Picasso Dali sculptures Gauguin
pathogens	toxins bacteria organisms bacterial parasite
senses	grasp psyche truly clumsy naive innate

WordSpace demo on web

## Query expansion at Search Engines Based On Query Log

---

- Main source of query expansion at search engines: query logs
  - Example 1: After issuing the query [herbs], users frequently search for [herbal remedies].
    - → “herbal remedies” is potential expansion of “herb”.
  - Example 2: Users searching for [flower pix] frequently click on the URL [photobucket.com/flower](http://photobucket.com/flower). Users searching for [flower clipart] frequently click on the [same URL](#).
    - → “flower clipart” and “flower pix” are potential expansions of each other.

## Take-away today

---

- **Interactive relevance feedback:** improve initial retrieval results by telling the IR system which docs are relevant / nonrelevant
  - Best known relevance feedback method: Rocchio feedback
- **Query expansion:** improve retrieval results by adding synonyms / related terms to the query
  - **Sources for related terms:** Manual thesauri, automatic thesauri, query logs



**PES**  
**UNIVERSITY**

**THANK  
YOU**

---

**Bhaskarjyoti Das**  
Department of Computer Science  
Engineering



# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

## Web Search Basics

---

**Bhaskarjyoti Das**

Department of Computer Science Engineering

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

## Advertising as an Economic Model – General Overview

**Bhaskarjyoti Das**  
Department of Computer Science Engineering

## Cost per Impression (CPM) Model – Derived from the Billboard Model

primary purpose of these was branding or awareness:

- to convey to the viewer a positive feeling about the brand of the company placing the ad
- to make some awareness
- Measured in cost per thousand impression (priced very low)

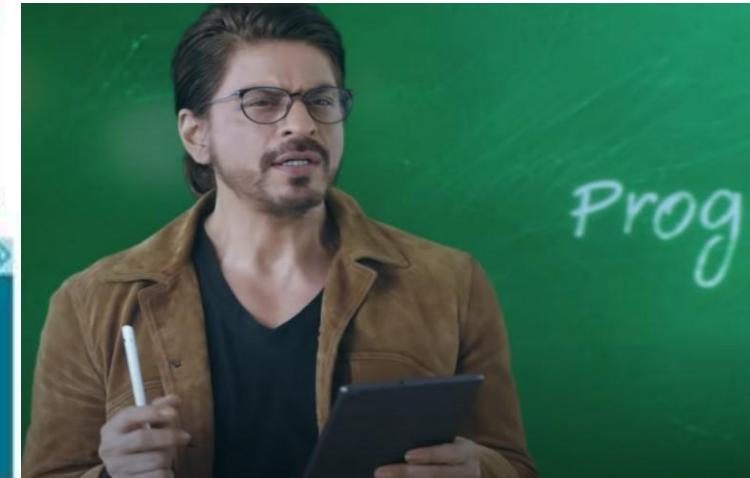
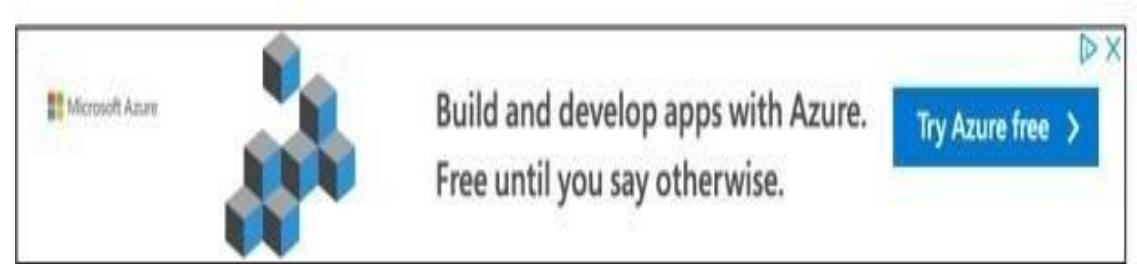


## Advertising as an Economic Model – Display or Banner Ad

The primary purpose :

- to convey to the viewer a positive feeling about the brand of the company placing the ad.

Television and Newspaper ads are also banner ads !



Shahrukh Khan @ Byju

## Do Search Engine Vendors Deal with Display Ad ?

---

- Yahoo! was a portal + search engine. So, it used to have lot of display ads (and display ad revenue).
- What about Google ?
  - People are served display ads while they're consuming content—not while they're actively looking for solutions.
    - Primary goal is branding and brand awareness
  - Google display ads are served on the Google Display Network—a network of over two million websites and apps that reaches somewhere in the ballpark of 90% of internet users.
  - Both uploaded and responsive ads
  - Just like search, Google display network runs on a live auction system

## Cost Per Click (CPC) Model

---

- Advertiser is charged when their advertisement gets clicked
- Advertisements are offered on search pages and offered by other publishers
- Affiliate Marketing Programs on publishers' webpages
  - It's free for website owners and bloggers to become **Amazon Affiliates**
  - **YouTube Affiliate Marketing** : YouTube Affiliate will make videos with the aim of getting you to buy products from a 3rd party site, such as Amazon

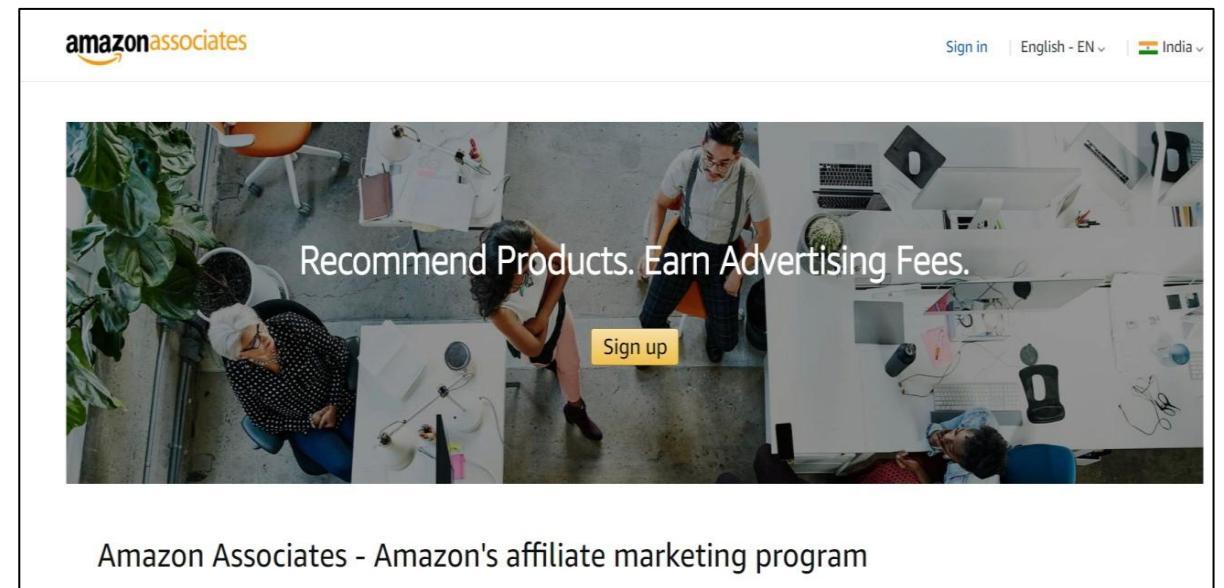
## Cost Per Action / Acquisition Model

- In this model, the **advertiser** will not pay for impression or click but **will pay for some action** i.e. buying a product

- **Amazon Associate** program

- Bloggers, publishers and content creators with a **qualifying website** or mobile app can participate

- **Amazon Associates** advertise products from **Amazon.com** on their websites by creating links. When customers click the links and buy products from **Amazon**, they earn referral fees.



## Which Rate should be Higher ?

---

- 1% of viewers click on an ad : Clickthrough rate ( CTR ) = 1%
- 1% of people who click, actually convert i.e. buy the product : Conversion rate = 1%
- For every 1000 impressions, there are 10 click throughs leading to 0.1 conversion
- So if the **advertiser** pays  $x$  to the **publisher** for thousand impressions,
- CPM model i.e. **Cost per impression** is  $x/1000 = 0.001 * x$
- CPC model i.e. Pay  $x$  for 10 clicks or **cost /click** =  $x/10 = 0.1 * x$
- CPA model i.e. Pay  $x$  for 0.1 conversion or **cost/action** =  $x/0.1 = 10 * x$

## Which Model is Most Risky or Most Suitable for Advertiser?

---

- **CPM is most Risky (for advertiser) and CPA is least risky for advertiser.**  
CPM is most risky as most of them need not be converted . Most money is spent on showing ad to people who may not be interested
- **CPC is in between** as advertiser is paying publisher at least when the people interested are clicking ( but risky still as CTR i.e. clickthrough rate may not yield enough buy)
- If the risk needs to be divided between advertiser and publisher, CPC is the chosen model !

## Which Model is Most Risky or Most Suitable for Publisher?

---

- It depends !
  - For the publisher, CPA is much more than CPM but very infrequent. So, he will prefer CPM
  - For a publisher (blogger or youtuber) trying to make some money (and he is not having millions of footfall), the advertiser will choose least risk (CPA) and publisher will have no option to negotiate
  - Google is also a publisher of some kind on its search page – will it agree for CPA ? May not be as it is the biggest publisher !

## Pay per Click ( PPC) vs. Cost per Click ( CPC)

---

- Advertisers bid for “keywords”. Ads for highest bidders displayed when user query contains a purchased keyword
- PPC vs. CPC – PPC is the approach and CPC is the metric or measure. To figure out how the PPC campaign is performing, start by measuring CPC.
- In some cases, it's helpful to actually *increase* cost per click budget if it helps reach a more qualified audience or if it helps you rank above key competitors.

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

**Advertising as an Economic Model  
For Search Engines – Search Ads**

**Bhaskarjyoti Das**  
Department of Computer Science Engineering

## Why Search Ads are Attractive ?

---

- Why is web search potentially more attractive for advertisers than TV spots, newspaper ads or radio spots?
  - Someone who just searched for “Saturn Aura Sport Sedan” is infinitely more likely to buy one than a random person watching TV.
  - Most importantly, the advertiser only pays if the customer took an action indicating interest (i.e., clicking on the ad)

## Search Ads : WIN-WIN-WIN ?

---

- Three parties : Publisher, Advertiser, Buyer
- The **Publisher** (search engine company) gets revenue every time somebody clicks on an ad.
- The **user** only clicks on an ad if they are interested in the ad.
  - Search engines punish misleading and **nonrelevant ads**.
  - As a result, **users are often satisfied** with what they find after clicking on an ad.
- The **advertiser** finds new customers in a cost-effective way

## When is it not Win-Win-Win ?

---

Issue : keyword arbitrage

- Buy a keyword at Google and redirect to a 3<sup>rd</sup> party that is paying more than you have to pay to Google! This may not make sense for Google as these are biased publishers !
- Click Spam – Also known as organics poaching, click spam is a type of advertising fraud that happens when a publisher executes clicks for users who haven't made them
- A devious advertiser may attempt to exhaust the advertising budget of a competitor by clicking repeatedly (through robotic click generator) on his sponsored search ads.

## Sponsored Search

---

Model brought in by Goto:

- For every query term  $q$ , it accepted bids from companies who wanted their web page shown on the query  $q$ . In response to the query  $q$ , Goto would return the pages of all advertisers who bid for  $q$ , ordered by their bids.
- Furthermore, when the user clicked on one of the returned results, the corresponding advertiser would make a payment to Goto (in the initial implementation, this payment equaled the advertiser's bid for  $q$ ).

## First Generation of Search Ad (Goto 1996) – Sponsored Search

No separation of ads/docs.  
Just one result!

Buddy Blake bid the maximum (\$0.38) for this search, paid \$0.38 to Goto every time somebody clicked on the link

Upfront and honest. No relevance ranking, but Goto did not pretend there was any.



## Keywords with High Bids – Old Data but This is the Model

---

- According to

<http://www.cwire.org/highest-paying-search-terms/>

\$69.1 mesothelioma treatment options  
\$65.9 personal injury lawyer michigan  
\$62.6 student loans consolidation  
\$61.4 car accident attorney los angeles  
\$59.4 online car insurance quotes  
\$59.4 arizona dui lawyer  
\$46.4 asbestos cancer  
\$40.1 home equity line of credit  
\$39.8 life insurance quotes  
\$39.2 refinancing  
\$38.7 equity line of credit  
\$38.0 lasik eye surgery new york city  
\$37.0 2nd mortgage  
\$35.9 free car insurance quote

## Combining Algorithmic Search with Sponsored Search

---

- Given these two kinds of search engines – the “**pure/algorithmic**” **search** engines such as **Google Altavista**, versus the **sponsored search** engines – the logical next step was to combine them into a single user experience.
- Next Generation search engines follow precisely this model:
  - they provide pure search results (generally known as **algorithmic search results**) as the primary response to a user’s search,
  - together with sponsored search results displayed separately and distinctively to the right of the algorithmic results

## Next Generation : Sponsored Search and Algorithmic Search

nigritude ultramarine - Google Search - Mozilla Firefox

File Edit View Go Bookmarks Yahoo! Tools Help

Getting Started Latest Headlines

Yahoo! Search Web Mail My Yahoo! Games Movies Music Answers Personals Sign In

pragh60@gmail.com | My Account | Sign out

Google Web Images Groups News Froogle Local more »

nigritude ultramarine Search Advanced Search Preferences

Results 1 - 10 of about 185,000 for **nigritude ultramarine**. (0.35 seconds)

**Anil Dash: Nigritude Ultramarine**  
Do me a favor: Link to this post with the phrase **Nigritude Ultramarine**. ... Just placed a link to your **Nigritude Ultramarine** article on my weblog. Cheers! ...  
[www.dashes.com/anil/2004/06/04/nigritude\\_ultra](http://www.dashes.com/anil/2004/06/04/nigritude_ultra) - 101k - Mar 1, 2006 -  
Cached - Similar pages

**Nigritude Ultramarine FAQ**  
Nigritude Ultramarine FAQ - frequently asked questions about **nigritude ultramarine** and the realted SEO contest.  
[www.nigritudeultramarines.com/](http://www.nigritudeultramarines.com/) - 59k - Cached - Similar pages

**SEO contest - Wikipedia, the free encyclopedia**  
The **nigritude ultramarine** competition by SearchGuild is widely acclaimed as ...  
Comparison of search results for **nigritude ultramarine** during and after the ...  
[en.wikipedia.org/wiki/Nigritude\\_ultramarine](http://en.wikipedia.org/wiki/Nigritude_ultramarine) - 37k - Cached - Similar pages

**Slashdot | How To Get Googled, By Hook Or By Crook**  
The current 3rd result showcases the "Nigritude Ultramarine Fighting Force" who ... When discussing **nigritude ultramarine** [slashdot.org] it is important to ...  
[slashdot.org/article.pl?sid=04/05/09/1840217](http://slashdot.org/article.pl?sid=04/05/09/1840217) - 110k - Cached - Similar pages

**The Nigritude Ultramarine Search Engine Optimization Contest**  
It's sweeping the web -- or at least search engine optimizers -- a new contest to rank tops for the term **nigritude ultramarine** on Google.  
[searchenginewatch.com/sereport/article.php/3360231](http://searchenginewatch.com/sereport/article.php/3360231) - 57k - Cached - Similar pages

**Sponsored Links**

**Business Blogging Seminar**  
Coming to L.A. March 16  
Top bloggers reveal key techniques  
[www.blogbusinesssummit.com](http://www.blogbusinesssummit.com)  
Los Angeles, CA

**Full-Time SEO & SEM Jobs**  
Find companies big & small hiring full-time SEO & SEM pros right now  
[CareerBuilder.com](http://CareerBuilder.com)

**SEO Contests**  
Information on SEO Contests like the **Nigritude Ultramarine** contest.  
[www.seo-contests.com/](http://www.seo-contests.com/)

**The SEO Book**  
**Nigritude Ultramarine** & SEO secrets  
Fun, free, raw, & different.  
[www.seobook.com](http://www.seobook.com)

**Ultramarine - Companion**  
Music - Dance - Electronic  
[Overstock.com](http://Overstock.com)

Done

## 2<sup>nd</sup> Generation of Search Ad – Google 2000/2001

The screenshot shows a Google search results page for the query "discount broker". The results are divided into two main sections: "Web" results and "Sponsored Links".

**Web Results:**

- Discount Broker Reviews**: Information on online discount brokers emphasizing rates, charges, and customer comments and complaints.
- Discount Broker Rankings (2008 Broker Survey) at SmartMoney.com**: Discount Brokers, Rank/Brokerage, Minimum to Open Account, Comments, Standard Commission\*, Reduced Commission, Account Fee Per Year (How to Avoid), Avg. ...
- Stock Brokers | Discount Brokers | Online Brokers**: Most Recommended, Top 5 Brokers headlines, 10. Don't Pay Your Broker for Free Funds May 15 at 3:39 PM, 5. Don't Discount the Discounters Apr 18 at 2:41 PM ...
- Discount Broker**: Discount Broker - Definition of Discount Broker on investopedia - A stockbroker who carries out buy and sell orders at a reduced commission compared to ...
- Discount Brokerage and Online Trading for Smart Stock Market ...**: Online stock broker SogoTrade offers the best in discount brokerage investing. Get stock market quotes from this Internet stock trading company.
- 15 questions to ask discount brokers - MSN Money**: Jan 11, 2004 ... If you're not big on hand-holding when it comes to investing, a discount broker can be an economical way to go. Just be sure to ask these ...

**Sponsored Links:**

- Rated #1 Online Broker**: No Minimums, No Inactivity Fees Transfer to FirstTrade for Free! [www.firstrade.com](http://www.firstrade.com)
- Discount Broker**: Commission free trades for 30 days. No maintenance fees. Sign up now. [TDAMERITRADE.com](http://TDAMERITRADE.com)
- TradeKing - Online Broker**: \$4.95 per Trade, Market or Limit. SmartMoney Top Discount Broker 2001 [www.TradeKing.com](http://www.TradeKing.com)
- Scottrade Brokerage**: \$7 Trades, No Share Limit, In Depth Research. Start Trading Online Now! [www.Scottrade.com](http://www.Scottrade.com)
- \$3.95 Online Stock Trades**: Market/Limit Orders, No Share Limit and No Inactivity Fees [www.Marsco.com](http://www.Marsco.com)
- INGDIRECT | ShareBuilder**: [www.sharebuilder.com](http://www.sharebuilder.com)

SogoTrade appears in search results.

SogoTrade appears in ads.

Do search engines rank advertisers higher than non-advertisers?

All major search engines claim no.

## A Bing Search Today –Algorithmic and Sponsored

gaming laptops in india 2020 - B X +

https://www.bing.com/search?q=gaming+laptops+in+india+2020&cvid=3d8c1d2a79a7...

gaming laptops in india 2020

ALL IMAGES VIDEOS MAPS NEWS

2,12,00,000 Results Date Language Region

See gaming laptops in india 2020

Ads

				
Asus Vivobook Gaming 2020... ₹ 82,990.00 Flipkart	Hp Chromebook X360 Core i3 8t... ₹ 51,297.00 Flipkart	Microsoft Surface Laptop 3 Ryzen ... ₹ 1,18,990.00 Flipkart	Lenovo Ideapad Gaming 3 Core i... ₹ 79,998.00 Flipkart	Lenovo Ideapad Gaming 3 Ryzen... ₹ 69,978.00 Flipkart

Related searches

- best gaming laptop india 2020
- best gaming 2020 laptops
- gaming laptop review 2020
- cheap gaming laptop 2020
- best budget gaming laptops 2020
- best affordable gaming laptops 2020
- most powerful gaming laptop 2020
- gaming laptops for sale

Laptops at Amazon - Get Upto 40% off Laptops

[http://www.amazon.in/laptops/best\\_laptop\\_for\\_gaming](http://www.amazon.in/laptops/best_laptop_for_gaming) ▾

(Ad) Best Prices on Laptops from Top Brands. Brand Warranty. Easy Returns. Buy Now!

Buy Gaming Laptop At Tata Cliq - Shop Laptops At Best Prices

<https://www.tatacliq.com> ▾

(Ad) Shop For The Best Laptop From Premium Brands At Affordable Prices On Tata Cliq. Find Your Dream Laptop From The Wide Range Of Pcs Available At Tata Cliq. Shop Now.

Get Up To 66% Off On Laptops - Get Up To 30% Off On Laptops

## A Google Search Today –Algorithmic and Sponsored

Google

gaming laptops india 2020

All News Shopping Images Videos More Settings Tools

About 16,20,00,000 results (0.68 seconds)

Ads · See gaming laptops india 2020



Inspiron 15 7501  
Laptop w/ Intel...  
₹91,989.98  
Dell India  
★★★★★ (107)  
Installed  
8 GB RAM



Inspiron 14 5490  
Laptop w/ Intel...  
₹61,490  
Dell India  
★★★★★ (286)  
Installed  
8 GB RAM



Inspiron 15 5590  
Laptop w/ Intel...  
₹62,489.96  
Dell India  
★★★★★ (434)  
Installed  
8 GB RAM



Latitude 5500  
Laptop w/ Intel...  
₹87,322.61  
Dell India  
★★★★★ (8)  
Installed  
8 GB RAM



Vostro 5401  
Laptop w/ Intel...  
₹59,990  
Dell India  
★★★★★ (6)  
Installed  
8 GB RAM

»

Ad · in.msi.com/ ▾

Laptop for AAA Games - Diwali Sale Dhamaka

Up to 35% Discount! Diwali Sale Dhamaka. Ignite Power within the Best Offer of the Year. Latest 10th Gen. Intel Core i7 processor powers up with up to 15% performance. Register A Product. Download Our Mobile App. View Brochure.

www.digit.in › Top Products ▾

Best Gaming Laptops to Buy in India with Price and Specs (2 ...

Best Gaming Laptops in India. HINDI. By Mithun Mohandas | Price Updated on 01-Oct-2020.

Gaming laptops, there's isn't a better platform ...

vsbytes.com › Gaming & PC Builds ▾



**THANK YOU**

---

**Bhaskarjyoti Das**

Department of Computer Science Engineering



# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

## Web Search Basics

---

**Bhaskarjyoti Das**

Department of Computer Science Engineering

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

**Advertising as an Economic Model  
For Search Engines – Ranking the  
Search Ads**

**Bhaskarjyoti Das**  
Department of Computer Science Engineering

## Recap From Previous Class

---

- Advertisement Business Models
  - CPM : cost per thousand impression - derived from Display Ad or Banner Ad
  - CPC : cost per click. PPC is the approach and CPC is the measure
  - CPA : cost per action
  - For the same spend on thousand impression, CPM is least risky for publisher and most risky for advertiser. Similar things can be argued for CPA
- Search Ad : Publisher, Advertiser, Buyer – WIN-WIN-WIN model theoretically
- Algorithmic Search vs. Sponsored Search – we see both
- Sponsored Search : advertiser bids for keywords and this brings the revenue for search engines

## What we have : Sponsored Search and Algorithmic Search

nigritude ultramarine - Google Search - Mozilla Firefox

File Edit View Go Bookmarks Yahoo! Tools Help

Getting Started Latest Headlines

Y! Search Web Mail My Yahoo! Games Movies Music Answers Personals Sign In

pragh60@gmail.com | My Account | Sign out

Google Web Images Groups News Froogle Local more »

nigritude ultramarine Search Advanced Search Preferences

**Web**

Results 1 - 10 of about 185,000 for nigritude ultramarine. (0.35 seconds)

**Anil Dash: Nigritude Ultramarine**  
Do me a favor: Link to this post with the phrase **Nigritude Ultramarine** ... Just placed a link to your **Nigritude Ultramarine** article on my weblog. Cheers! ...  
[www.dashes.com/anil/2004/06/04/nigritude\\_ultra](http://www.dashes.com/anil/2004/06/04/nigritude_ultra) - 101k - Mar 1, 2006 - Cached - Similar pages

**Nigritude Ultramarine FAQ**  
Nigritude Ultramarine FAQ - frequently asked questions about nigritude ultramarine and the realted SEO contest.  
[www.nigritudeultramaries.com/](http://www.nigritudeultramaries.com/) - 59k - Cached - Similar pages

**SEO contest - Wikipedia, the free encyclopedia**  
The **nigritude ultramarine** competition by SearchGuld is widely acclaimed as ... Comparison of search results for **nigritude ultramarine** during and after the ...  
[en.wikipedia.org/wiki/Nigritude\\_ultramarine](http://en.wikipedia.org/wiki/Nigritude_ultramarine) - 37k - Cached - Similar pages

**Slashdot | How To Get Googled, By Hook Or By Crook**  
The current 3rd result showcases the "Nigritude Ultramarine Fighting Force" who ... When discussing **nigritude ultramarine** [slashdot.org] it is important to ...  
[slashdot.org/article.pl?sid=04/05/09/1840217 - 110k](http://slashdot.org/article.pl?sid=04/05/09/1840217 - 110k) - Cached - Similar pages

**The Nigritude Ultramarine Search Engine Optimization Contest**  
It's sweeping the web -- or at least search engine optimizers -- a new contest to rank tops for the term **nigritude ultramarine** on Google.  
[searchenginewatch.com/sereport/article.php/3360231](http://searchenginewatch.com/sereport/article.php/3360231) - 57k - Cached - Similar pages

Done

b gaming laptops in india 2020 - Bing

https://www.bing.com/search?q=gaming+laptops+in+india+2020&cvid=3d8c1d2a79a7...

gaming laptops in india 2020

ALL IMAGES VIDEOS MAPS NEWS

2,12,00,000 Results Date Language Region

See gaming laptops in india 2020 Ads

Related searches

- best gaming laptop india 2020
- best gaming 2020 laptops
- gaming laptop review 2020
- cheap gaming laptop 2020
- best budget gaming laptops 2020
- best affordable gaming laptops 2020
- most powerful gaming laptop 2020
- gaming laptops for sale

Image	Product Name	Price	Offer	Source
	Asus Vivobook Gaming 2020...	₹ 82,990.00	Flipkart	
	Hp Chromebook X360 Core i3 8th...	₹ 51,297.00	Flipkart	
	Microsoft Surface Laptop 3 Ryzen ...	₹ 1,18,990.00	Flipkart	
	Lenovo Ideapad Gaming 3 Core I...	₹ 79,998.00	Flipkart	
	Lenovo Ideapad Gaming 3 Ryzen...	₹ 69,978.00	Flipkart	

Laptops at Amazon - Get Upto 40% off Laptops  
<http://www.amazon.in/laptops/best-laptop-for-gaming>

Buy Gaming Laptop At Tata Cliq - Shop Laptops At Best Prices  
<https://www.tatacliq.com>

Get Up To 66% Off On Laptops - Get Up To 30% Off On Laptops

## How Google Indicates Sponsored Search ( Ads) ?

2007

**Go Compare Car Insurance - GoCompare.com**  
[www.gocompare.com/Car-Insurance](http://www.gocompare.com/Car-Insurance)  
99% Could Save with Go Compare! 50% + Could Save up to \$293  
Gocompare.com has 111 followers on Google+  
Compare 1000+ Car Insurance Prices - From Over 120 Insurance Providers

2010

**Go Compare Car Insurance - GoCompare.com**  
Compare 1000+ Car Insurance Prices - From Over 120 Insurance Providers  
[www.gocompare.com/Car-Insurance](http://www.gocompare.com/Car-Insurance)

2011

**Go Compare Car Insurance - GoCompare.com**  
[www.gocompare.com/Car-Insurance](http://www.gocompare.com/Car-Insurance)  
Compare 1000+ Car Insurance Prices - From Over 120 Insurance Providers

2013  
(early)

**Go Compare Car Insurance - GoCompare.com**  
[www.gocompare.com/Car-Insurance](http://www.gocompare.com/Car-Insurance)  
Compare 1000+ Car Insurance Prices - From Over 120 Insurance Providers

2013  
(late)

**Go Compare Car Insurance - GoCompare.com**  
Ad [www.gocompare.com/Car-Insurance](http://www.gocompare.com/Car-Insurance)  
Compare 1000+ Car Insurance Prices - From Over 120 Insurance Providers

2014

**Go Compare Car Insurance - GoCompare.com**  
Ad [www.gocompare.com/Car-Insurance](http://www.gocompare.com/Car-Insurance)  
Compare 1000+ Car Insurance Prices - From Over 120 Insurance Providers

2017  
(early)

**Go Compare Car Insurance - GoCompare.com**  
Ad [www.gocompare.com/Car-Insurance](http://www.gocompare.com/Car-Insurance)  
Compare 1000+ Car Insurance Prices - From Over 120 Insurance Providers

2017  
(late)

**Go Compare Car Insurance - GoCompare.com**  
Ad [www.gocompare.com/Car-Insurance](http://www.gocompare.com/Car-Insurance)  
Compare 1000+ Car Insurance Prices - From Over 120 Insurance Providers

## How are the Ads on the Right Ranked ?

Web Images Maps News Shopping Gmail more

Sign in

Google discount broker

Search Advanced Search Preferences

Web Results 1 - 10 of about 807,000 for discount broker [definition]. (0.12 seconds)

**Discount Broker Reviews**  
Information on online discount brokers emphasizing rates, charges, and customer comments and complaints.  
[www.broker-reviews.us/](http://www.broker-reviews.us/) - 94k - Cached - Similar pages

**Discount Broker Rankings (2008 Broker Survey) at SmartMoney.com**  
Discount Brokers. Rank/ Brokerage/ Minimum to Open Account, Comments, Standard Commission\*, Reduced Commission, Account Fee Per Year (How to Avoid), Avg. ...  
[www.smartmoney.com/brokers/index.cfm?story=2004-discount-table](http://www.smartmoney.com/brokers/index.cfm?story=2004-discount-table) - 121k - Cached - Similar pages

**Stock Brokers | Discount Brokers | Online Brokers**  
Most Recommended. Top 5 Brokers headlines. 10. Don't Pay Your Broker for Free Funds May 15 at 3:39 PM. 5. Don't Discount the Discounters Apr 18 at 2:41 PM ...  
[www.fool.com/investing/brokers/index.aspx](http://www.fool.com/investing/brokers/index.aspx) - 44k - Cached - Similar pages

**Discount Broker**  
Discount Broker - Definition of Discount Broker on Investopedia - A stockbroker who carries out buy and sell orders at a reduced commission compared to a ...  
[www.investopedia.com/terms/d/discountbroker.asp](http://www.investopedia.com/terms/d/discountbroker.asp) - 31k - Cached - Similar pages

**Discount Brokerage and Online Trading for Smart Stock Market ...**  
Online stock broker SogoTrade offers the best in **discount brokerage** investing. Get stock market quotes from this internet stock trading company.  
[www.sogotrade.com/](http://www.sogotrade.com/) - 39k - Cached - Similar pages

**15 questions to ask discount brokers - MSN Money**  
Jan 11, 2004 ... If you're not big on hand-holding when it comes to investing, a **discount broker** can be an economical way to go. Just be sure to ask these ...  
[moneycentral.msn.com/content/Investing/StartInvesting/P66171.asp](http://moneycentral.msn.com/content/Investing/StartInvesting/P66171.asp) - 34k - Cached - Similar pages

**Sponsored Links**

**Rated #1 Online Broker**  
No Minimums. No Inactivity Fee Transfer to Firstrade for Free!  
[www.firstrade.com](http://www.firstrade.com)

**Discount Broker**  
Commission free trades for 30 days. No maintenance fees. Sign up now.  
[TDAMERITRADE.com](http://TDAMERITRADE.com)

**TradeKing - Online Broker**  
\$4.95 per Trade, Market or Limit SmartMoney Top Discount Broker 2007  
[www.TradeKing.com](http://www.TradeKing.com)

**Scottrade Brokerage**  
\$7 Trades, No Share Limit. In-Depth Research. Start Trading Online Now!  
[www.Scottrade.com](http://www.Scottrade.com)

**Stock trades \$1.50 - \$3**  
100 free trades, up to \$100 back for transfer costs, \$500 minimum.  
[www.sogotrade.com](http://www.sogotrade.com)

**\$3.95 Online Stock Trades**  
Market/Limit Orders, No Share Limit and No Inactivity Fees  
[www.Marsco.com](http://www.Marsco.com)

**INDIRECT | ShareBuilder**  
Brokerage Offers Many Advantages

## How are the Ads Ranked (1) ?

---

- Advertisers bid for keywords – sale by auction.
- Open auction system: Anybody can participate and bid on keywords.
- How does the auction determine an ad's rank and the price paid for the ad?
  - Basis is a second price auction, but with twists
- Advertisers are only charged when somebody clicks on your ad.
  - For the bottom line, this is perhaps the most important research area for search engines – computational advertising.
  - Squeezing an additional fraction of a cent from each ad means billions of additional revenue for the search engine

## How are the Ads Ranked (2) ?

---

- First cut: according to bid price `a la Goto
  - Bad idea: open to abuse
  - Example: query [does my husband cheat?] → ad for divorce lawyer
  - We don't want to show nonrelevant ads.
  
- Instead: rank based on **bid price and relevance**

## How are the Ads Ranked (3) ?

---

- **Second cut** - bid price and relevance

- **Result:** A non-relevant ad will be ranked low.

- Even if this decreases search engine revenue short-term

- **Hope:** Overall acceptance of the system and overall revenue is maximized if users get useful information.

- **The main ranking factor:** the query

- **Key measure of ad relevance:** ClickThrough Rate

- Click Through Rate (CTR) = clicks per impressions

- **Other ranking factors:** location, time of day, landing page quality and loading speed of landing page

- A minimal model will combine **CTR with Bid Price in an Auction Market Place**

## How Search Ads Work ?

---

- Google uses a modified **second-price auction system** to **rank the ads** that appear on Google search-engine results pages (SERPs), and **determine the cost** advertisers have to pay to appear on top of the results page.
- Google's **second-price auction mechanics** basically mean that advertisers pay the amount needed to beat the nearest competitor, depending on their Ad Rank

## Page Rank vs. Ad Rank ?

---

- **Page Rank** decides the ranking in algorithmic search results
- **Ad Rank** decides the ranking in sponsored search results
- **Ad Rank** is the PageRank equivalent for search ads in Google.
  - It is an algorithm used to determine how highly ads rank in the sponsored results and how much the advertiser has to pay when an internet user clicks the ad.
- The Ad Rank formula may consist of:
  - **Expected Click Through Rate**
  - **Landing Page Experience**
  - **Ad Relevance ( to the query)**
  - **Ad Formats ( price, direction, phone no etc. )**

## Auction vs. Buy/Sell

---

- If we want to buy a television, we go to showroom, ask the shopkeeper, look at the product and then pick that up if we like ( after some bargaining)
- **We make a bid** i.e. the price a buyer is willing to pay is called a **bid**
- **In Auction**, the **seller** may not know the **private valuation of the** buyer and **may sell to the highest bidder**
- Since many markets are imperfect and **it is hard to discover the potential buyers' true valuation of the item, auction is used to discover this.**
- To summarize
  - Auction is a **NOT bargaining or negotiation**
  - Auction is a **price discovery mechanism**

## Auctions in Search

Web Images Videos Maps News Shopping Gmail more ▾

blumrosen@gmail.com



Web [Show options...](#)

Results 1 - 10 of about 154,000,000 for [ticket new york](#) (0.22 seconds)

[Orbitz Travel: Airline Tickets, Cheap Hotels, Car Rentals ...](#)  
Book cheap airline [tickets](#), hotel reservations, car rentals, vacations and ... Nevada, New Hampshire, New Jersey, New Mexico, [New York](#), North Carolina ...  
[+ Show stock quote for OWW](#)  
[Hotels - Cheap Flights\\_ Cheap Tickets](#) - Car rental - Travel deals  
[www.orbitz.com/](#) - Cached - Similar -

[Cheap Flights - Compare prices on cheap flights, airlines & last ...](#)  
from £27.60 per person Transfers to [New York](#) with [HolidayTaxis.com](#) ... Cheap flights (UK & Ireland); [Airline Tickets \(USA\)](#); Cheap flights (Canada) ...  
[USA - Search by date](#) - [Australia](#) - [New York](#)  
[www.cheapflights.co.uk/](#) - Cached - Similar -

[Discount Broadway Tickets and Free TV Show Tickets in New York City](#)  
Discount [Broadway Tickets](#), Free TV Show [Tickets](#), Discount Parking and Free Coupons in [New York City](#). The complete guide to discount to Broadway discounts ...  
[Discount tickets](#) - [Discount Broadway Tickets](#) - [Broadway](#)  
[www.nytix.com/](#) - Cached - Similar -

[New York Event Ticket Broker, New York Concert Tickets, Sports ...](#)  
GOTickets is your source for [New York Event Tickets](#), so BUY with confidence and KNOW your order is secure and guaranteed.  
[www.gotickets.com/broker/ny/new\\_york.php](#) - Cached - Similar -

[CityPass® save up to 50%: New York, Chicago, Atlanta, Southern ...](#)  
CityPass is available for [New York](#), Chicago, Atlanta, Southern California, ... You'll skip most [ticket lines](#), too. CityPass is the ideal bargain for both ...

Sponsored Links

[New York Airfare Bargains](#)  
Save on [New York Ticket](#)  
Search 100s of Sites for Deals  
[www.newyork.Kayak.com](#)

[Fly to New York at \\$1099](#)  
Don't Miss Our Limited Winter Offer  
& Book Your Flight to [New York](#) Now!  
[ELAL.co.il/?New-York-Flights](#)

[See your ad here »](#)

An online auction is run for every individual search.

Real ("organic") search result

Ads: "sponsored search"

## Vickery Auction or 2<sup>nd</sup> Price Sealed Bid Auction

---

- Also called **Vickery Auction**
- The bidders **submit simultaneous sealed bids to seller** . All bids are **opened at once and highest bidder wins**.
- He **pays price of second highest bid**
- Second price variety
  - Bids are **private information**
  - Bids are **made simultaneously**
  - **Highest bidder wins**
  - Winner pays **second highest bid**

## Google's 2<sup>nd</sup> Price Auction

- Each bidder writes his bid in a sealed envelope.
- The seller:
  - Collects bids
  - Open envelopes.
- Winner:  
bidder with the highest bid.



Payment:

winner pays the **2<sup>nd</sup> highest bid**.



## Google's 2<sup>nd</sup> Price Auction

advertiser	bid	CTR	ad rank	rank	paid
A	\$4.00	0.01	0.04	4	(minimum)
B	\$3.00	0.03	0.09	2	\$2.68
C	\$2.00	0.06	0.12	1	\$1.51
D	\$1.00	0.08	0.08	3	\$0.51

- **Bid:** maximum bid for a click by advertiser
- **CTR(Click-Through Rate) is a measure of relevance :** when an ad is displayed, what percentage of time do users click on it?
- **Ad rank: bid × CTR:** this trades off (i) how much money the advertiser is willing to pay **against** (ii) how relevant the ad is
- **Rank:** rank in auction derived from ad-rank
- **Paid:** second price paid by advertiser

## Google's 2<sup>nd</sup> Price Auction

---

advertiser	bid	CTR	ad rank	rank	paid
A	\$4.00	0.01	0.04	4	(minimum)
B	\$3.00	0.03	0.09	2	\$2.68
C	\$2.00	0.06	0.12	1	\$1.51
D	\$1.00	0.08	0.08	3	\$0.51

**Second price auction:** The advertiser pays the minimum amount necessary to maintain their position in the auction ( next lower + 1 cent)

$$\text{price}_1 \times \text{CTR}_1 = \text{bid}_2 \times \text{CTR}_2 \text{ (this will result in rank}_1 = \text{rank}_2\text{)}$$

$$\text{price}_1 = \text{bid}_2 \times \text{CTR}_2 / \text{CTR}_1$$

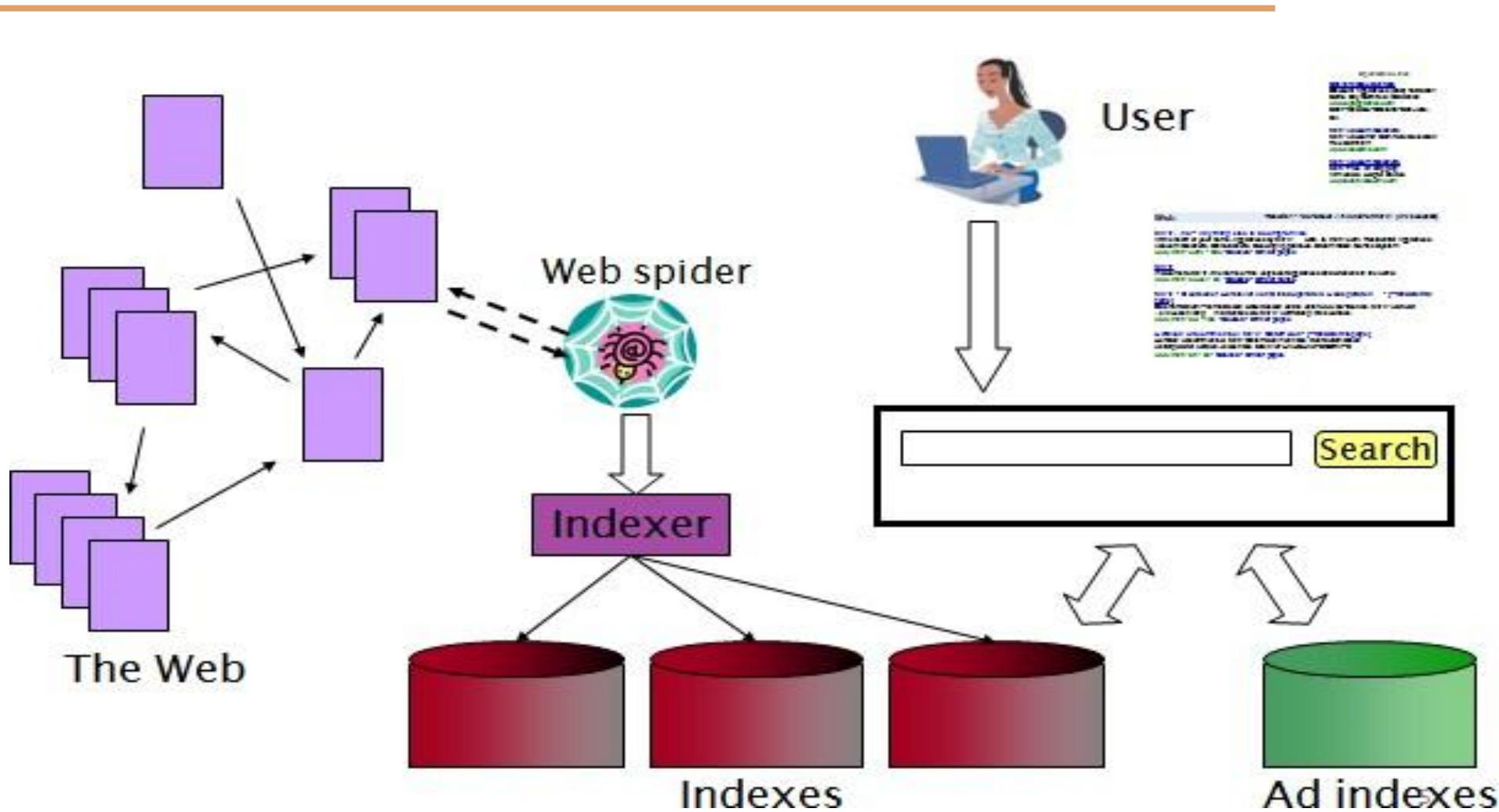
$$\begin{aligned}
 p_1 &= \text{bid}_2 \times \text{CTR}_2 / \text{CTR}_1 = 3.00 \times 0.03 / 0.06 = \\
 1.50 & \quad p_2 = \text{bid}_3 \times \text{CTR}_3 / \text{CTR}_2 = 1.00 \times 0.08 / 0.03 \\
 &= 2.67 \quad p_3 = \text{bid}_4 \times \text{CTR}_4 / \text{CTR}_3 = 4.00 \times
 \end{aligned}$$

## What is the Final Criteria for Ad Ranking ?

---

- In fact there can be **multiple criteria** – exact algorithm is not known ( well kept secret)
- **Ad Rank** ( derived from **Bid Price** and **Clickthrough Rate** )
- **Advertiser Quality**
- **Similarity of the Ad with the Query etc.**

## Now We Need an Index For Advertisement



## Brief (non technical) history

---

- Early keyword-based engines
  - 1994 -
    - Infoseek : Webmasters could submit a page in real time
    - Yahoo! Search : by David Filo and Jerry Yang, beginning as a collection of favorable web pages that included a man-made description with each URL
    - Lycos : By August 1994 they had identified 394,000 documents; 1.5 million by January 1995
  - 1995 -
    - Excite : created by six Stanford undergrads
    - Altavista : First to allow natural language queries
    - Inktomi - Inktomi pioneered paid inclusion model

## Brief( non-technical) history

---

- 1998 -
  - Overture - Formerly Goto.com, it was the first company to successfully provide a pay-per-click placement search service
  - MSN search - Relied on Overture, Looksmart, and Inktomi until Google proved their backlinks model
  - Google Launches – Link-based ranking pioneered by Google. Blew away all early engines save Inktomi.
    - Paid Search: Meanwhile Goto/Overture's annual revenues were nearing \$1 billion. Google added paid search “ads” to the side, independent of search results
    - Yahoo! followed suit, acquiring Overture (for paid placement) and Inktomi (for search)
- 2005+: Google gains search share, dominating in Europe and North America
- 2009: Yahoo! and Microsoft got into a deal - rebranding of MSN/Live Search essentially using Yahoo search as Bing



**THANK YOU**

---

**Bhaskarjyoti Das**

Department of Computer Science Engineering



# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

## Web Search Basics

---

**Bhaskarjyoti Das**

Department of Computer Science Engineering

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

**Advertisement Technology  
Infrastructure**

**Bhaskarjyoti Das**  
Department of Computer Science Engineering

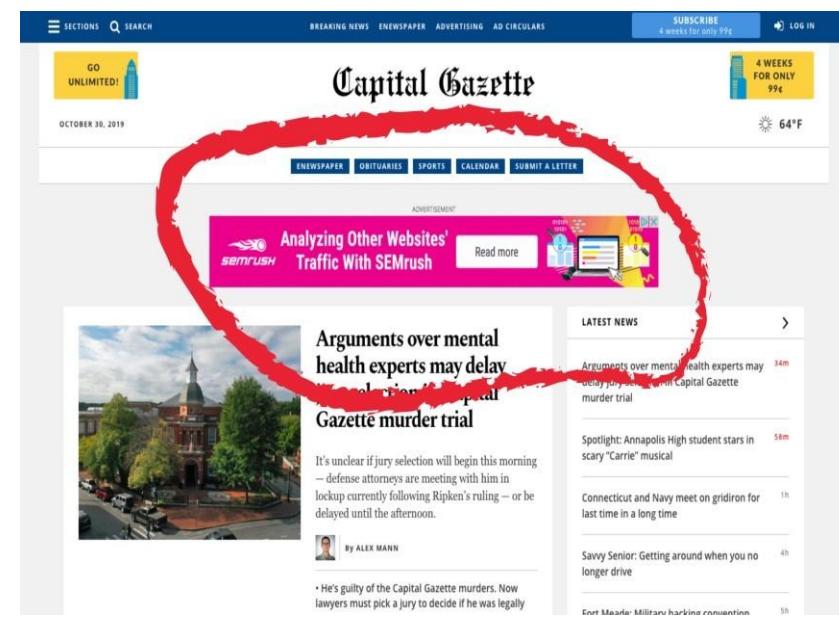
## Google Adword (2000)

---

- **Keyword based**
- Businesses that use AdWords will pay Google a sum based on their campaign budget for advertisement based on keyword search :
  - Be seen by customers at the very moment they're searching on Google for the things you offer.
- With AdWords, businesses are paying Google to place their ads on Google's search and display network.

## Google Display Networks (GDN)

- Not related to Search
- The GDN is Google's network of sites and apps. **Google has agreements with millions of websites and applications**, wherein the site receives revenue for allowing Google to advertise there.
- The GDN is where your **display ads** appear to users throughout the internet.



## Google Adsense (2003)

---

- **Adsense : for Website Publishers**

- AdSense allows publishers to monetize their websites by displaying relevant Google AdWords ads. Publishers get paid when visitors click on the ads.

- Placing Adword Ads **but not on Google Display Network**

- Google **uses its technology to serve advertisements based on website content, the user's geographical location, and other factors.**

- With AdSense, **website owners are getting paid by Google** to offer up precious website real estate for AdWords placements. **Google is an intermediary providing Ad Technology**

## Google AdWords Vs. AdSense

---

1. **AdSense** Is for Website Publishers, **AdWords** Is for Businesses
2. Google AdSense pays Website Owners, Businesses pay **Google AdWords**
3. Example : **Adwords For Video**
  - AdWords for video enables you to display video ads
    - in the YouTube search results
    - Before, during, and after videos on YouTube
    - And the Google Display Network.
  - Unlike a traditional AdWords campaign, with AdWords for video, you can use demographic targeting to more efficiently reach your audience. The targeting options are (age, gender, and interests)

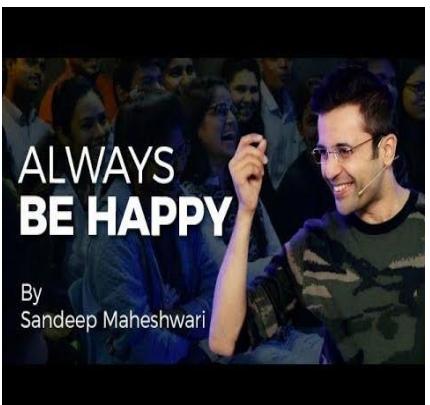
## AdWords at work : few YouTube Millionaires in India @ 2020



Amit Bhadana–  
Channel full of fun  
and 15 M subscriber  
at no 1 position



Technical Guruji –  
essentially  
product reviews  
at no 3 position



Sandeep  
Maheshwari – Free  
life changing  
sessions and  
seminars



*Harpal Singh Sokhi*

Harpal Singh Sokhi– Of Namak Shamak fame and now offers franchise for chain of restaurants “Twist Of Tadka”. Food is about love, positive mind, and affection while cooking ;-

## Google AdX Vs. AdSense

---

### 1. AdSense

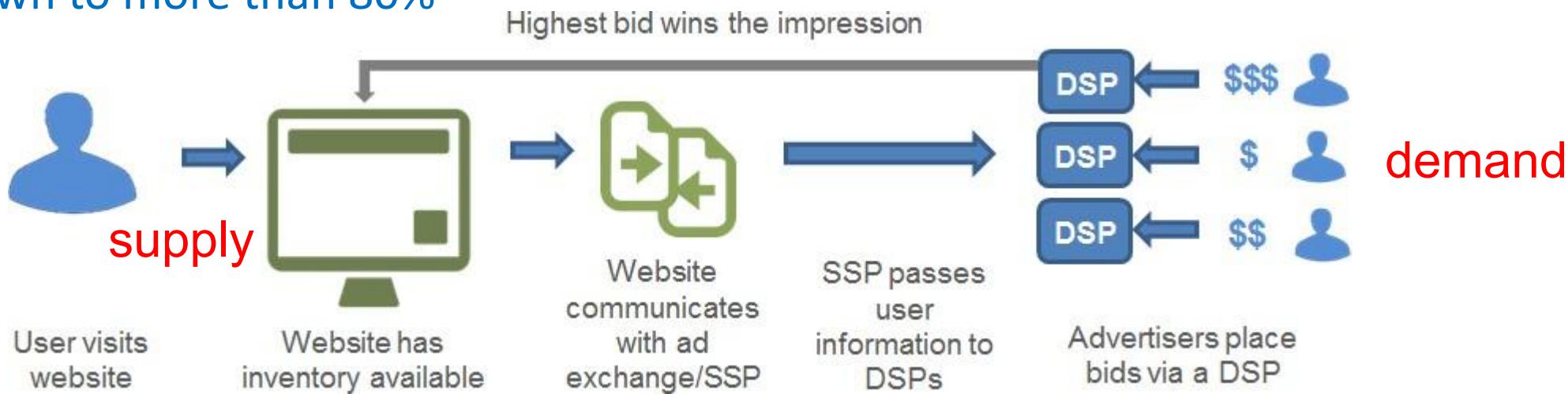
- It allows publishers to monetise their content on desktop and mobile like a living wage ( limited monetizability). The platform is geared towards providing low-cost impressions to advertisers.
- Publishers can't set a floor price on their ad units, and they can't also sell those ad units on their own. So they're stuck with what Google offers them.

### 2. AdX

- For large publishers and large spenders of ad budgets
- This means it is a programmatic advertising platform, offering real-time bidding (RTB) on ad spaces to ad networks.

## Programmatic Advertisement – Rise of Intermediaries

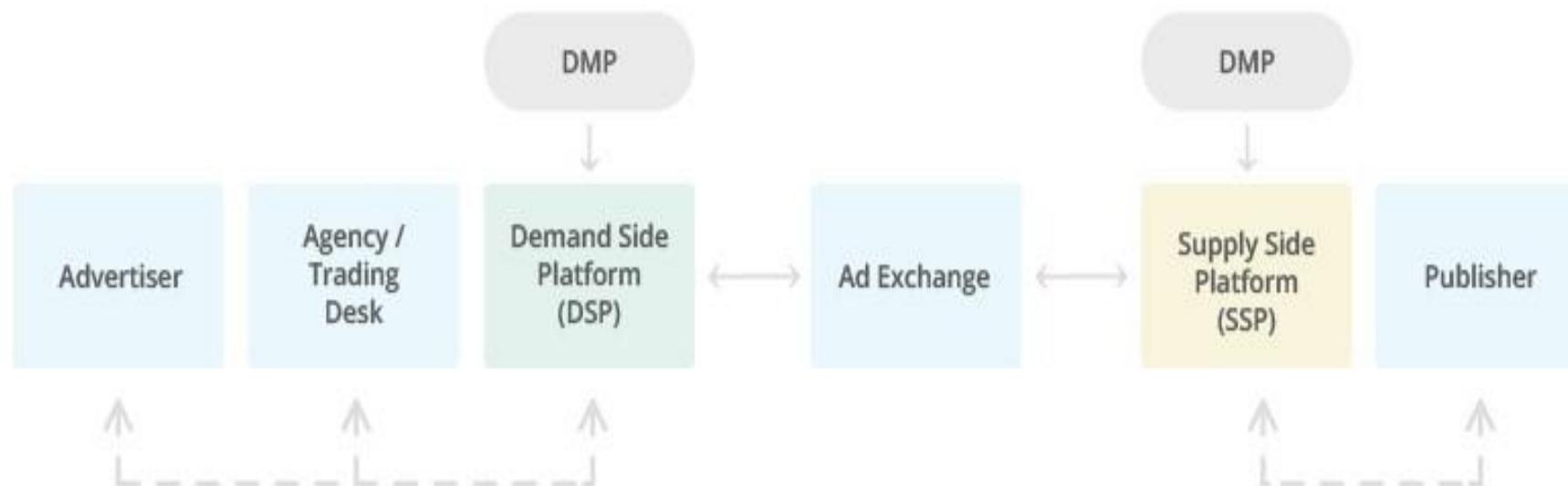
**Programmatic Advertisement** is the process of buying and selling ads with software and publishing those ads contextually based on complex algorithms that take care of many things like user location, browsing history, IP address, time of day. **As of 2019, its US ad market share has grown to more than 80%**



**Inventory of Ad Placement slots from publishers**

**Inventory of mobile, search and video ads**

## Bringing in Ad Exchange for a Complete Picture

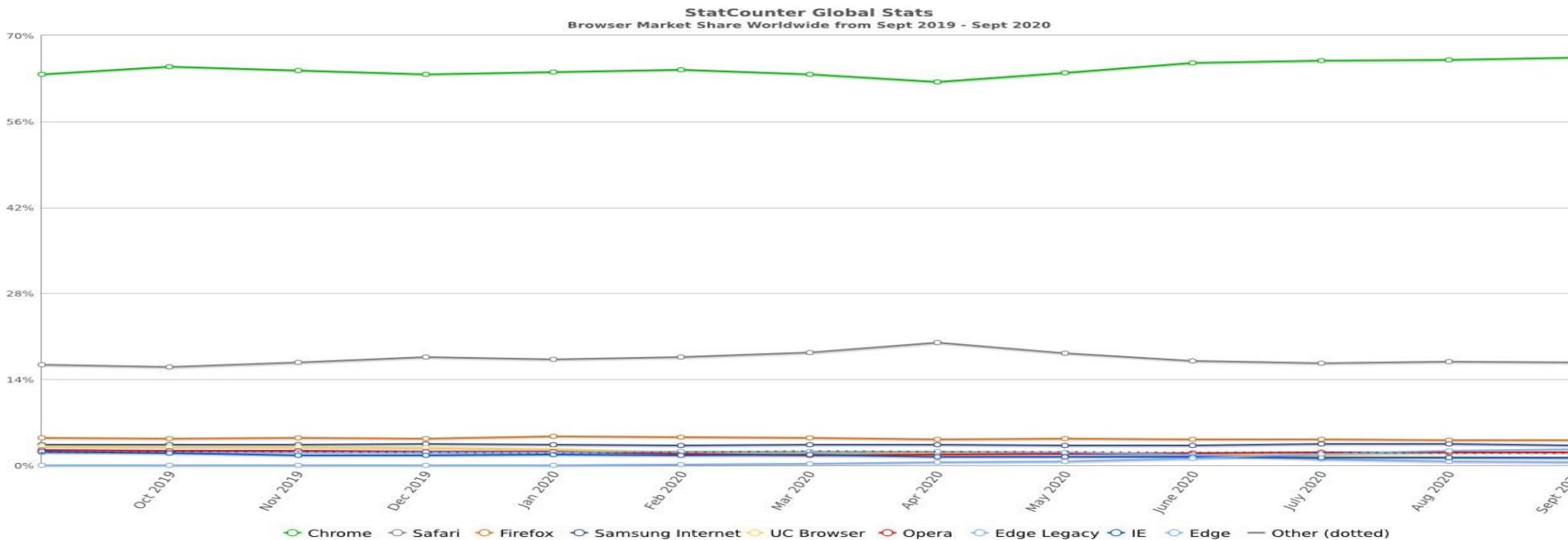


## Programmatic Advertisement

---

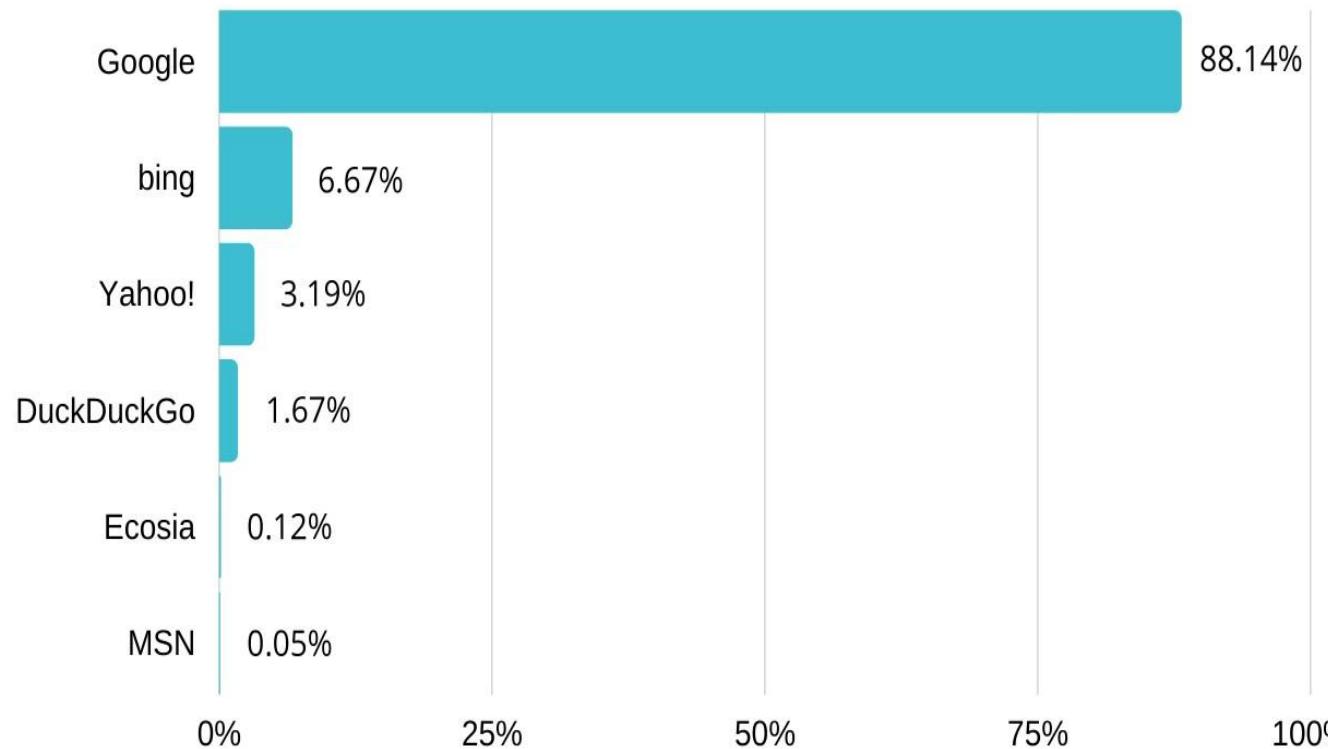
1. **Demand Side Platforms** are third-party software that allow you to purchase analyze, manage ads across *many networks* from a single place (**not only** FaceBook Ad Manager for Facebook and Instagram, Google Display network Manager for Google Display Network ).
2. **Supply Side Platforms** are third-party software that enable publishers to auction off their inventory, fill it with the winning buyer's creative, and earn revenue. They communicate with DSPs about the details of an impression.
3. **Data management Platform** are third-party software used to compile, store, and analyze data. DSPs and SSPs then use that information to optimize. In this way, a DSP, SSP, and DMP can all work together along the **programmatic supply chain** to serve an ad.

## Browser Market Share : One Does Search on Browsers



## Search Engine Market Share

US Search Engine Market Share in 2020



Source: Gs.statcounter.com

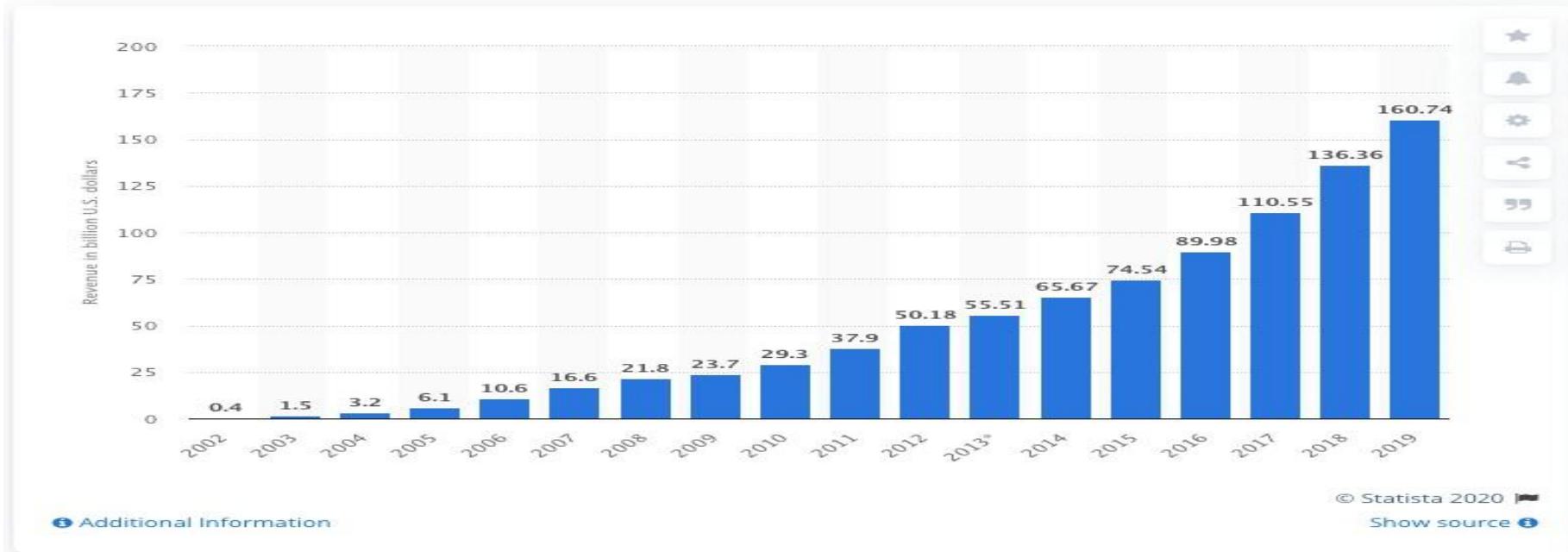
## Search Business in Not Only Search

Google's no 1 in Search Business is related to in many things

- **Understanding of the users** (who search and gets targeted by Advertisements)
  - **No 1 in Browser Market Share**
- **Android** : reach across all **mobile channels** ( mobile foot print is growing faster than PC for past many years)
- **Biggest publisher in video** : YouTube
- **Search Engineering** : preferred partners for all publishers and advertisers for it's index of the internet
- **Advertisement or Audience Engineering** ( very advanced in ad placement technology – adword, adsense, adX)
- **Cloud Platform Technology** : Google **scales it better than anyone else** on **cloud** (;-

## Why Advertisement ( Search ) is Big ?

Annual revenue of Google from 2002 to 2019  
(in billion U.S. dollars)



- Google's 2019 Revenue : 160 B
- Google Ad Revenue in 2019 : 134.81 B
- Google Net Income in 2019 : 34.3 B
- Google's R&D Budget in 2019 : 26.018 B
- Infosys revenue in 2019 : 11.79 B, net income : 2.1 B



**THANK YOU**

---

**Bhaskarjyoti Das**

Department of Computer Science Engineering



# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

## Web Search Basics

---

**Bhaskarjyoti Das**

Department of Computer Science Engineering

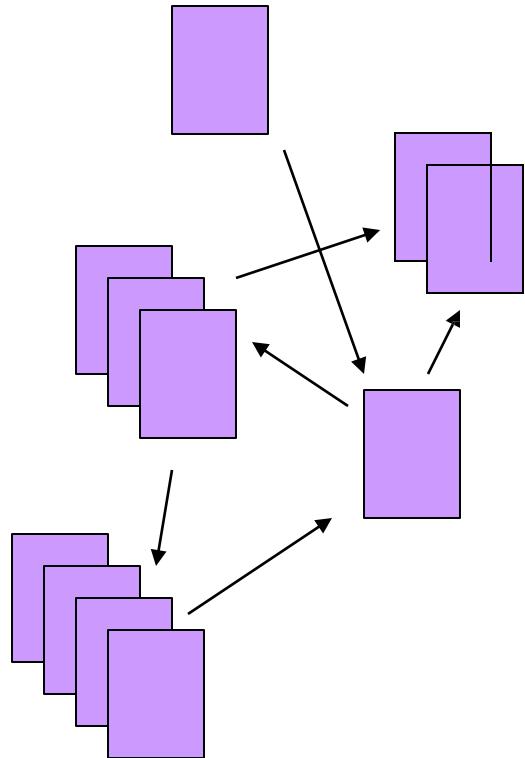
# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

## Web Characteristics

**Bhaskarjyoti Das**

Department of Computer Science Engineering



- No design/co-ordination/rules
- Distributed content creation, linking,  
democratization of publishing
- Content includes truth, lies, obsolete information, contradictions ... new level of granularity
- No universal notion of trust – trusted website for two users can be axially different
- Unstructured(text, html, ...),  
semi-structured (XML, annotated photos),  
structured (Databases)...

## The Web document collection

- Scale much larger than previous text collections ...
- Growth – slowed down from initial “volume doubling every few months” but still expanding
- Content can be *dynamically generated*
- Method of ranking documents by TF /IDF based similarity in a static collection does not work any more ...



## Web Graph

---

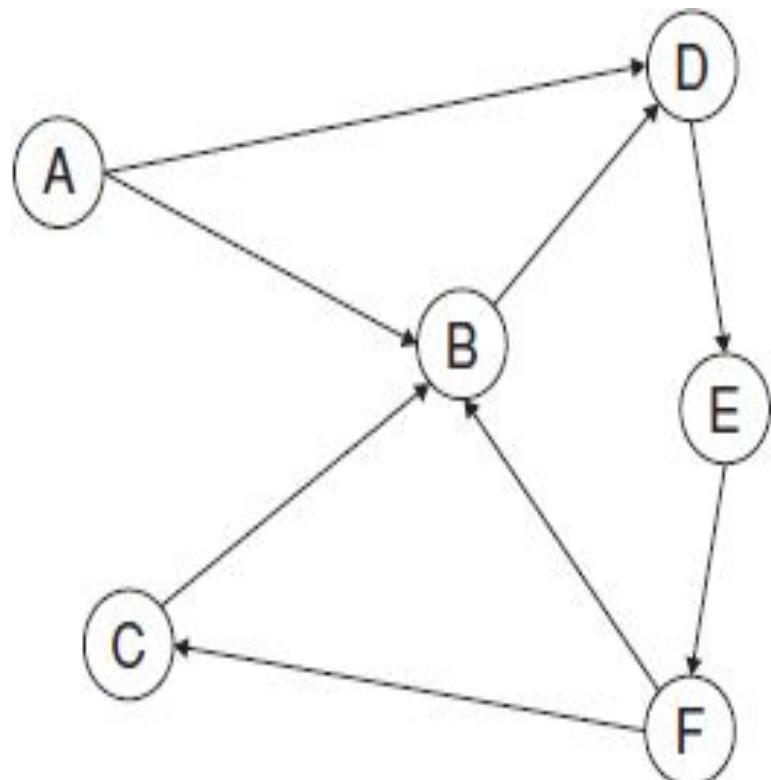
- **Static Web** consists of static HTML pages together with **hyper links** between them as a **directed graph** in which web page is a **node** and hyperlink a **directed edge**



Figure 19.2 Two nodes of the web graph joined by a link.

## In Degree and Out Degree

---



**In-degree (node)** : no of edges coming in

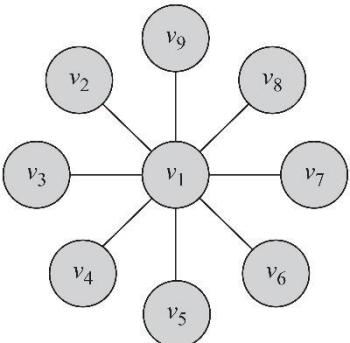
**Out-degree(node)** : no of edges going out

**Figure 19.3** A sample small web graph. In this example we have six pages labeled A through F. Page B has in-degree 3 and out-degree 1. This example graph is not strongly connected: There is no path from any of pages B through F to page A.

## Concept of Degree Centrality in a Graph

---

- The degree centrality measure ranks nodes/pages with more connections higher in terms of centrality
- You **ignore both the direction and the value of the tie**



- $d_i$  is the degree (number of adjacent edges) for vertex  $v_i$

## Issue with Degree Centrality

---

- It is all **about local neighborhood !!**
  - A node with a high degree centrality may be **capable of affecting a lot of neighbors in its neighborhood** at once, but **we cannot say anything about the opportunities for global outreach.**
- Does **not work with directed graph !**
  - Calculated on matrices holding **binary symmetric data**. So before calculating degree centrality, ensure that edge data are both binary and symmetrical

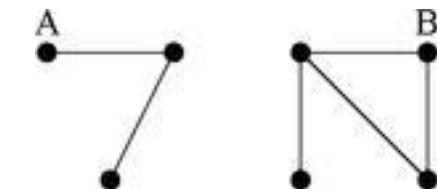
## Think About Eigen Vectors...

---

- Consider two actors, A and B and think of closeness.
  - Actor **A is quite close to a small and fairly closed group** within a larger network, and **rather distant from many of the members** of the population.
  - Actor B is at a **moderate distance from all of the members** of the population. In a sense, however, **actor B is really more "central" than actor A** in this example
  - The **far-ness measures for actor A and actor B could be quite similar to magnitude ( due to average etc.).!**
- The eigenvector approach is an effort to **find the most central actors in terms of the "global" or "overall" structure of the network, and to pay less attention to patterns that are more "local".**
- It is **a good idea to examine both, and to compare them.**

## Disconnected Undirected Network and components

- A network for which there exists pairs of vertices that there is no path between them is called disconnected
- If there exists a path between any possible pair of vertices in a network the latter is called connected
- Component is a *maximal* subset of vertices of a network such that there exists at least one path from every vertex of the subgroup to any other vertex
- Components of a graph are sub-graphs that are connected within, but disconnected between themselves.
- If a graph contains one or more "isolates," these actors are components.



## Directed Graph – Weak and Strong Components

---

- For **directed graphs** (in contrast to simple graphs), we can define two different kinds of components.
  - A **weak component** is a set of nodes that are connected, regardless of the direction of ties.
  - A **strong component** requires that there be a directed path from any node A to any other node B in order for the two to be in the same component
  - Strong components are rare ( no one way friendship)

## Directed Graph – IN and OUT Components

---

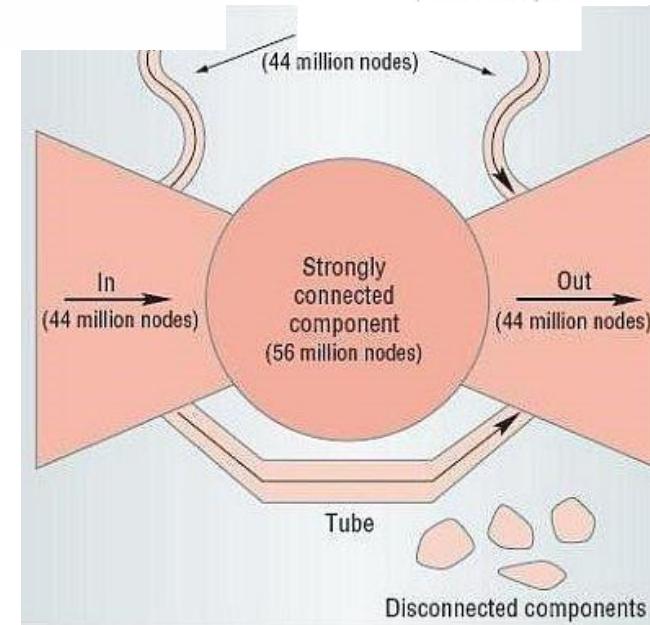
### Out Components :

- The **set of vertices** that are **reachable via directed paths** from node A (including A), form its **out-component**
- Out-component is a **property of both the network and the starting vertex**
- Hence, a **vertex can belong to more than one out-components**

### In Components :

- The **in-component** of a specific **vertex A** is the **set of all vertices** (including A) from which there is a **directed “path” to A**
- All the members of a **strongly connected component** have the same in-component
- All members of the **strongly connected component that A belongs to, are members of its out-component**

## Bow Tie Structure in Web Graph



- Directed graphs have weakly and strongly connected components
  - Associated with each strongly connected component is an **out-component** (the set of vertices that can be reached from the strongly connected component but not the reverse) and an **in-component** (the set of vertices that can reach the strongly connected component but are not reachable from it).
- Corollary : If a node would belong to both the in and out component of a strongly connected component, then that node **would be part of the strongly connected component**.
- Web graph is directed and forms a **bowtie**



**THANK YOU**

---

**Bhaskarjyoti Das**

Department of Computer Science Engineering



# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

## Web Search Basics

---

**Bhaskarjyoti Das**

Department of Computer Science Engineering

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

## Understand the Search Users

**Bhaskarjyoti Das**

Department of Computer Science Engineering

- Need [Brod02, RL04]

- **Informational** – want to learn about something (~40% / 65%)

Low hemoglobin

- **Navigational** – want to go to that page (~25% / 15%)

United Airlines

- **Transactional** – want to do something (web-mediated) (~35% / 20%)

- Access a service

Seattle weather

- Downloads

Mars surface images

- Shop

Canon S410

- **Gray areas**

- Find a good hub

Car rental Brasil

- Exploratory search “see what’s there”

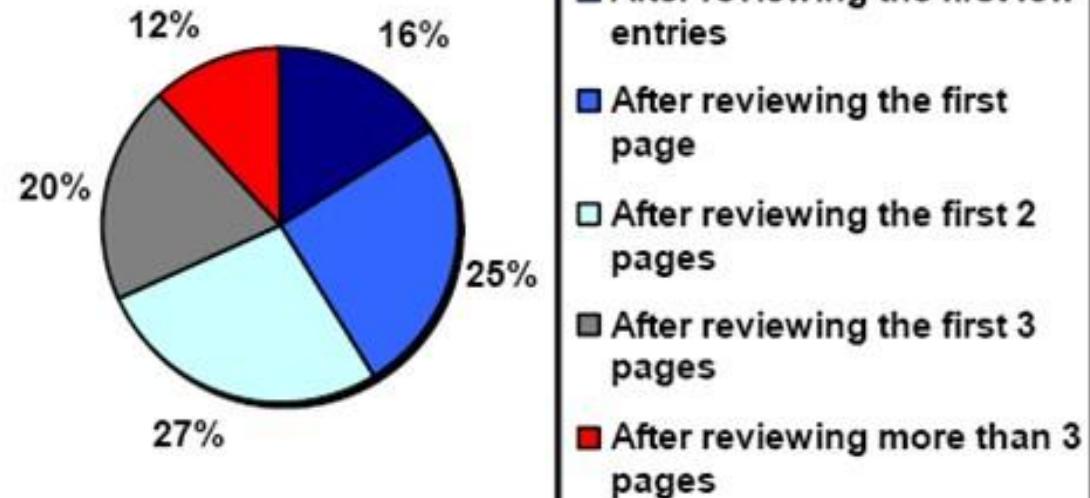
## Why it Matters ?

---

- Discerning the kind of query may be important
- May govern algorithmic search result and suitability for sponsored search results
- Example – informational query may require comprehensive listing; very good scope search !
- Example – navigational query may require only the target pages ; does not offer much for sponsored search !
- Example – transactional query may be super important for the ads & sponsored search

## How far do people look for results?

"When you perform a search on a search engine and don't find what you are looking for, at what point do you typically either revise your search, or move on to another search engine? (Select one)"



(Source: [iprospect.com](http://iprospect.com) WhitePaper\_2006\_SearchEngineUserBehavior.pdf)

Bottom line : do not assume search users are patient  
(;-)

## Users' empirical evaluation of results – Quality of Pages

---

- Quality of pages varies widely
  - Relevance is not enough
  - Other desirable qualities (non IR!!)
    - Content: Trustworthy, diverse, non-duplicated, well maintained
    - Web readability: display correctly & fast
    - No annoyances: pop-ups, etc.
    - UI – Simple, no clutter, error tolerant
  - User perceptions may be unscientific, but are significant over a large aggregate

## Users' empirical evaluation of results - Precision Comprehensiveness

---

- What matters
- Precision vs. recall
  - On the web, recall seldom matters
  - Recall matters **only** when the **number** of matches is very small
- Precision Comprehensiveness – must be able to deal with obscure queries

## Users' empirical evaluation of engines- Other Factors

---

- What matters other than Quality of pages, Precision/recall etc. ?
- Trust – Results are objective
- Coverage of topics for polysemic queries
- Pre/Post process tools provided
  - Mitigate user errors (auto spell check, search assist,...)
  - Explicit: Search within results, more like this, refine ...
  - Anticipative: related searches



**THANK YOU**

---

**Bhaskarjyoti Das**

Department of Computer Science Engineering



# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

## Web Search Basics

---

**Bhaskarjyoti Das**

Department of Computer Science Engineering

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

**SPAM, SEO and SEM**

**Bhaskarjyoti Das**  
Department of Computer Science Engineering

## Sponsored Search and Algorithmic Search

nigritude ultramarine - Google Search - Mozilla Firefox

File Edit View Go Bookmarks Yahoo! Tools Help

Getting Started Latest Headlines

Yahoo! Mail My Yahoo! Games Movies Music Answers Personals Sign In

pragh60@gmail.com | My Account | Sign out

Google Web Images Groups News Froogle Local more »

nigritude ultramarine Search Advanced Search Preferences

Results 1 - 10 of about 185,000 for **nigritude ultramarine**. (0.35 seconds)

**Anil Dash: Nigritude Ultramarine**  
Do me a favor: Link to this post with the phrase **Nigritude Ultramarine**. ... Just placed a link to your **Nigritude Ultramarine** article on my weblog. Cheers! ...  
[www.dashes.com/anil/2004/06/04/nigritude\\_ultra](http://www.dashes.com/anil/2004/06/04/nigritude_ultra) - 101k - Mar 1, 2006 -  
Cached - Similar pages

**Nigritude Ultramarine FAQ**  
Nigritude Ultramarine FAQ - frequently asked questions about **nigritude ultramarine** and the realted SEO contest.  
[www.nigritudeultramarines.com/](http://www.nigritudeultramarines.com/) - 59k - Cached - Similar pages

**SEO contest - Wikipedia, the free encyclopedia**  
The **nigritude ultramarine** competition by SearchGuild is widely acclaimed as ...  
Comparison of search results for **nigritude ultramarine** during and after the ...  
[en.wikipedia.org/wiki/Nigritude\\_ultramarine](http://en.wikipedia.org/wiki/Nigritude_ultramarine) - 37k - Cached - Similar pages

**Slashdot | How To Get Googled, By Hook Or By Crook**  
The current 3rd result showcases the "Nigritude Ultramarine Fighting Force" who ... When discussing **nigritude ultramarine** [slashdot.org] it is important to ...  
[slashdot.org/article.pl?sid=04/05/09/1840217](http://slashdot.org/article.pl?sid=04/05/09/1840217) - 110k - Cached - Similar pages

**The Nigritude Ultramarine Search Engine Optimization Contest**  
It's sweeping the web -- or at least search engine optimizers -- a new contest to rank tops for the term **nigritude ultramarine** on Google.  
[searchenginewatch.com/sereport/article.php/3360231](http://searchenginewatch.com/sereport/article.php/3360231) - 57k - Cached - Similar pages

**Sponsored Links**

**Business Blogging Seminar**  
Coming to L.A. March 16  
Top bloggers reveal key techniques  
[www.blogbusinesssummit.com](http://www.blogbusinesssummit.com)  
Los Angeles, CA

**Full-Time SEO & SEM Jobs**  
Find companies big & small hiring full-time SEO & SEM pros right now  
[CareerBuilder.com](http://CareerBuilder.com)

**SEO Contests**  
Information on SEO Contests like the **Nigritude Ultramarine** contest.  
[www.seo-contests.com/](http://www.seo-contests.com/)

**The SEO Book**  
**Nigritude Ultramarine** & SEO secrets  
Fun, free, raw, & different.  
[www.seobook.com](http://www.seobook.com)

**Ultramarine - Companion**  
Music - Dance - Electronic  
[Overstock.com](http://Overstock.com)

Done

## SEM and SEO : Typically Two Separate Organizations

---

- SEM ( Search Engine Marketing)

- Understand how search engines rank advertisements
- What should be the strategy for bidding for Ad Keywords
- The goal is to have the ads from the organization secure a top rank in the ads shown by the search engine by spending optimum \$

- SEO ( Search Engine Optimization)

- Understand how search engines rank pages
- The goal to have the web pages of the organization secure top rank in the pages shown by search engine for a given query ( having certain keywords)

- Where you pay money to the Search Engine ? Only for the ads. That is why SEO ( not SEM !) plays an important role!

## The Trouble with Paid Search Ads and Importance of SEO

---

- It costs money. What's the alternative?
- *Search Engine Optimization:*
  - “Tuning” your web page to rank highly in the algorithmic search results for select keywords
  - Alternative to paying for ad placement
  - Thus, intrinsically a marketing function
- Performed by companies, webmasters and consultants (“Search engine optimizers”) for their clients
- Some perfectly legitimate, some very shady

## SPAM (Part of Search Engine Optimization ? )

---

- Motives

- Commercial, political, religious, lobbies
- Promotion funded by advertising budget

- Operators

- Contractors (Search companies) Engine Optimizers) for lobbies,
- Web masters
- Hosting services

- Forums

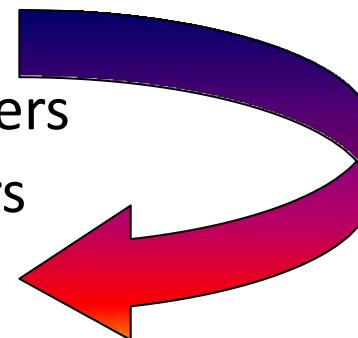
- E.g., Web master world ( [www.webmasterworld.com](http://www.webmasterworld.com) )
  - Search engine specific tricks
  - Discussions about academic papers ☺

## Simplest forms – Keyword Stuffing

---

- First generation engines relied heavily on *tf/idf*
  - The top-ranked pages for the query **maui resort** were the ones containing the most **maui's** and **resort's**
- SEOs responded with dense repetitions of chosen terms ( large term frequency)
  - e.g., **maui resort maui resort maui resort**
  - Often, the repetitions would be in the same color as the background of the web page
    - Repeated terms got indexed by crawlers
    - But not visible to humans on browsers

Pure word density cannot  
be trusted as an IR signal



## Variants of Keyword Stuffing – Meta Tags and Hidden Texts

---

- Defines meta data, that can be parsed but not human readable
- Misleading meta-tags, excessive repetition –
- Hidden text with colors, style sheet tricks, etc.

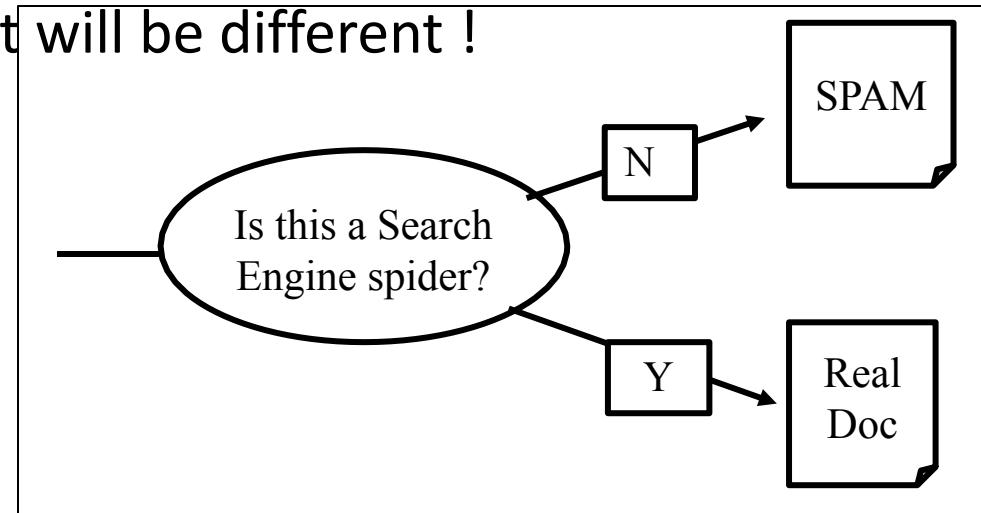
**Meta-Tags =**

"... London hotels, hotel, holiday inn, hilton, discount, booking, reservation, sex, mp3, britney spears, viagra, ..."

## Cloaking – a SPAM Technique

---

- DNS cloaking: Switch IP address. Impersonate
- Serve fake content to search engine spider with misleading keywords
- For the actual user, the content will be different !



## More Spam techniques

---

- **Doorway pages**

- Pages optimized for a single keyword that re-direct to the real target page having content for commercial purpose

- **Clickbaits**

- something (such as a headline) designed to make readers want to click on a hyperlink, especially when the link leads to content of dubious value or interest i.e. “over-promises in its headline and under-delivers in its content,” and “preys on users’ curiosity to drive clicks on its links, but doesn’t have rich, detailed content that the user wants.”

- **Link spamming became popular after Google started link analysis**

- Mutual admiration societies, hidden links, awards
- *Domain flooding:* numerous domains that point or re-direct to a target page

## The war against spam

---

- **Quality signals - Prefer authoritative pages based on:**
  - Votes from authors (linkage signals)
  - Votes from users (usage signals)
- **Policing of URL submissions**
  - Anti robot test
- **Limits on meta-keywords**
- **Robust link analysis**
  - Ignore statistically implausible linkage (or text)
  - Use link analysis to detect spammers (guilt by association)
- **Spam recognition by machine learning**
  - Training set based on known spam
- **Family friendly filters**
  - Linguistic analysis, general classification techniques, etc.
  - For images: flesh tone detectors, source text analysis, etc.
- **Editorial intervention**
  - Blacklists
  - Top queries audited
  - Complaints addressed
  - Suspect pattern detection



**THANK YOU**

---

**Bhaskarjyoti Das**

Department of Computer Science Engineering



# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

## Web Search Basics

---

**Bhaskarjyoti Das**

Department of Computer Science Engineering

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

## Index Size

**Bhaskarjyoti Das**

Department of Computer Science Engineering

## Why Index Size ?

---

- Search Engine **comprehensiveness** grows with index size

- **Issues**

- The web is really infinite
  - Dynamic content, e.g., calendars
  - Soft 404: www.yahoo.com/<anything> is a valid page
- Static web contains syntactic duplication, mostly due to mirroring (~30%)

- **Who cares?**

- Media, and consequently the user
- Engine design
- Engine crawl policy. Impact on recall.

## What can we Attempt to Measure?

---

• Given 2 Search Engines, the relative sizes of search engines ?

- Though the notion of a page being indexed is still *reasonably* well defined.
- **Disclaimer:** This question itself is imprecise
  - Search Engine can return web pages whose content it has not fully indexed . Different engines have different preferences
    - max url depth, max count/host, anti-spam rules, priority rules, etc.
  - Search Engines index different things under the same URL and all of them may not get used for all queries
    - frames, meta-keywords, document restrictions, document extensions, ...
- So there is no single measure of index size as indexes themselves vary in scope

## Ratio of Index Size Of Engines E1 and E2

**Hypothesis :** Each search engines indexes a fraction of the web chosen independently and uniformly at random

- There is a finite (?) web from which each engine is choosing a subset
- Each engine choosing an independent, uniformly chosen subset (?)
- Questionable assumption but that leads us to a statistical method called **capture recapture**

## Capture Recapture Method

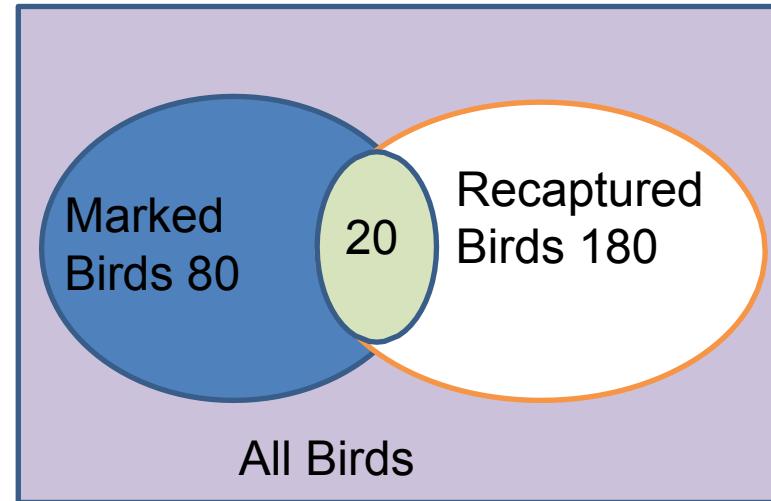
---

- How many birds/mosquitoes... are there in the park? Difficult to count!

### Assumptions 1 :

- Closed population : no birth, death, migration
- No misdiagnosis
- Homogeneity

**Assumption 2 :** cases captured in system 1 should have the same probability to show up in system 2 as those not captured in system 1

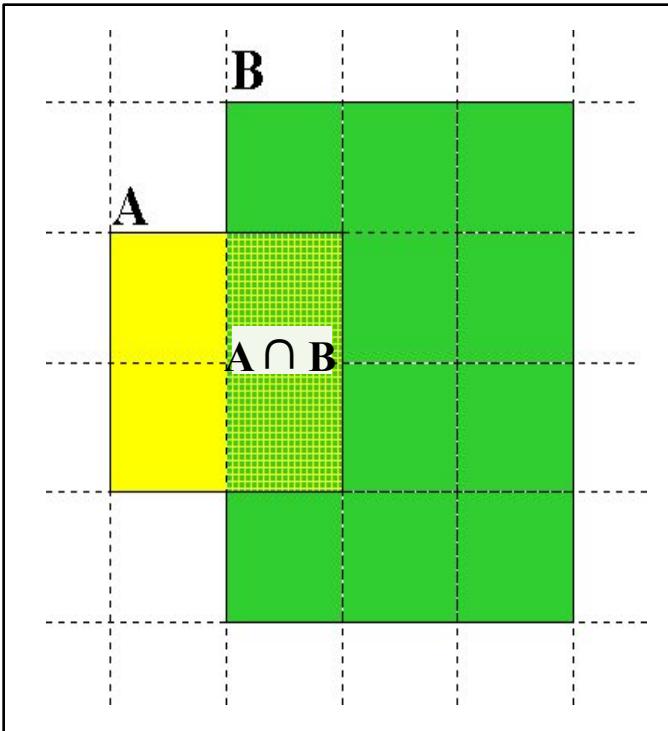


What if ,instead of number 20, a fraction is given ?

$$P(A \text{ AND } B) = P(A) \cdot P(B) \text{ i.e. } \frac{N_{AB}}{N} = \frac{N_A}{N} * \frac{N_B}{N}$$

$$N = (80+20)*(180+20)/ 20 = 1000 \text{ approx}$$

## Capture Recapture Method : Relative Size from Overlap



**Sample** URLs randomly from A

**Check** if contained in B and vice versa

$$A \cap B = (1/2) * \text{Size}$$

$$A \cup B = (1/6) *$$

$$\text{Size}_B (1/2) * \text{Size}_A = (1/6) * \text{Size}_B$$

$$\text{Therefore, } \text{Size}_A / \text{Size}_B =$$

$$(1/6) / (1/2) = 1/3$$

*We pick up a random page of index of E1 and test if it is in E2 index and vice versa . It may lead us to  $x * |E1| = y * |E2|$*

**Each test involves:** (i) Sampling (ii)  
Checking

- Ideal strategy: Generate a random URL and check for containment in each index.
- Problem: Random URLs are hard to find!
- Bias inherent in each strategy !

### •Approach 1 : Random searches

- Begin with a search log and send a random search from this to E1 along with a random page from result
- But
  - logs are with search engines only and every search engine user group is biased
  - So it is not really a random search !

### •Approach 2 : Random IP address

- Generate a random IP address and send a request to web server at that collecting all pages at that server
- But
  - web Server itself can be skewed in types of pages it has
  - Many hosts may use virtual address and it is negating the effect of random IP

- **Approach 3 : Random Walk**

- from an arbitrary page and this walk will finally converge to a steady state where probability of picking a page is constant ! (more on this in PageRank)

- **But**

- the web need not be strongly connected !
- We do not know how much time it would take for achieving this steady state ( and we are not Google !)

### •Approach 4 : Random Query

- Pick a set of random terms from a web dictionary (put together by crawling a part of web)
  - Build a random conjunctive query on E1
  - Pick from top 100 results on E1 , a page p at random
- We then test p for presence in E2 by choosing some terms in p and using it as a conjunctive query on E2
- But
  - Picking from top 100 results of E1 introduces the bias of E1 (ranking algorithm bias)
  - We do not know if E2 will handle this multiword test query properly ( treating them as robotic spam)
  - Operational issues

## Conclusions

---

- No sampling solution is perfect.
- Lots of new ideas ...
- ....but the problem is getting harder
- Quantitative studies are fascinating and a good research problem



**THANK YOU**

---

**Bhaskarjyoti Das**

Department of Computer Science Engineering



# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

## Handling Near Duplicates

---

**Bhaskarjyoti Das**

Department of Computer Science Engineering

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

## Handling Near Duplicates - 3

**Bhaskarjyoti Das**

Department of Computer Science Engineering

## Exact Duplicate

---

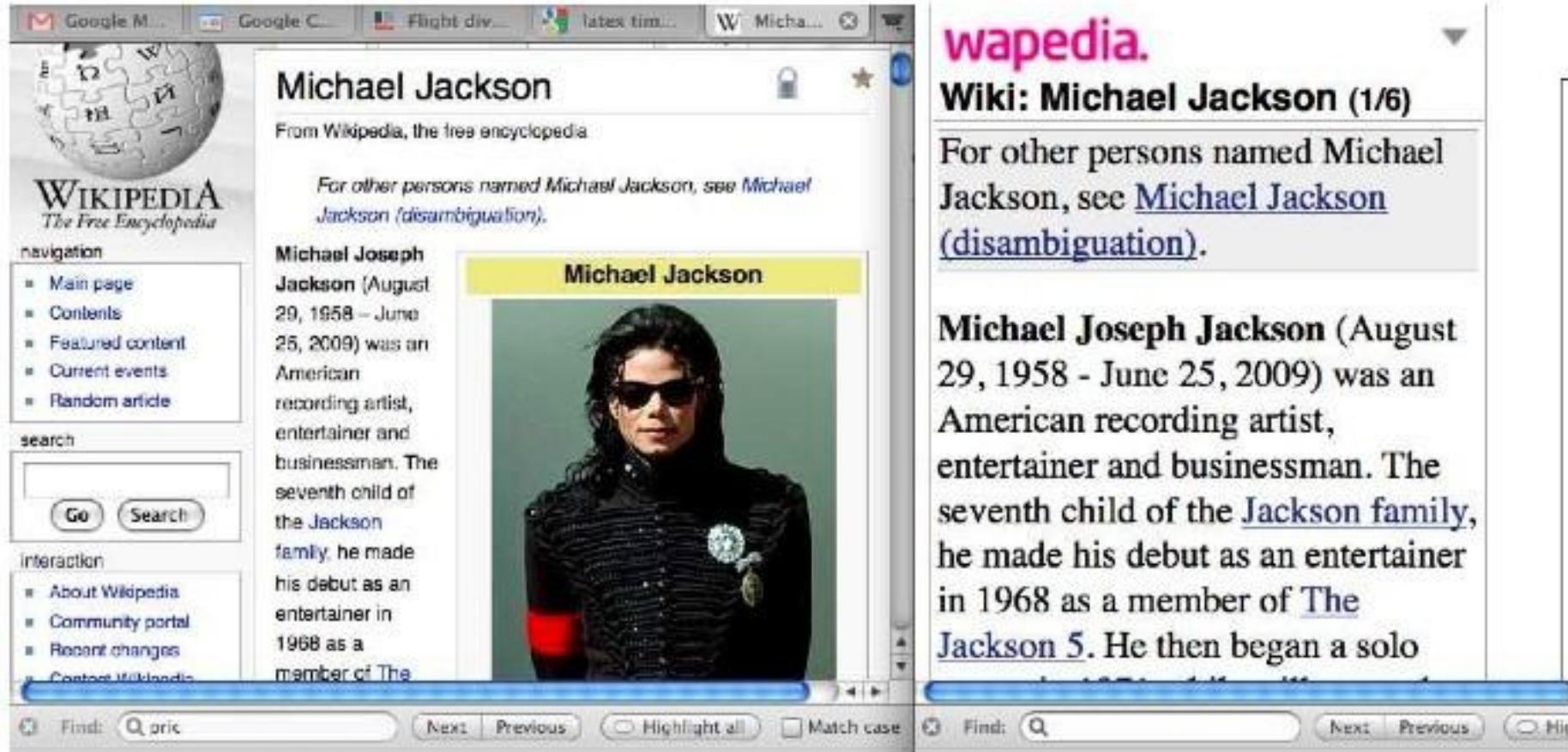
- The web is full of **duplicated content** ( 30% in 2003)
- Exact duplicate
  - Important to detect and remove “exact duplicate”
  - Content is identical bit by bit
  - Non-content part may vary
  - Not as common

## Near Duplicate

---

- Near duplicates
- Very similar but not exactly similar
- Marginal relevance is zero: even a highly relevant document becomes nonrelevant if it appears below a (near-)duplicate
- Examples
  - last-modified date the only difference between two copies of a page
  - Reworded few sentences

## Near-Duplicate Examples



The image shows two side-by-side web browser windows. The left window is a Wikipedia page for "Michael Jackson" from the English Wikipedia. The right window is a search result from a search engine, likely Google, for the same query. Both pages contain nearly identical text about Michael Joseph Jackson, including his birth date (August 29, 1958), death date (June 25, 2009), and his status as an American recording artist, entertainer, and businessman. Both pages also mention his debut as an entertainer in 1968 as a member of The Jackson 5. The right window includes a search bar at the top and navigation buttons (Find, Next, Previous) at the bottom.

Michael Jackson

From Wikipedia, the free encyclopedia

For other persons named Michael Jackson, see Michael Jackson (disambiguation).

Michael Joseph Jackson (August 29, 1958 – June 25, 2009) was an American recording artist, entertainer and businessman. The seventh child of the Jackson family, he made his debut as an entertainer in 1968 as a member of The Jackson 5. He then began a solo

wapedia.

Wiki: Michael Jackson (1/6)

For other persons named Michael Jackson, see [Michael Jackson \(disambiguation\)](#).

Michael Joseph Jackson (August 29, 1958 - June 25, 2009) was an American recording artist, entertainer and businessman. The seventh child of the [Jackson family](#), he made his debut as an entertainer in 1968 as a member of [The Jackson 5](#). He then began a solo

## Detecting Near- Duplicates : Syntactic vs. Semantic ?

---

- We want “syntactic” (as opposed to semantic) similarity.
- True semantic similarity (similarity in content) is too difficult to compute.
- We do not consider documents near-duplicates if they have the same content, but express it with different words (semantically duplicate).
- Use similarity threshold  $\theta$  to make the call “is/isn’t a near-duplicate”. Example - two documents are near-duplicates if similarity  $> \theta = 80\%$ .

## Detecting Near-Duplicates by Set Similarity

---

- Document as Set
- Set Similarity: similarity between two articles can be thought of similarity between two set of words
- Do this by calculating the Jaccard similarity between each pair of documents, and then selecting those with a similarity above some threshold.
- For N documents, it will be  ${}^N C_2$  or the order of  $N^2$ 
  - For a collection of 1 million documents, it is 500 Billion comparisons
  - If one comparison takes 1 ms, it translates to 500 million seconds or 16 years !!

## Duplicate / Near- Duplicate Detection – Naïve Approach

---

- *Duplication:* Exact match can be detected with **fingerprints**
- *Naïve Approach*
  - *Compare every document against all others*
  - *Comparison is by means of a calculated finger print that considers all words in a document*
  - *Complexity (  $n^2d$  ) where n= no of documents, d is no of words on a document*
  - *Impractical and not feasible*
  - *Will declare “near duplicates” as different documents !!*

## Shingles - a Possible Approach

---

1. Jaccard similarity on the words **doesn't take into account word order**.  
So, Instead of using words, we use **shingles**.
  
2. Shingles are consecutive overlapping sequences of letters. By using these instead of the words directly, we can get an idea about how the words are related to each other.
  
3. There are many ways to build shingles (typically consecutive letters).
  - Once we decided how we're going to shingle
  - we need to decide how large shingles should be.

## Small Vs.Large Shingles

---

1. We need to decide how large shingles should be.
2. We want shingles that are small enough, that they appear in more than one article, but not so small that they appear in too many articles. Too large will appear in too few documents.
3. In general, shingle size should be proportional to average length of document in corpus i.e. 9 or 10 are common sizes.
4. Once we shingle all the articles (articles are now set of shingles), we calculate the Jaccard similarity for all pairs of articles ( article = set of shingles)
5. Then group the ones that are most similar according to some threshold

## Shingles Example (in Words)

- Consider an example where shingle consists of a number of words.
- Each shingle contains a set number of consecutive words, and a document is broken down into total words - single length + 1 number of shingles.
- Example : document contained a single sentence of "The quick brown fox jumps over the lazy dog"
- **5 word long shingles :**
  - The quick brown fox jumps
  - quick brown fox jumps over
  - brown fox jumps over the
  - fox jumps over the lazy
  - jumps over the lazy dog
  - So our document as a set looks like:

Set S = new Set(["The quick brown fox jumps", "quick brown fox jumps over", "brown fox jumps over the", "fox jumps over the lazy", "jumps over the lazy dog"]).

## Should We Compare all Shingle Sets ?

---

- Comparing *all pairs of documents* for shingle set in each document will not scale – **still  $O(n^2)$** . This is high time complexity!
- The Document Matrix (*shingle vs. Document*) is a sparse and large matrix . This is high space complexity !

## Further Optimization : Sketch

---

- Compare sets of randomly selected shingles of size 5 from two documents.
- So for a document d1 that is 10000 words long, we break it down into 9996 shingles, and **randomly select say 200 of those to represent the document. We call it a sketch!**
- If the document d2 is 20000 words long, we boil that down from 19996 shingles to another set of 200 randomly selected shingles.
- Now instead of matching 9996 shingles to 19996 other shingles, **compare 200 shingles to another 200 shingles. Reduced our workload at the expense of accuracy**, but this is still good enough for most scenarios.

## Another Optimization : Hash the Shingles

---

- Treating each shingle as a sequence of characters not efficient from either a speed or space perspective.
- Represent each shingle as a unique number.
  - By *hashing* the shingle using a 64 bit hash function is used that minimizes collision and can handle large shingle
  - This also saves space as a single number is stored instead of a shingle
  - Hashing the shingles gives us a speedup because we simply need to look at one number — their hash



**THANK YOU**

---

**Bhaskarjyoti Das**

Department of Computer Science Engineering



# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

## Handling Near Duplicates

---

**Nagegowda K S**

Department of Computer Science Engineering

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

## Handling Near duplicates - 2

Nagegowda K S

Department of Computer Science Engineering

## Represent a Document in Terms of Shingles

- We get a document-shingles matrix that is sparse

		Documents			
		1	1	1	0
		1	1	0	1
Shingles		0	1	0	1
		0	0	0	1
		1	0	0	1
		1	1	1	0
		1	0	1	0

## Jaccard Index : Are We All Set ?

---

- We have a representation of each document in the form of shingles.
- We need a metric to measure similarity between documents. Jaccard Index is a good choice for this.

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

## We Need Locality Sensitive Hashing ...

---

- After all the optimizations previously mentioned (sketch of shingles, representing shingle by a hash value etc.) , it still will not scale !!
- Every time a new document comes in the corpus, this similarity has to be calculated for all existing documents
- With growth in corpus, this will soon become unmanageable and the system will become slower
- Requirement : a way to group documents together without explicitly comparing with every other

## General Idea of Locality Sensitive Hashing

---

- The general idea of LSH is to **find an algorithm such that if we generate signatures of 2 documents**, it tells us if their similarity is greater than a threshold **t in a method that is not brute force Jaccard Similarity.**
- The idea is : **this algorithm itself will deliver an approximate value of Jaccard Similarity** had it been done in a brute force way !

## What Kind of Hashing Given We are Inclined to Jaccard Similarity ?

---

- For a document  $d$ ,  $H(d)$  is the signature or hash for that document

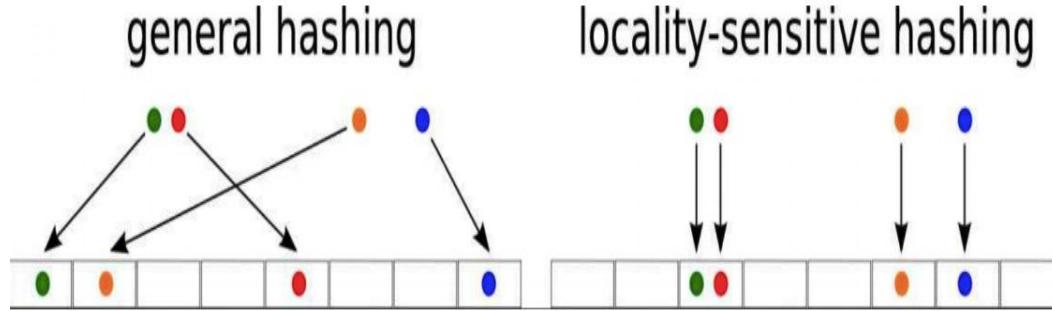
- **Key Idea :**

- If  $\text{similarity}(d_1, d_2)$  is high then  $\text{Probability}(H(d_1) == H(d_2))$  is high
- If  $\text{similarity}(d_1, d_2)$  is low then  $\text{Probability}(H(d_1) == H(d_2))$  is low

- **Key : choice of hashing function is tightly linked to the similarity metric.**

- For Jaccard similarity the appropriate hashing function is **min-hashing**.

## General Hashing vs. Locality Sensitive Hashing



- Min Hashing is a type of Locality Sensitive Hashing (LSH)
- LSH is a hashing method where similar items are mapped to the same cell in the hash table with good probability.
  - Still want non-similar items to have a low collision probability, to make search efficient.
- This makes it easier to identify observations with various degrees of similarity
- Applications: detection of near duplicate document, detection of similar gene expression, audio/video finger printing

## MinHash, SimHash and Locality Sensitive Hashing ( LSH)

- Minhash is a Locality Sensitive Hashing algorithm. A major difference of MinHash and SimHash is the probability of hash collisions.
- In case of SimHash it is equal to the cosine similarity and in case of MinHash it is equal to the Jaccard similarity.
- Depending how you define the similarity between sets, one or the other algorithm could be more appropriate.
- Regardless of the chosen hashing algorithm, the values of the calculated signature are equally partitioned over a certain number of bands.
- If the signatures of any two sets are identical within at least one band, the corresponding pair of sets is selected as candidate for similarity.

## Minhash and it's Inventor

---

- It is a Clustering Algorithm!
- Invented by Andrei Broder in 1997 and used by Altavista Search Engine. He was VP of Computational Advertisement @ Yahoo!, Distinguished Scientist at Google
- Applications in document similarity, clustering documents, Shopping cart analysis, behavior segmentation, Market Basket Analysis etc.

## MinHash Signatures to Get an Estimate Of Jaccard Similarity

---

- The MinHash algorithm provides a fast approximation to the Jaccard Similarity between two sets.
  - For each set of shingles for a document, calculate a MinHash signature.
    - MinHash signatures will all have a fixed length, independent of the size of the set.
    - And the signatures will be relatively short compared to the set
  - To approximate the Jaccard Similarity between two sets, take their MinHash signatures, and simply count the number of components which are equal.
  - Divide this count by the signature length to get a pretty good approximation to the Jaccard Similarity

## MINHASH – Step 1

---

- Random permutation ( $\pi$ ) of row index of document shingle matrix.
- Not showing the permuted version (but indicating by index)
- Ex- 5<sup>th</sup> row is the 1<sup>st</sup> in order and the top row becomes the 2<sup>nd</sup>

C1 C2 C3 C4

Permutation  $\pi$  Input matrix (Shingles x Documents)

2	1	0	1	0
3	1	0	0	1
7	0	1	0	1
6	0	1	0	1
1	0	1	0	1
5	1	0	1	0
4	1	0	1	0

## MINHASH – Step 1

---

- Hash function is the *index of the first (in the permuted order) row in which column C has value 1.*

$$h_{\pi}(C) = \min_{\pi} \pi(C)$$

Permutation  $\pi$    Input matrix (Shingles x Documents)

→

2	1	0	1	0
3	1	0	0	1
7	0	1	0	1
6	0	1	0	1
1	0	1	0	1
5	1	0	1	0
4	1	0	1	0

C1 C2 C3 C4

,,, , 1 ,,,, , 1

## Minhash step 2 : Do this Several Times Using Different Permutations

- Permutation 1 – Index 1 : x 1 x

1

- Permutation 1 – index2 : 2 1 2

1

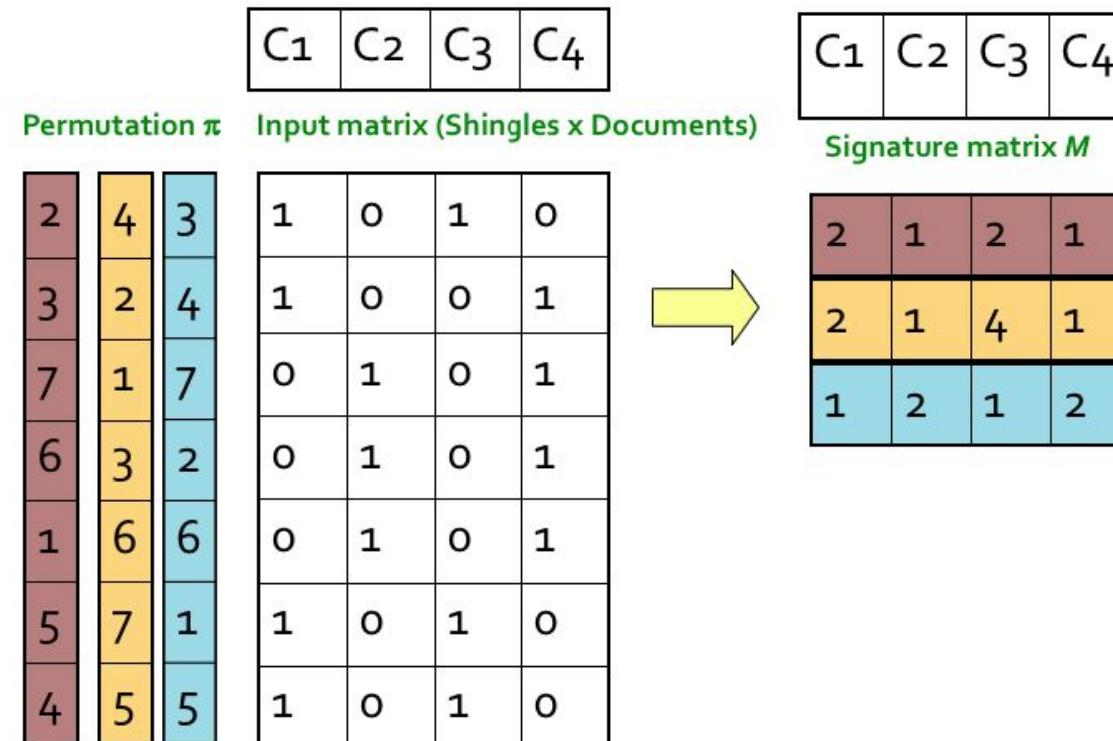
- Permutation 1 : done

- Permutation 2 – index 1: x1x1

- Permutation 2 – index 2 : 21x1

- Permutation 2 –index 3 : 21x1

- Permutation 2 –index 4 : 2141



Like this create the signature matrix for x permutations !

## Minhash Property

- The **similarity of the column signatures** is the fraction of the min-hash functions (=rows) in which they agree !
- So the similarity of signature for C1 and C3 is  $2/3$  as 1<sup>st</sup> and 3<sup>rd</sup> row are same

C <sub>1</sub>	C <sub>2</sub>	C <sub>3</sub>	C <sub>4</sub>
----------------	----------------	----------------	----------------

Signature matrix  $M$

2	1	2	1
2	1	4	1
1	2	1	2

Minhash Property : Expected Similarity of Two Signature Columns

---

Claim:  $P[h_\pi(C_1) = h_\pi(C_2)] = \text{sim}(C_1, C_2)$

- Expected similarity of two signatures (=columns) is equal to the Jaccard similarity of the columns. **The longer the signatures, the lower the error !**
  - A : where same value is present in both ; B : Present in either ; C : in neither of them
  - Formula for the Jaccard index as  $a/(a+b)$  where a is a number of rows of type A and b of type B (skipping type C rows)
  - With random permutations, the probability that two documents will have an equal fingerprint component i.e. Type A row, from the set of A and B rows?  $P = a/(a + b)$  which is exactly the same as Jaccard index!
  - Hence approximations were close and gives an intuition of why more permutations mean better approximations !

## Permutations, Input Matrix, Signature Matrix and Jaccard

Permutation

6	7	1
3	6	2
1	5	3
7	4	4
2	3	5
5	2	6
4	1	7

Input Matrix

0	1	1	0
0	0	1	1
1	0	0	0
0	1	0	1
0	0	0	1
1	1	0	0
0	0	1	0

Signature Matrix

3	1	1	2
2	2	1	3
1	5	3	2

With more permutations,  
Expected similarity of two  
signatures will be approximately  
equal to Jaccard Coefficient

	1-2	2-3	3-4	1-3	1-4	2-4
Jaccard	1/4	1/5	1/5	0	0	1/5
Signature	1/3	1/3	0	0	0	0

## What is Good About Minhash Signature ?

---

- To approximate the Jaccard Similarity between two documents, we will take their **MinHash signatures**, and simply count the number of components which are equal.
- If you divide this count by the signature length, you have a pretty good approximation to the Jaccard Similarity between those two sets.
- The **MinHash algorithm** will provide us with a fast approximation to the Jaccard Similarity between two sets !
- For each document , calculate a **MinHash signature of a fixed length**, independent of the size of the set of shingles that represent the document.
- And the **signatures will be relatively short**

## Are Permutations Really Used in Implementations ?

---

- We now have an algorithm which could potentially perform better
- With larger corpus,
  - bigger will be the dictionary of shingles and higher will be the cost of creating permutations, both in time and hardware.
  - Hard to pick a random permutation from a large no of rows and representing all of them require so much space
- So, in the actual implementation of **this permutation operation is done by using many hash functions**

- **Original Idea : For each document d:**

- Generate K bit hash code
- Accordingly put the document in a particular bucket
  - If there are other documents in the same bucket, then there is a possible collision implying “possible duplicates”
  - Compare these possible duplicates for duplication
- **What is the issue :** This method will catch exact duplicate but will NOT catch near duplicates

## What is Remaining : An Intuition

---

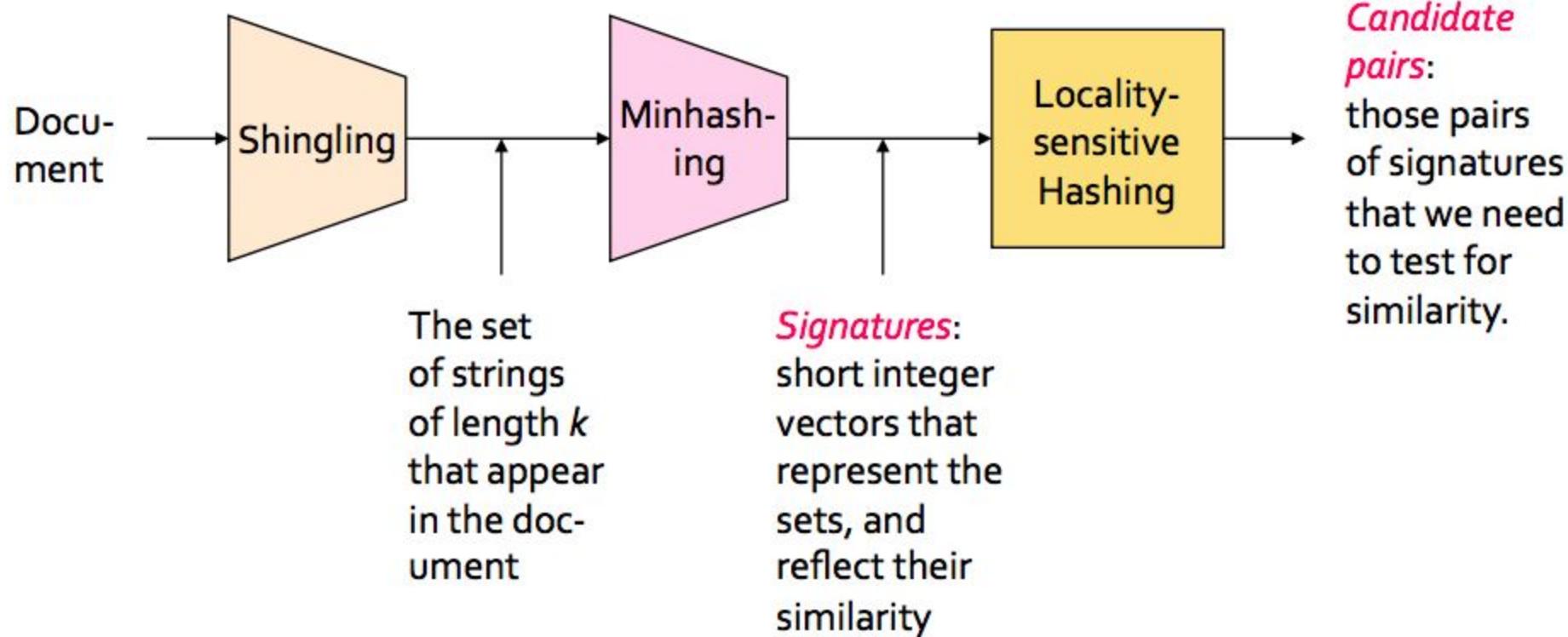
- **Idea For the Remaining Part Of MinHash:**

- Repeat the above process of generating signature matrix n times with m different hash table or hash function – each of them K bits long ( a long string )
- Instead of using the full k bits, use subsequences as keys into the hash tables or final buckets so that we have collisions for even near duplicates and catch near duplicates

H1	H 2	H 3
D1 : 0100	1011	0100
D2 : 1100	1011	0010

## Finding Near Duplicates Is A Three Step Process

It is a three step process..





THANK  
YOU

**Nagegowda K S**

Department of Computer Science Engineering



# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

## Handling Near Duplicates

---

**Bhaskarjyoti Das**

Department of Computer Science Engineering

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

## Handling Near Duplicates - 3

**Bhaskarjyoti Das**

Department of Computer Science Engineering

## MinHash Recap

---

- Document is a **set of shingles** ( each Shingle is a number by hashing for our convenience)
- The idea was to have a **signatures** of sets of shingles for a document instead of this column of 1 and 0
- **Important desired property of signatures :** we can compare the signatures of two sets and **estimate Jaccard Similarity of the sets**
- To minhash a **set (represented by a column of the characteristic matrix)**,
  - pick a permutation of the rows. Each permutation produces a row  $a$  of the signature matrix by applying a hash function.
  - The **minhash value of any column** is the **index of the first row**, in the permuted order, in which the column has a 1

Claim:  $P[h_\pi(C_1) = h_\pi(C_2)] = \text{sim}(C_1, C_2)$

- Remarkable connection between the MinHash and Jaccard Similarity
- **Intuitive explanation:**
  - A : where same value is present in both ; B : Present in either ; C : in neither of them ( we are not concerned about C)
  - Formula for the **Jaccard index is  $a/(a+b)$**  where a is a number of rows of type A and b of type B (skipping type C rows) as  $x = C_1 \cap C_2$  and  $y = C_1 \cup C_2$
  - If we proceed **from top of signature matrix**, the **Probability that we will meet a type A row is  $a/(a+b)$**  and this happens for  $h_\Pi(C_1) = h_\Pi(C_2)$
  - Conversely, proceeding **from top of signature matrix**, If we meet a type B row first, then it is a case of  $h_\Pi(C_1) \neq h_\Pi(C_2)$

## The Last Remaining Step : Banding

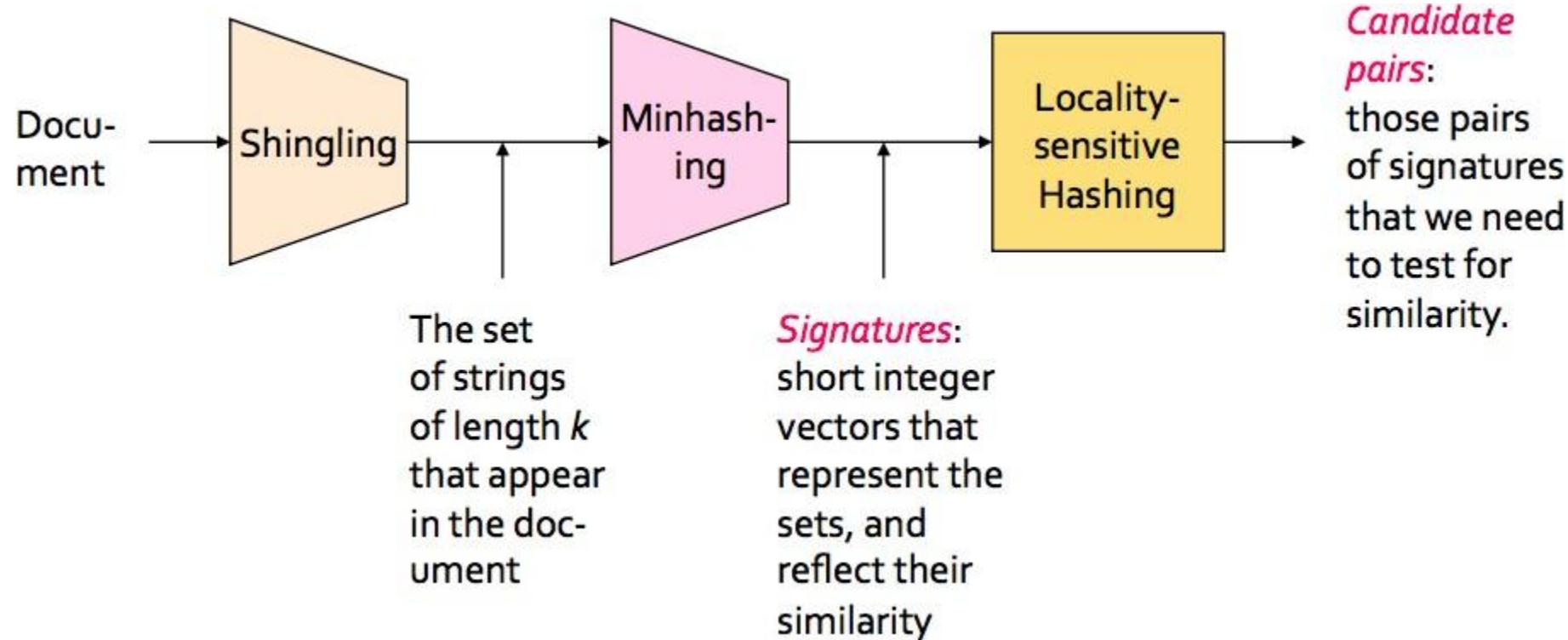
- Process of generating signature matrix with a hash function produces K bits long signature !
- In the diagram for D1 and D2, hashing the full signature matrix will put them in separate bucket and we will NOT detect the “near duplicate” !
- If 2 documents **hash into same bucket** for **at least one of the hash function** (in this case H2) we can take the 2 documents as a candidate near duplicate pair
- How does this “hashing into same buckets” happen ? That is the **Banding Technique**

H1	H 2	H 3
D1 : 0100	1011	0100
D2 : 1100	1011	0010

Note: D and H are interchanged location wise

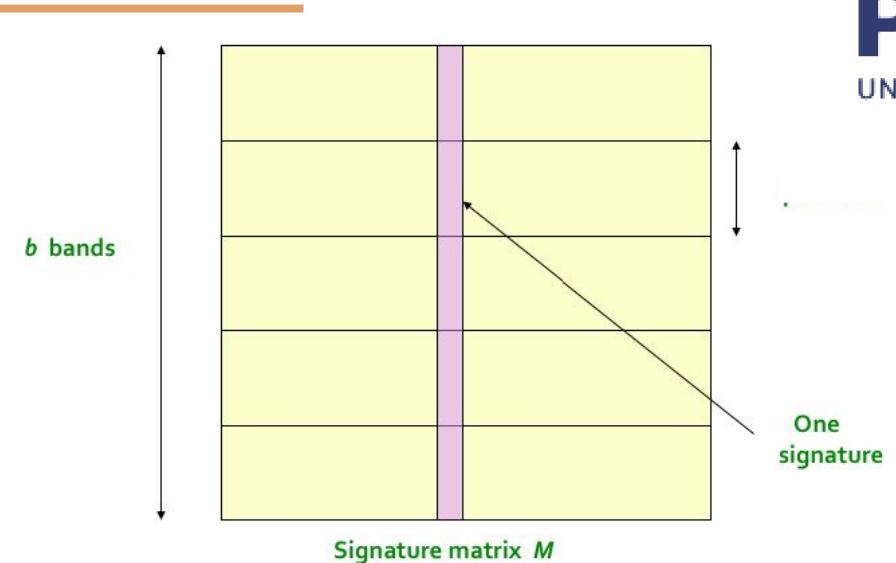
## Finding Near Duplicates Is A Three Step Process

It is a three step process..



## Locality Sensitive Hashing – Band Partition

- Divide the signature matrix into  $b$  bands, each band having  $r$  rows
- For each band, hash its portion of each column to a hash table with  $k$  buckets
- Candidate (suspects and not convicts ;- ) column pairs are those that hash to the same bucket for *at least 1 band*
- **Tune b and r** to catch most similar pairs but few non similar pairs



## Tuning b and r – False Negative Case

---

- 100k documents stored as signature of length 100 i.e. Signature matrix:

100\*100000

•  $b=20 \Rightarrow r = 5$

- **Requirement :** 2 documents (D1 & D2) with 80% similarity to be hashed in the same bucket for at least one of the 20 bands.

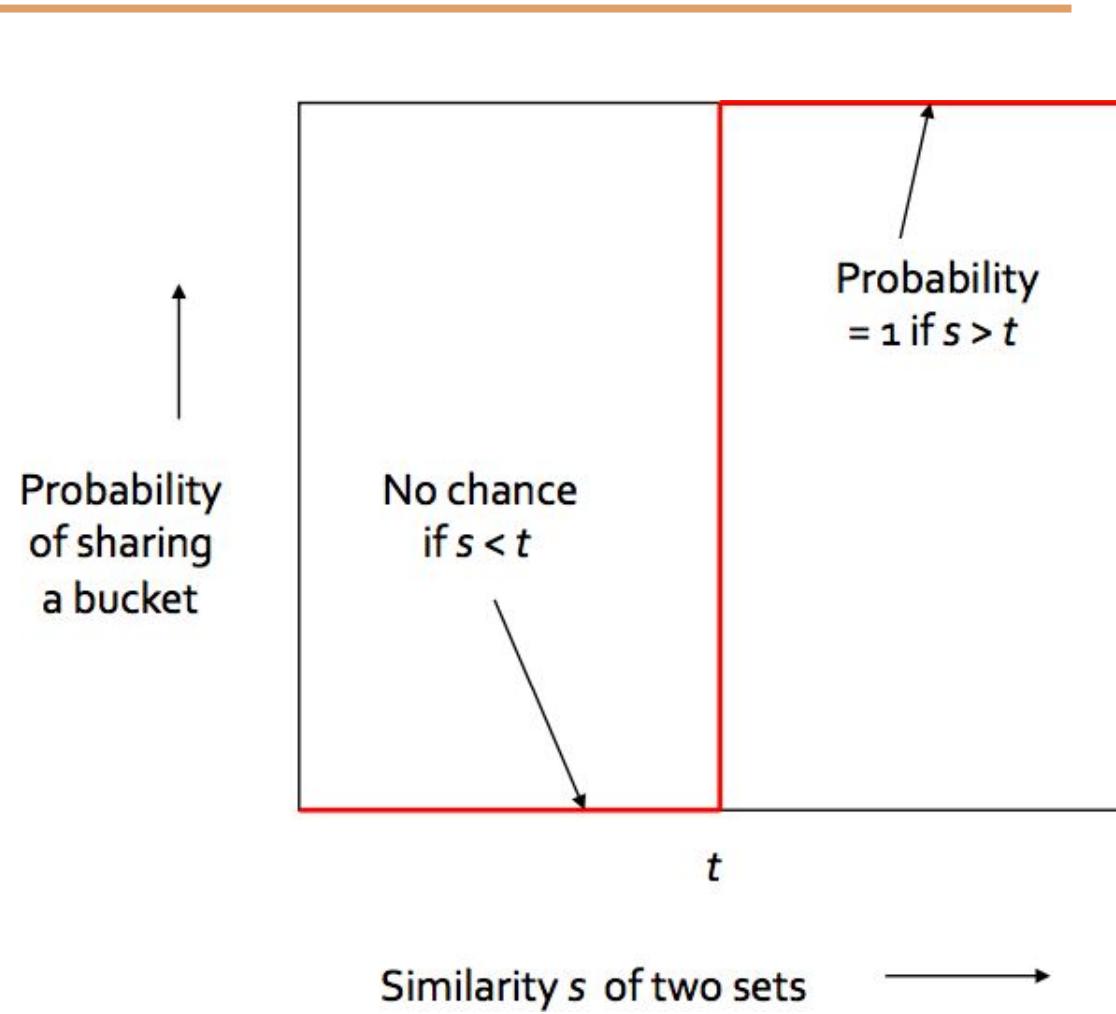
- Threshold = 80%
- 80% similar documents imply 80% chance of  $h_{\Pi}(C_1) = h_{\Pi}(C_2)$
- Since  $r=5$ ,  $P(D1 \& D2 \text{ identical in a particular band}) = (0.8)^5 = 0.328$
- $P(D1 \& D2 \text{ are } \underline{\text{not similar in all 20 bands}}) = (1-0.328)^{20} = 0.00035$
- This means in this scenario we have ~.035% chance of a **false negative** for 80% similar documents.

## Tuning b and r – False Positive Case

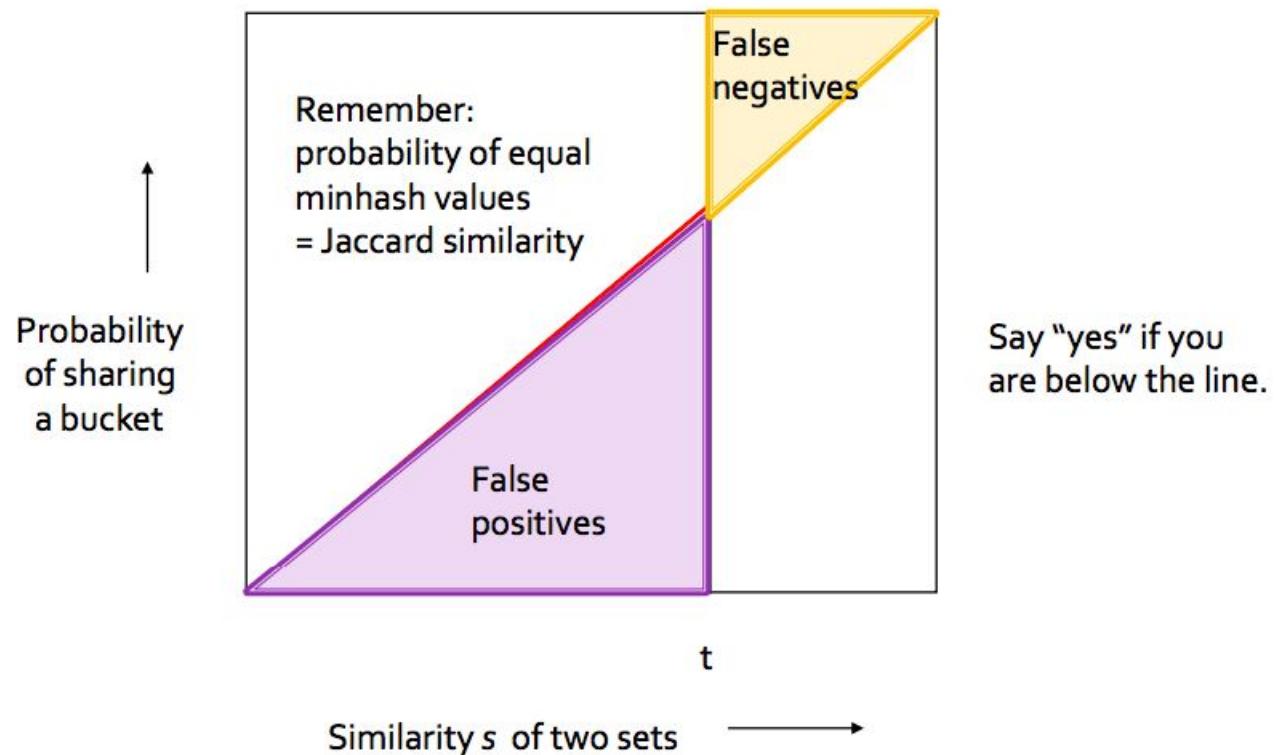
- Same Threshold = 80% but we want to ensure that there is no false positive case for documents with lower similarity i.e. 30%
- Additional Requirement : 2 documents (D1 & D2) with 30% similarity not to be hashed in the same bucket for at least one of the 20 bands.
  - 30% similar documents imply 30% chance of  $h_{\Pi}(C_1) = h_{\Pi}(C_2)$
  - Since  $r=5$ ,  $P(D1 \& D2 \text{ identical in a particular band}) = (0.3)^5 = 0.00243$
  - $P(D1 \& D2 \text{ are not similar in all 20 bands}) = (1 - 0.00243)^{20}$
  - $P(D1 \& D2 \text{ are similar in at least one of 20 bands}) = 1 - (1 - 0.00243)^{20} = 0.0474$
  - This means in this scenario we have ~4.74% chance of a false positive for 30% similar documents.

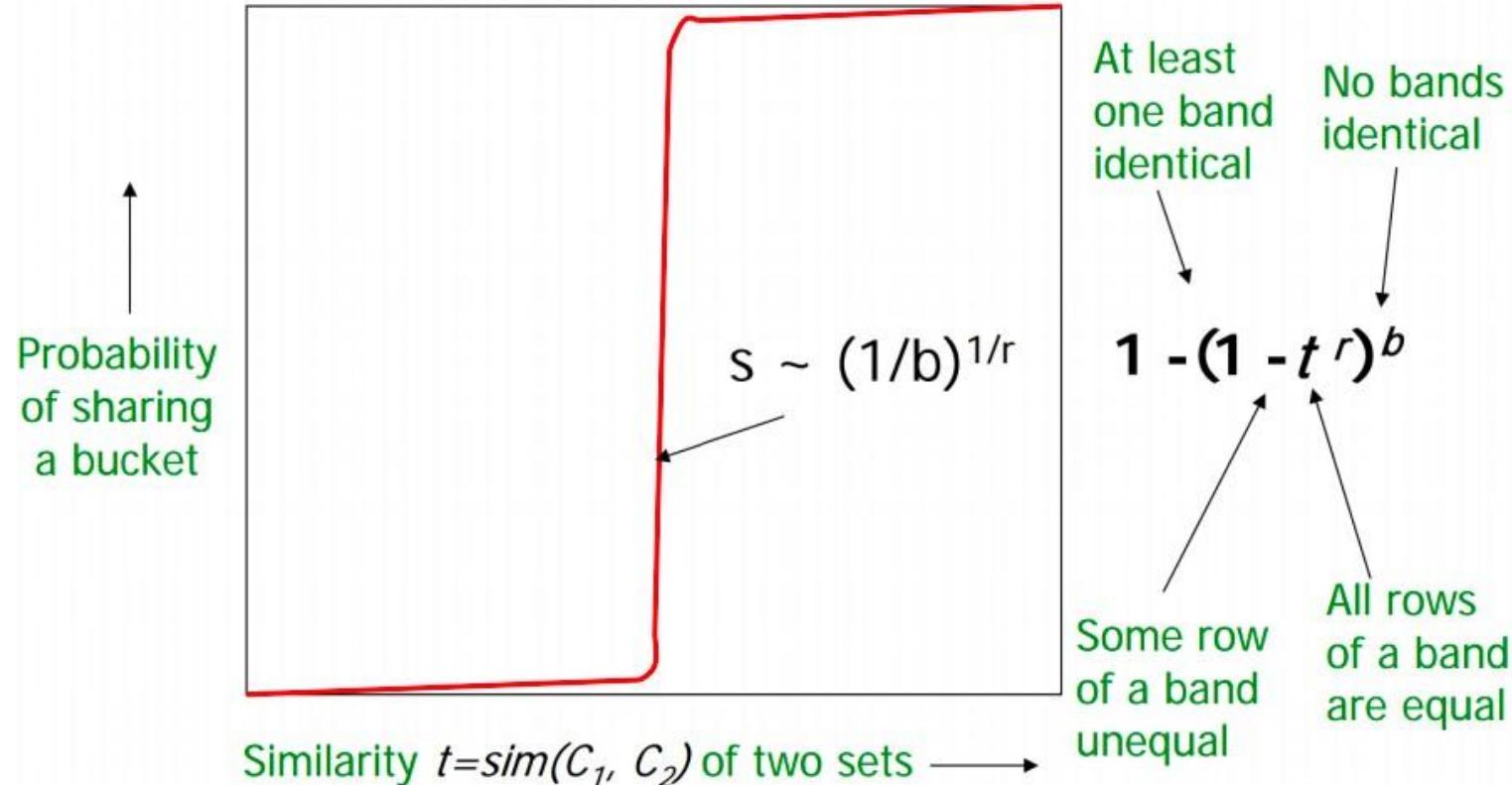
## What We Need Ideally

If 2 documents have similarity greater than the threshold then probability of them sharing the same bucket in at least one of the bands should be 1 else 0.

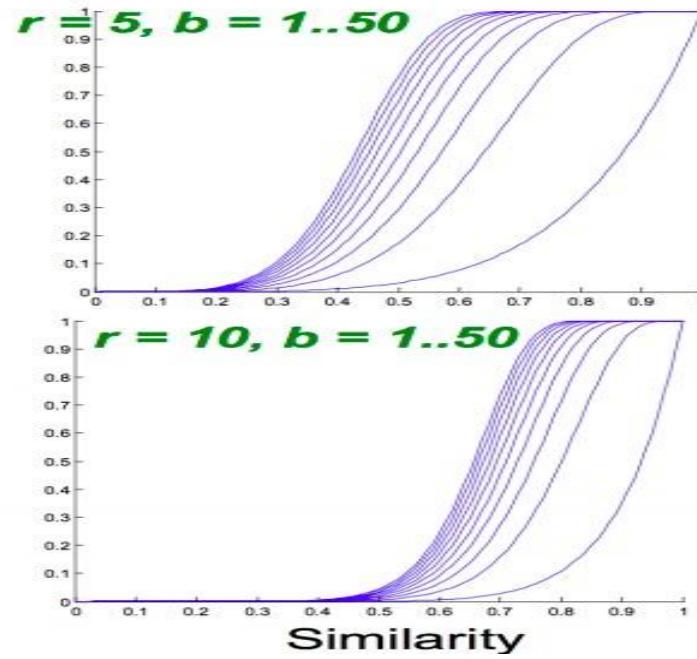
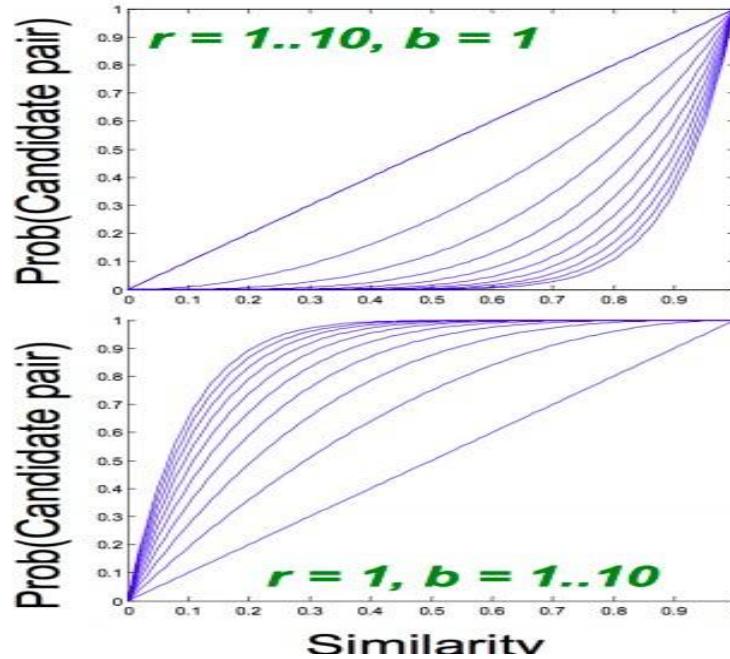


## Worst case if $b=r$





Choose  $b$  and  $r$



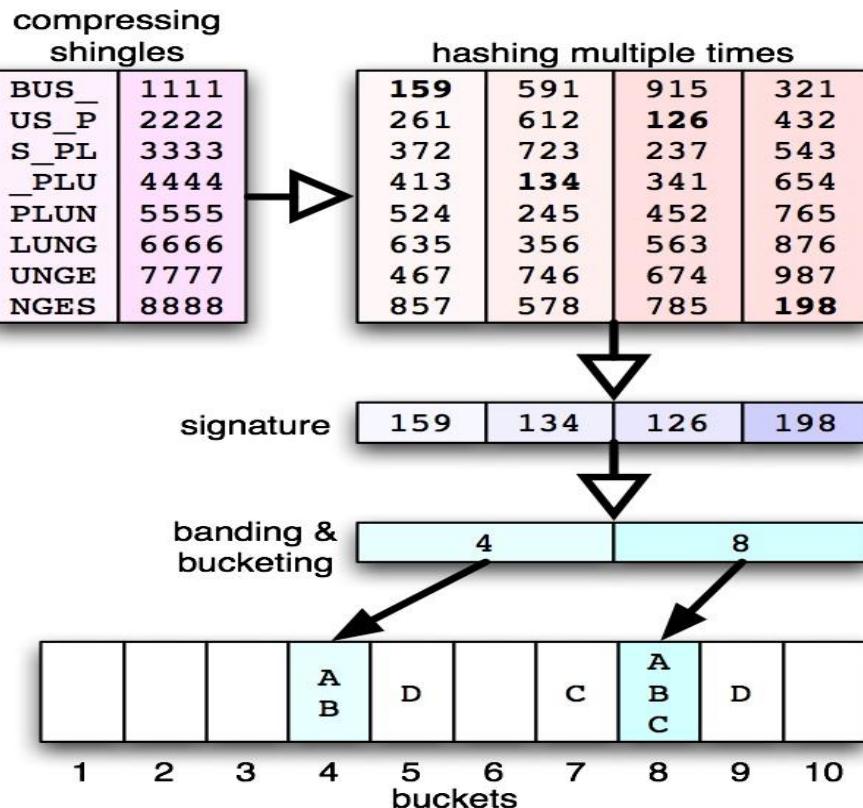
- Try to get minimum false negative and false positive rate i.e. get almost all document pairs with similar signatures, but eliminate most pairs that do not have similar signatures
- Check in main memory that *candidate pairs* really do have *similar signatures*

## The Actual Implementation ?

---

- It's simply **too slow to shuffle the bits.**
- Instead, people **simply rehash the shingle to get a new for it. (hash the hash.)** and then store **smallest one (from set of shingles)** in the first **signature slot.** Hashing is faster than permuting the bits
- For **each slot in the signature, apply a different hash function to set of shingles, and then take the smallest value store that into the signature slot.**
  - Each permutation is by a new hash function
- Typically **universal hash function** is used with separate parameters for each slot so that we get effectively a different hash

## The Actual Implementation ?



- First shingles are compressed/hashed
- Those hashed are hashed again. Note that lowest value is picked ( 159 for 1<sup>st</sup> hash corresponds to 1<sup>st</sup> signature slot)
- Done it four times with four hash functions ( one hash function per signature slot.)
- Then the banding is done

## Actual Implementation ?

- Key component in MinHash : we have a hash function which takes a 64-bit integer (shingle hash) and maps/hashes it to a different integer, with no collisions.
- $h(x) = ((ax+b) \bmod p) \bmod N$  given an input shingle integer  $x$  where  $a$  and  $b$  are randomly chosen integer less than maximum value of  $x$  and  $p$  is a prime number slightly bigger than max value of  $x$  and  $N$  is the maximum number we want to generate
- Essentially we have  $n$  different hash if the signature length is  $n$

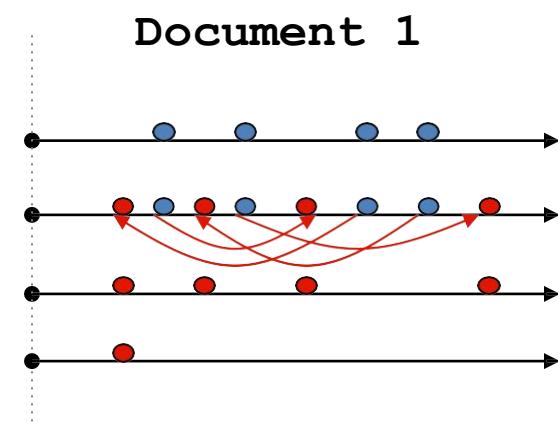
## Few points

---

- We have covered more than what is in the text for the following reasons
  - MinHash is an important technique to know specially when you are dealing with massive dataset like web and trying to cluster them – used in many domains
  - The text book is very cryptic i.e. does not explain banding completely though it is about detecting near duplicates
  - Text explains you the MinHash algorithm (permuting the Shingle bits) but that is never the way it is implemented !

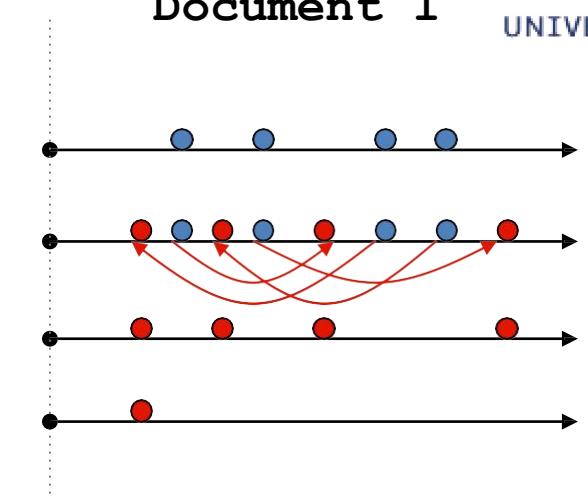
## Algorithm As Told in the Text -1

1. Take the hashed or compressed value of each of the shingles from set of shingles  $S(d_j)$  for  $j^{\text{th}}$  document  $d_j$
2. Shuffle the bits in the hashed shingle to get a new number  $\Pi(h)$  associated with a shingle and  $\Pi(h) \in \Pi(d_j)$
3. Take the smallest number corresponding to  $j^{\text{th}}$  document  $X_j^\Pi$  in  $\Pi(d_j)$  (minimum hash) and we store this in the first slot of the signature
  - Repeat the above step with another permutation  $\Pi$  till all slots of the signature are occupied
  - If the signature slot is defined as 200, then we have 200 permutations  $X_j^\Pi$  for  $i=1$  to 200
  - The text says that the sketch of the shingle is  $\Psi(d_i)$  is nothing but the set of  $X_j^\Pi$ . This is the prevalent definition of shingle sketch
  - Randomly picking up 200 shingles as a sketch is an optimization idea but that is not common in implementations for the sake of accuracy



## Algorithm As Told in the Text

4. Next compare  $X_j^{\sqcap}$  (smallest integer in  $\sqcap(d_j)$ ) for two documents  $d_1$  and  $d_2$  and if any two in a pair are same, we have candidate duplicates i.e.  $J(S(d_1), S(d_2)) = P(X_1^{\sqcap} = X_2^{\sqcap})$
- As the signature length is increased,  $P(X_1^{\sqcap} = X_2^{\sqcap})$  tends to be closer to the  $J(S(d_1), S(d_2))$
5. One can always do an exhaustive Jaccard Similarity to verify



## Summarizing Locality Sensitive Hashing

---

- *Same principle in different domains ( example – audio finger printing to detect song plagiarism) of LSH can be extended to many*
- *Without LSH it is a 2 step process*
  - Extract elaborate set of features and do a technique like PCA to render the high dimensional set to low dimension
  - Do clustering or nearest neighbor search
- *LSH addresses both by reducing to low dimension signature (min hashing) and then grouping near similar objects into same bucket with high probability*
- *Normal hashing tries to avoid collision but LSH tries to maximize it for near similar or similar items*
- *Naïve pair wise comparison is  $O(n^2)$ , standard KNN has  $O(nxm)$  where n is training examples and m is dimension. For  $n \gg m$ , it is  $O(n)$ . Using LSH, it can be sublinear achieved by reducing no of comparisons needed*

## References

- The best coverage of LSH is in the Stanford Course ( Mining Massive data set ) from where this presentation has primarily borrowed besides borrowing from few blogs.
- For the interested students, Professor Ullman's Lecture is the very best !
- The book material is actually more extensive (60+ pages), if you have time <http://infolab.stanford.edu/~ullman/mmds/ch3n.pdf>

Theoretical proof of the theorem is usually not discussed in these courses but it is rooted in probability theory and understanding of distributions.



Mining of Massive Datasets

Minhashing

- Jaccard Similarity Measure
- Constructing Signatures

Leskovec, Rajaraman, and Ullman  
Stanford University

25/18





**THANK YOU**

---

**Bhaskarjyoti Das**

Department of Computer Science Engineering



# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

## PageRank

---

**Bhaskarjyoti Das**

Department of Computer Science Engineering

## Order in Which we will Cover PageRank

---

- Recap Linear Algebra Basics necessary in this module
- Discuss Random Walk and define PageRank from that perspective
- Spectral Analysis to give an explanation of why PageRank is conceived the way it is !
- Discuss the Random Surfer Model as the underlying process, Basic and Scaled PageRank update rule and look at the equivalence of expression that we got from Spectral Analysis
  
- Topic Specific PageRank , PageRank Issues
- Finally we will compare with section 10 pages in section 21.2

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

## Recap Ergodic Markov Chain

**Bhaskarjyoti Das**

Department of Computer Science Engineering

1. Discrete Markov process is a series of experiments that are performed at a regular time interval and always have the same set of possible outcomes.
2. Process proceed in steps like a probability tree
3. These outcomes are called states.
4. Why discrete ? Because we are talking about discrete states of time i.e.  $t + \Delta t, t + 2 \Delta t, \dots$

1. Model can be in any **one of the states**
2. At **the next step**, the model can go to **same state or can move to another state**
3. Movement **between steps** is defined by probability
4. By **looking forward many steps** into the future, we can find probability of being in any one state !

## Stochastic Transition Probability Matrix

- *Transition Probability*  $a_{ij} = P(q_{t+1} = S_j \mid q_t = S_i)$
- Sum of  $a_{ij}$  for any column = 1
- A square matrix A is **stochastic** if all of its entries are non-negative and the **entries of each row here sum to 1 i.e. row stochastic matrix**

		Time $t + 1$				<b>Total</b>
Stat e		$S_1$	$S_2$	$S_3$	$S_4$	
<b>Time <math>t</math></b>	$S_1$	$a_{11}$	$a_{12}$	$a_{13}$	$a_{14}$	1
	$S_2$	$a_{21}$	$a_{22}$	$a_{23}$	$a_{24}$	1
	$S_3$	$a_{31}$	$a_{32}$	$a_{33}$	$a_{34}$	1
	$S_4$	$a_{41}$	$a_{42}$	$a_{43}$	$a_{44}$	1

## Eigen Decomposition Theorem and Right Eigen Vectors

---

- Eigen Decomposition Theorem: A square matrix can be decomposed into eigen vectors and eigen values.
- Let a square **MxM matrix C** and a **vector x that is not all zeros**, the **values of  $\lambda$**  satisfying  $Cx_R = \lambda_R x_R$  are the eigen values of C . The n **column vectors  $x_R$**  satisfying the above equation for n eigen value  $\lambda_R$  are called the **right eigen vectors** .
- The eigen vector corresponding to the eigen value of largest magnitude is the **Principal Right Eigen Vector**.
- In many common applications, only right eigenvectors (and not left eigenvectors) need be considered. Hence the unqualified term "eigenvector" can be understood to refer to a **right eigenvector**.

- A **left eigen vector** is defined as a **row vector** unlike the right eigen vector which is a column vector.
- Similarly we can have **Principal left eigen vector**.
- The **left eigen vectors** of C are the **M** vectors of y such that  $y^T C = \lambda y^T$   
In this case, y is transposed as it needs to be a row vector. We could have also written as  $X_L C = \lambda X_L$
- The **number of non zero eigen values of C** are at most rank of matrix C where Rank is the number of linearly independent rows or columns

Consider the matrix A

$$\begin{matrix} 30 & 0 & 0 \\ 0 & 20 & 0 \\ 0 & 0 & 1 \end{matrix}$$

The matrix has rank =3 and 3 eigen values 30, 20, 1

The corresponding eigen vectors are  $x_1, x_2, x_3$

$$\begin{matrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{matrix}$$

Consider an arbitrary vector  $V$

$$\begin{matrix} 2 \\ 4 \\ 6 \end{matrix}$$

It can be expressed as a linear combination of three eigen vectors  
i.e.  $2x_1 + 4x_2 + 6x_3$

- The probability for transitioning to a state will approach a limiting value as time goes to infinity.
  - These limiting values are called "**stable probabilities**".
  - The system will then be in a "**steady state**".
- Check: if I take a **probability vector** and multiply it by transition matrix and get out the same exact probability vector, it is a steady state.
  - A steady state is an **eigenvector for a stochastic matrix**.

## Ergodic Markov Chain

---

- A **Markov Chain is Ergodic** if there exists a positive integer  $T_0$  such that for all pairs of state  $i$  and  $j$ , if the chain is started at time  $t=0$  in state  $i$ , then for all  $t > T_0$ , probability of being in state  $j$  at time  $t$  is greater than 0.
- For Ergodicity in Markov chain, two technical conditions
  - Irreducibility : sequence of transitions of non-zero probability from any state to any other
  - Aperiodicity : states are not partitioned into sets such that all state transitions occur cyclically from one set to another.

## Steady State Theorem for Ergodic Markov Chain

---

- **Steady State Theorem** : For any ergodic Markov Chain, there is a unique steady state probability vector  $\Pi$  that is the **principal left eigen vector of transition Matrix P**
  - Because it is steady state, there is no further change in the probability vector i.e.  $\Pi P = \lambda \Pi$

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

## Random Walk and Page Rank

**Bhaskarjyoti Das**

Department of Computer Science Engineering

## Random Walk and Probability vectors

---

- Markov chains are abstractions of random walks.  
Transitioning to a state can be thought of as random walk.
- A probability (row) vector  $\mathbf{x} = (x_1, \dots x_n)$  tells us where the walk is at any point at what probability .  
Example :  $(000\dots\underset{i}{1}\dots000)$  means we're in state  $i$ .  
$$\begin{matrix} 1 & i & n \end{matrix}$$
- More generally, the vector  $\mathbf{x} = (x_1, \dots x_n)$  means the walk is in state  $i$  with probability  $x_i$
- And probability vector is stochastic 
$$\sum_{i=1}^n x_i = 1.$$

## Change in probability vector : Describing the Walk

---

- If the probability vector is  $x = (x_1, \dots, x_n)$  at this step, what is it at the next step?
- Recall that row  $i$  of the transition prob. Matrix  $P$  tells us where we go next from state  $i$ .
- So from  $x$ , our next state is distributed as  $xP$
- The one after that is  $xP^2$ , then  $xP^3$ , etc.
- (Where) Does this converge?
  - It converges at Steady State i.e.  $xP = x$

## Long Term Visit Rate for a Page

---

For any *ergodic* Markov chain, there is a unique long-term visit rate for each state that can be interpreted as *Steady-state probability distribution*

- Over a long time-period, we visit each state in proportion to this rate.
- It does not matter where we start.

- The steady state in vector notation is simply a vector  $p = (p_1, p_2, \dots, p_N)$  of probabilities.
- $p$  is the long-term visit rate (or PageRank) of page  $i$ .
- So we can think of PageRank as a very long vector – one entry per page.



**THANK YOU**

---

**Bhaskarjyoti Das**

Department of Computer Science Engineering



# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

## PageRank

---

**Bhaskarjyoti Das**

Department of Computer Science Engineering

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

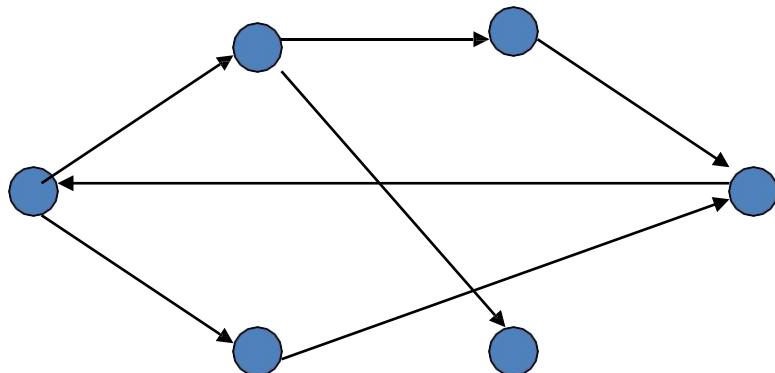
---

## PageRank Formula

**Bhaskarjyoti Das**

Department of Computer Science Engineering

- Web pages are organized in a network.
  - Each webpage is represented as a node.
  - Each hyperlink is a directed edge
  - The entire web can be viewed as a **directed graph**.

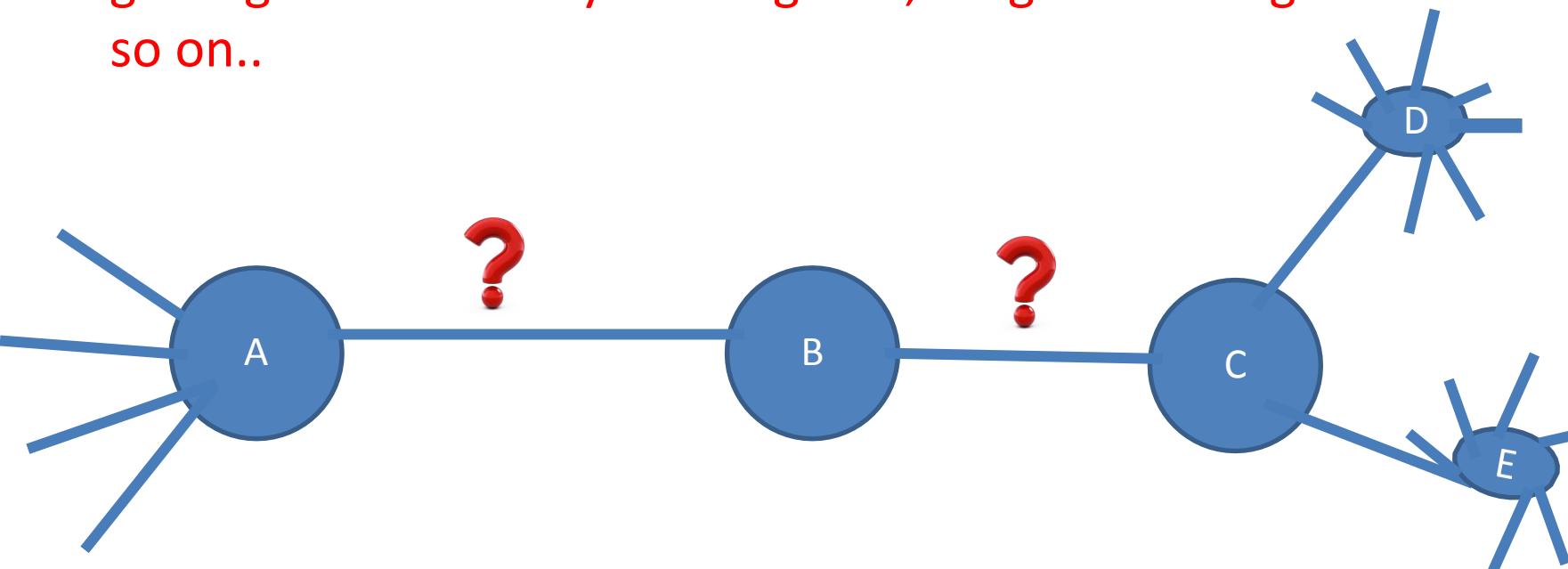


PageRank is a **numeric value that represents how important a page** is on the web.

- Webpage importance
  - One page links to another page is **a vote** for the other page
  - More votes = More important the page must be
- How can we model this importance?

## Concept of Eigen Vector Centrality For Undirected Graph

- Eigenvector centrality tries to generalize degree centrality by incorporating the importance of the neighbors (undirected)
- Which edge A-B or B-C should be severed to make B safe from Sexually Transmitted Disease (STD)?
- This is a recursive definition : depends on propensity of getting infected via your neighbor, neighbor's neighbor and so on..



## Why Eigen Vector Makes Sense

---

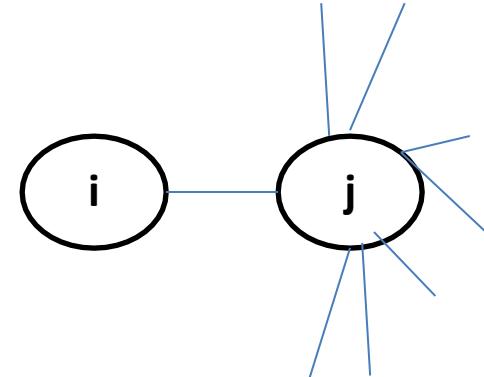
- Degree centrality **uses Only the number of neighbors -> all neighbors are equal**
- We want the centrality of  $v_i$  to be a function of its neighbors' centralities.
  - We posit that it is proportional to the summation of their centralities
  - Assume initially everyone has score  $x_i=1$  and we update its centrality using the sum of the scores/centralities of his neighbors

## Eigen Vectors Centrality by Power Method (by Hotelling )

- $X$  is the popularity vector of all nodes
  - Let  $X_j(0)$  be the popularity of the adjacent nodes  $j$  at time  $t=0$
  - Since popularity of a node depends on the popularity of nodes it is connected to , at time  $t=1$ , for node  $i$ ,
- $X_i(1) = \sum A_{ij} X_j(0)$  , summing over  $j$  nodes
- or  $X(1) = A \cdot X(0)$
- Continuing our recursive definition (using power method) after  $t$  such steps, we can say

$$X(t) = A \cdot X(t-1) = A^2 X(t-2) = A^3 X(t-3) = A^t \cdot X(0)$$

- $X(t) = A^t \cdot X(0)$
- As per iterative power method by Hottelling to calculate dominant eigen values , when  $t$  is very large, it will stabilize and will not change any further



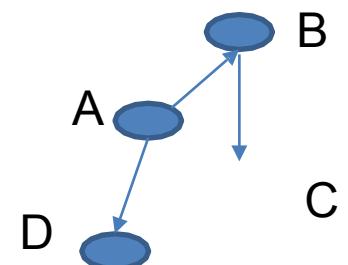
- $X(t) = A^t \cdot X(0)$
- Let  $X(0)$ , the vector, be a linear combination of the eigen vectors of  $A$  (adjacency matrix)
- $X(0) = \sum c_i v_i$  over  $i$  arbitrary eigenvectors of matrix  $A$
- Now plugging in,  $X_t(t) = A^t \sum c_i v_i$
- Now in equation  $Av = \lambda v$ 
  - Multiplying both sides by  $A$ ,  $A.Av = A.(\lambda v) = \lambda (AV) = \lambda^2 v$
  - Continuing for  $t$  times,  $A^t v = \lambda^t v$
  - Now after plugging in,  $X(t) = \sum \lambda_i^t c_i v_i$

- $X(t) = \sum \lambda_j^t c_i v_i$
- Let  $\lambda_1$  be the principal eigen value, we divide both sides of the equation by  $\lambda_1^t$
- $X(t) / \lambda_1^t = \sum_{i=1} (\lambda_1 / \lambda_i)^t c_i v_i$
- If  $t \rightarrow \infty$ , Lt  $(X_t(t) / \lambda_1^t) = c_1 v_1$  as the first eigen vector is largest and all other terms in the sum will become zero
- Hence, in this **recursive model, popularity value converges to (proportional to ) “principal eigen vector” of adjacency matrix !!**
- Note that, this exercise is done for an undirected graph and adjacency matrix is assumed to be symmetric

- Popularity for undirected graph can be **recast for popularity in a directed graph**
- In a directed graph (such as web), **popularity is passed on only by incoming edges**. Hence, popularity becomes zero **even though the node can have many outgoing edge connected to it if it does not have an income edge**
  - In diagram, A has popularity = 0 and since B has one incoming edge from A, B also has popularity or centrality 0 !! This may propagate through the network !
- To resolve this problem we **add initial popularity  $\beta$  to the centrality values for all nodes whereas alpha depends on the neighborhood**

$$\cdot X_i(1) = \alpha \sum_{j=1}^n A_{ji} X_j(0) + \beta$$

$$C_{Katz}(v_i) = \alpha \sum_{j=1} A_{j,i} C_{Katz}(v_j) + \beta.$$



## Katz Centrality For Directed Graph

$$C_{Katz}(v_i) = \alpha \sum_{j=1}^n A_{j,i} C_{Katz}(v_j) + \beta.$$

vector of all 1's

Rewriting equation in a vector form  
 with C as the centrality

$$\mathbf{C}_{Katz} = \alpha A^T \mathbf{C}_{Katz} + \beta \mathbf{1}$$

**Katz centrality:**

$$\mathbf{C}_{Katz} = \beta(\mathbf{I} - \alpha A^T)^{-1} \cdot \mathbf{1}.$$

For the matrix  $(\mathbf{I} - \alpha A^T)$  to be invertible, we must have

- $\det(\mathbf{I} - \alpha A^T) \neq 0$  and since matrix equation is hard to solve (inverse of matrix to be calculated), we can also go by the iterative equation and solve for steady state of

$$\mathbf{C}(t) = \alpha A \mathbf{C}(t-1) + \beta$$

- Issue : Everyone known by a well-known person is assumed to be well-known
  - To mitigate this problem we can divide the value of passed centrality by the number of outgoing links
- PageRank centrality is the correction introduced to Katz Centrality.
- Both Katz and PageRank Centrality are variants of Eigen Vector Centrality !!

$$C_p(v_i) = \alpha \sum_{j=1}^n A_{j,i} \frac{C_p(v_j)}{d_j^{out}} + \beta.$$



**THANK YOU**

---

**Bhaskarjyoti Das**

Department of Computer Science Engineering



# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

## PageRank

---

**Bhaskarjyoti Das**

Department of Computer Science Engineering

- $X(t) = A^t \cdot X(0)$  Using recursive definition of popularity . Essentially Eigen Vector Centrality
- Hotteling Power Method : **popularity value converges to ( proportional to ) “principal eigen vector” of adjacency matrix !!**
- To handle Directed Graph ( Katz Centrality):

$$\begin{aligned} X_i(1) &= \alpha \sum_{j=1}^n A_{j,i} X_j(0) + \beta \\ \text{▪ PageRank Correction : } C_p(v_i) &= \alpha \sum_{j=1}^n A_{j,i} \frac{C_p(v_j)}{d_j^{out}} + \beta. \end{aligned}$$

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

An Underlying process to Explain  
Page Rank Formula

Bhaskarjyoti Das  
Department of Computer Science Engineering

PageRank can be thought of as a **model of user behavior**.

1. Initially every web page is chosen uniformly at random
2. With probability  $\alpha$ , choose a hyperlink on page ( adjacent neighbor)
3. With probability  $1 - \alpha$ , perform **random walk** on web by randomly choosing a node and then restart the web surfing at step 1

## Basic Page Rank Update Rule

---

1. In a network of **n nodes**, we assign all nodes same initial PageRank  $1/n$
2. We choose **no of steps k**
3. We then perform **a sequence of k updates** as
  1. Each page divides its current PageRank equally across its outgoing links and passes that to the pages it points to
  2. Each page updates its PageRank to be sum of the shares that it receives
4. So, the PageRank **remains constant ( no scaling needed)** – neither created nor destroyed **but is just moved around**
5. As  $K \rightarrow \infty$  , the **PageRank of all nodes converge to limiting values** ( don't change any further)

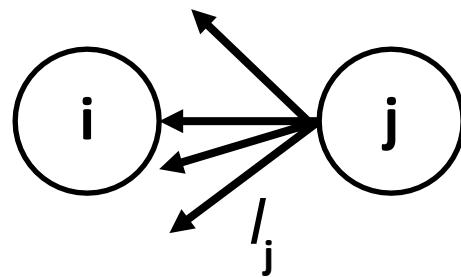
## Scaled Page Rank Update Rule

---

1. **First** apply basic PR update rule and **second** scale down all PR values by a factor of  $s$  when  $s < 1$
  2. It means that **total PR has gone down from 1 to  $s$**
  3. We then divide the **residual PR i.e.  $(1-s)$**  into  $n$  nodes as  $(1-s)/n$
- 
- It is like **water gets vaporized** ( leakage ?) and then it finally **comes back to the system as rain** (;-

## Intuitive Formulation Of PageRank Using Random Walk

1. If  $b_1, b_2$  etc denote the probability of the walk being at node 1,2 etc at a given step, what is the probability of the walk being at node  $i$  in next step ?
  1. For each node  $j$  that links to  $i$ , if the walk is currently at  $j$ , the chance of moving to  $i$  is  $1/l_j$  where  $l_j$  is the no of links out at  $j$
  2. Given **probability of walk** at  $j$  is  $b_j$ , the above **probability of reaching node  $i$**  is  $b_j(1 / l_j)$
  3. We can now sum the above over all  $j$  that link to  $i$
  4.  $b_i = \sum b_j / l_j$  i.e.  $N_{1i}b_1 + N_{2i}b_2 \dots$  etc
  5. This is using matrix based vector update Rule we write as  $\mathbf{b} = \mathbf{N}^T \mathbf{b}$



Scaled Version Of Random walk

---

- **Modified walk is** : for a number  $s > 0$ , with probability  $s$  , the walk follows a random edge as before and with probability  $(1-s)$  , it jumps to a node chosen randomly
- Probability of being at node  $i$  is **now the sum of two probabilities**  
$$b_i = \sum s b_j / l_j + (1-s)/n$$
 summed over all  $j$  that links to  $i$
- With the revised definition of matrix  $N$  as  $N$ , **we can say**  
$$b = N^T b$$

After  $k$  update,  $b_k = (N^T)^k b_0$  will converge to  $b^*$  so that  
$$N^T b^* = b^*$$

The  $b^*$  will be the eigen vector of  $N^T$  with corresponding eigen value 1

## Why Does the PageRank Converge?

---

**Perron Frobenious Theorem** states for any matrix M on which all entries are positive and it is a column stochastic matix,

- a) **largest eigenvalue  $c=1$**  such that  $c > |c'|$  for all other eigen values  $c'$
- b) **1 is an eigen value with multiplicity 1**
- c) For the eigen value 1: there exists a unique eigen vector  
**where the sum of the elements = 1**
- d) For any starting vector  $x \neq 0$ , the sequence vector  $M^kx$  converges to the direction converges to an eigenvector corresponding to the largest eigenvalue  $c = 1$  as  $k \rightarrow \infty$

- Modified walk is : for a number  $s > 0$ , with probability  $s$  , the walk follows a random edge as before and with probability  $(1-s)$  , it jumps to a node chosen randomly
- Probability of jumping to a random page will be small and is called the damping factor (typically 0.15)
- Probability of continuing random walk is higher i.e. typically it is :  
 $1 - 0.15 = 0.85$
- Since the surfer can jump to any page, that probability is typically  $1/n$

- **Rank Sinks or leak.** A rank sink occurs when a page does not have outgoing edges but has incoming edges. Rank sinks monopolize scores by refusing to share.
- **Hoarding or circular reference.** Extends the concept of rank sinks, a group of pages that only link between each other will also monopolize PageRank.
- **Dangling nodes:** it is a case of the random walk will not work.
- **Solution : Teleporting**
  - a random surfer- a proportion of the time he will be following links at random and a proportion of the time he will be 'teleporting' to a new random URL.

- $\text{PR}(A) = \alpha(\text{PR}(T_1)/C(T_1) + \dots + \text{PR}(T_n)/C(T_n)) + (1-\alpha)/n$ 
  - $\text{PR}(T_n)$  : That's "PR(T1)" for the first page in the web all the way up to "PR(T<sub>n</sub>)" for the last page with link to page A
  - $C(T_n)$  : The count, or number, of outgoing links for "C(T<sub>n</sub>)" for page n
  - $\text{PR}(T_n)/C(T_n)$  : Each page spreads its vote out evenly amongst all of its outgoing links.
  - $\alpha$  : All the incoming fractions of votes are added together but, this total vote is "damped down" by multiplying it by value such as 0.85
  - $(1 - \alpha)$  : The  $(1 - \alpha)$  bit at the beginning is a bit of probability math magic so the "*sum of all web pages PageRanks will be 1.*" It also means that if a page has no links to it, even then it will still get a small PR of 0.15

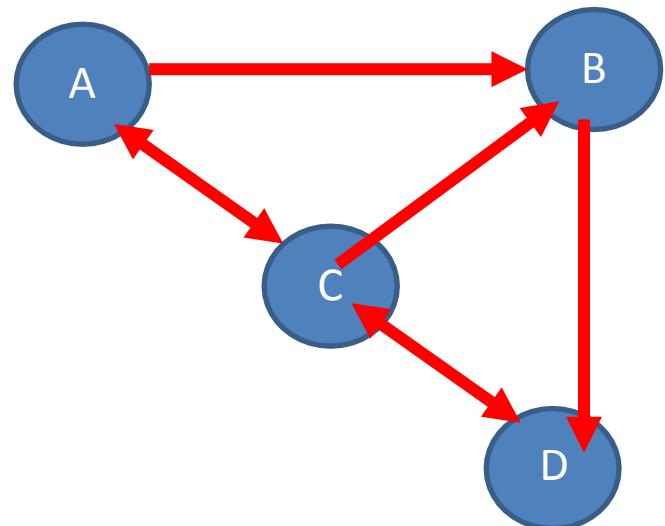
$$\text{PR}(A) = \alpha(\text{PR}(T_1)/C(T_1) + \dots + \text{PR}(T_n)/C(T_n)) + (1-\alpha)/n$$

$$C_p(v_i) = \alpha \sum_{j=1}^n A_{j,i} \frac{C_p(v_j)}{d_j^{out}} + \beta,$$

We are coming up with the same model !!

- Matrix M is enormous
  - For a sparse Google matrix, the  $1/n$  values in matrix B can be approximated with 1
- So you come across PageRank Problems or examples of 3 variants
- Basic Page Rank Update Rule
  - $PR(A) = PR(T_1)/C(T_1) + \dots + PR(T_n)/C(T_n)$
- Scaled Page Rank Update Rule **without Google Approximation**
  - $PR(A) = \alpha(PR(T_1)/C(T_1) + \dots + PR(T_n)/C(T_n)) + (1-\alpha)/n$
- Scaled Page Rank Update Rule **with Google Approximation**
  - $PR(A) = \alpha(PR(T_1)/C(T_1) + \dots + PR(T_n)/C(T_n)) + (1-\alpha)$

## PageRank Scoring Using Basic PageRank Update Rule



	Iteration 0	Iteration 1	Iteration 2	Rank
A	$\frac{1}{4}$	$1/12$	$1.5/12 = 0.125$	4
B	$\frac{1}{4}$	$2.5/12$	$2/12 = 0.166$	3
C	$\frac{1}{4}$	$4.5/12$	$4.5/12 = 0.375$	1
D	$\frac{1}{4}$	$4/12$	$4/12 = 0.333$	2

**Iteration 1 (damping not considered):**

$$\text{PR}(A) = \text{PR}(C) / \text{Out}(C) = \frac{1}{4} / 3 = 1/12$$

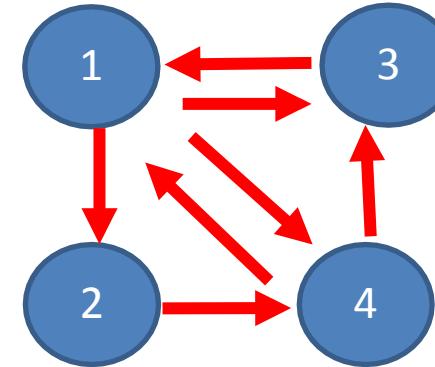
$$\begin{aligned} \text{PR}(B) &= \text{PR}(A) / \text{out}(A) + \text{PR}(C) / \text{Out}(C) \\ &= \frac{1}{4} / 2 + \frac{1}{4} / 3 = 2.5 / 12 \end{aligned}$$

## PageRank Scoring Using Scaled PageRank Update Rule

Given : damping factor  $\alpha = 0.5$  , initial rank =  $\frac{1}{4} = 0.25$

Use Google Approximations i.e.

No need to divide  $(1 - \alpha)$  by number of nodes !



If  $C(n)$  = no of out links of  $n$  ,  $C(1)=3$ ,  $C(2) = 2$ ,  $C(3) = 1$ ,  $C(4) = 2$

**Iteration 0 :**  $PR(1)=PR(2)=PR(3) = PR(4) = 0.25$

**Iteration 1 (with damping):**

$$\begin{aligned} PR(1) &= (1-0.5) + 0.5 * (PR(3) / C(3) + PR(4) / C(4)) \\ &= 0.5 + 0.5 * (0.25/1 + 0.25/2) = 0.6875 \end{aligned}$$

$$PR(2) = 0.541, PR(3) = 0.667, PR(4) = 0.604$$

**Iteration 2 :** We plug in these found PageRank and continue...

At some iteration, we will reach steady state ...



**THANK YOU**

---

**Bhaskarjyoti Das**

Department of Computer Science Engineering



# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

## PageRank

---

**Bhaskarjyoti Das**

Department of Computer Science Engineering

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

## PageRank As a Final Measure

**Bhaskarjyoti Das**

Department of Computer Science Engineering

## Topic Specific Pagerank

---

- One can envision **topic specific pagerank** for science, fiction, religion etc. For users interested in such a topic, only such a specific pagerank distribution can be invoked. Can be done in a case when the search engine knows the user's interests !
- **User having a mixture of interests** : if this can be approximated as a linear combination of interests i.e. 60% sports and 40% religion for example,
  - While teleporting , determine randomly whether to teleport to a set S of sports pages 60% of the time and set of religion pages 40% of the time
  - Once the above decision is made, choose a page in set S uniformly at random . The personalized Pagerank in this case is :  $0.6 \Pi_s + 0.4 \Pi_p$  when  $\Pi_s$  and  $\Pi_p$  are steady state distributions
  - This leads to an Ergodic Markov Chain with a steady state distribution personalized to users interests

- Real surfers are not random surfers.
  - Examples of nonrandom surfing: back button, short vs. long paths, bookmarks, directories – and search!
  - Markov model is not a good model of surfing.
  - But it's good enough as a model for our purposes.

- Simple PageRank ranking (as described on previous slide) produces bad results for many pages.
  - Consider the query [video service].
  - The Yahoo home page (i) has a very high PageRank and (ii) contains both *video* and *service*.
  - If we rank all Boolean hits according to PageRank, then the Yahoo home page would be top-ranked.
  - Clearly not desirable.
- In practice: rank according to weighted combination of raw text match, anchor text match, PageRank & other factors.

## How important is PageRank?

---

- Frequent claim: PageRank is the most important component of web ranking.
- The reality:
  - There are several components that are at least as important: e.g., anchor text, phrases, proximity, tiered indexes ...
  - Rumor has it that PageRank in his original form (as presented here) now has a negligible impact on ranking!
  - However, variants of a page's PageRank are still an essential part of ranking.
  - Addressing link spam is difficult and crucial.

## References and the Order in Which we Covered PageRank

---

- Text book covers the followings concepts ( and is cryptic)
  - Section 21.2.1 Ergodic Markov Chain, steady state theorem
  - Does not have the spectral analysis such as eigen vector based on eigen vector that tells you why PageRank is designed this way
  - Section 21.2.2
    - It starts by telling that the **PageRank matrix** is the probability for **Random Walk with Teleporting** without the explanation
    - Random surfer model but does not have the analysis of scaled PageRank update rule based on Random Surfer Model
  - Section 21.2.3 Personalized and topic specific page rank
- Reference used : chapter 14 ( Networks Crowds and Market : Reasoning about a Highly Connected World) by Jon Kleinberg & David Easley

## References and the Order in Which we will Cover PageRank

---

- Section 21.2.1
  - Markov Chain, stochastic matrix, probability vector in a Markov Chain, Random surfer as a Markov Chain.
  - It then discusses the **transition probability matrix with teleporting or damping** without explaining the reasons
  - Ergodic Markov Chain and steady state theorem
  - Using the above theorem, Random walk results in a unique distribution of Steady State probability of the state in a induced Markov Chain
- Does not have the spectral analysis that tells you why PageRank is using eigen vector
- Does not have the explanation of how Random Surfer model correlates to Page Rank with teleportation formula

## References and the Order in Which we will Cover PageRank

---

- Section 21.2.2
  - Essentially uses power method to calculate left eigen vector and gives a long example of how it reaches steady state with successive iterations
  - Section 21.2.3 Personalized and topic specific page rank
- Reference used : chapter 14 ( Networks Crowds and Market : Reasoning about a Highly Connected World) by Jon Kleinberg & David Easley
- Section 21.2 – 10 pages and cryptic
- So, we have followed a slightly different sequence and little bit more details to develop the understanding



**THANK YOU**

---

**Bhaskarjyoti Das**

Department of Computer Science Engineering



# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

## Hyperlink Induced Topic Search (HITS)

---

**Bhaskarjyoti Das**

Department of Computer Science Engineering

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

## Understanding HITS Algorithm

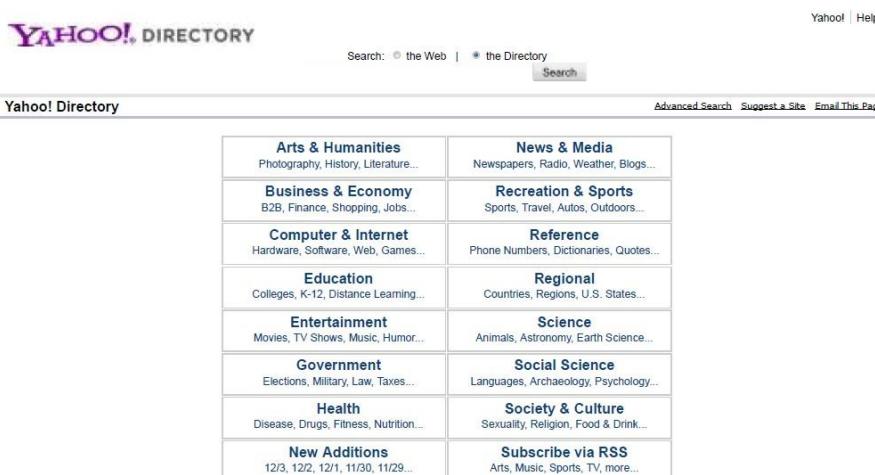
**Bhaskarjyoti Das**

Department of Computer Science Engineering

## How to Organize the Web

---

- First Try : Human Curated Web : Yahoo!
- Second Try: Web Search : Find relevant documents in web that is huge
- Web has shifted the Information Retrieval question from a problem of scarcity to a problem of abundance
- Which few web pages the search Engine should recommend ?



The screenshot shows the Yahoo! Directory homepage. At the top, there's a search bar with options for "the Web" or "the Directory" and a "Search" button. Below the search bar, there are links for "Advanced Search", "Suggest a Site", and "Email This Page". The main content area is titled "Yahoo! Directory" and features a grid of category boxes. Each box contains a title and a brief description. The categories are:

Arts & Humanities Photography, History, Literature...	News & Media Newspapers, Radio, Weather, Blogs...
Business & Economy B2B, Finance, Shopping, Jobs...	Recreation & Sports Sports, Travel, Autos, Outdoors...
Computer & Internet Hardware, Software, Web, Games...	Reference Phone Numbers, Dictionaries, Quotes...
Education Colleges, K-12, Distance Learning...	Regional Countries, Regions, U.S. States...
Entertainment Movies, TV Shows, Music, Humor...	Science Animals, Astronomy, Earth Science...
Government Elections, Military, Law, Taxes...	Social Science Languages, Archaeology, Psychology...
Health Disease, Drugs, Fitness, Nutrition...	Society & Culture Sexuality, Religion, Food & Drink...
New Additions 12/3, 12/2, 12/1, 11/30, 11/29...	Subscribe via RSS Arts, Music, Sports, TV, more...

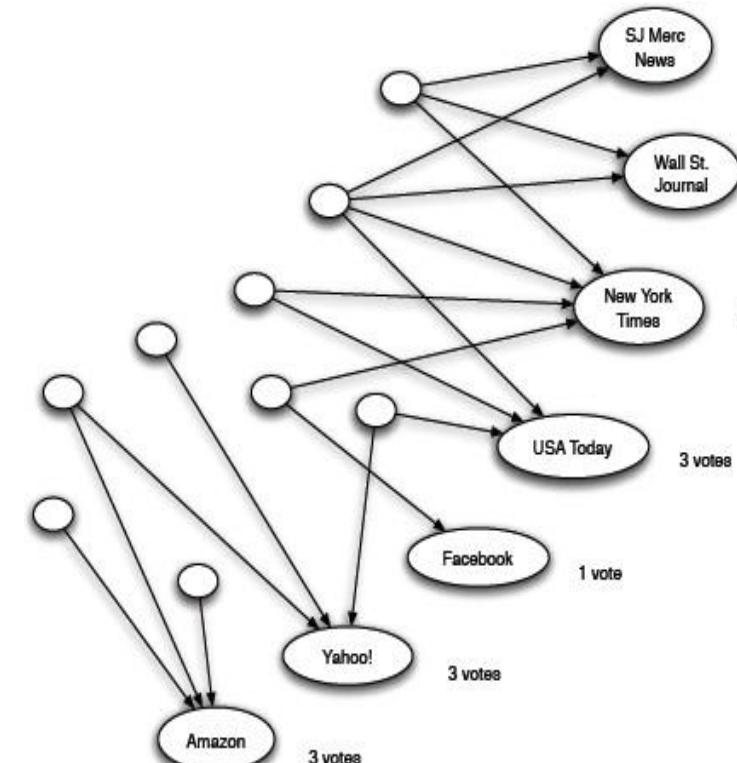
## Voting by In-links

---

- Suppose somebody does a single word query “Cornell” and [www.cornell.edu](http://www.cornell.edu) should be the right answer
- Search Engine does not have much context to assess the intent of all links that the search engine may offer as query result
- One way to assess the possible links is to look at a bunch of web pages that is mentioning the word “Cornell” and we find maximum number of links from those pages pointing to [www.cornell.edu](http://www.cornell.edu) !
  1. Collect a large sample of web pages that relevant to the query by a text only retrieval system
  2. Let these pages vote through their links
  3. Find out the page on the web that receives largest number of in-links from the large sample of pages we collected for a query : “Cornell”

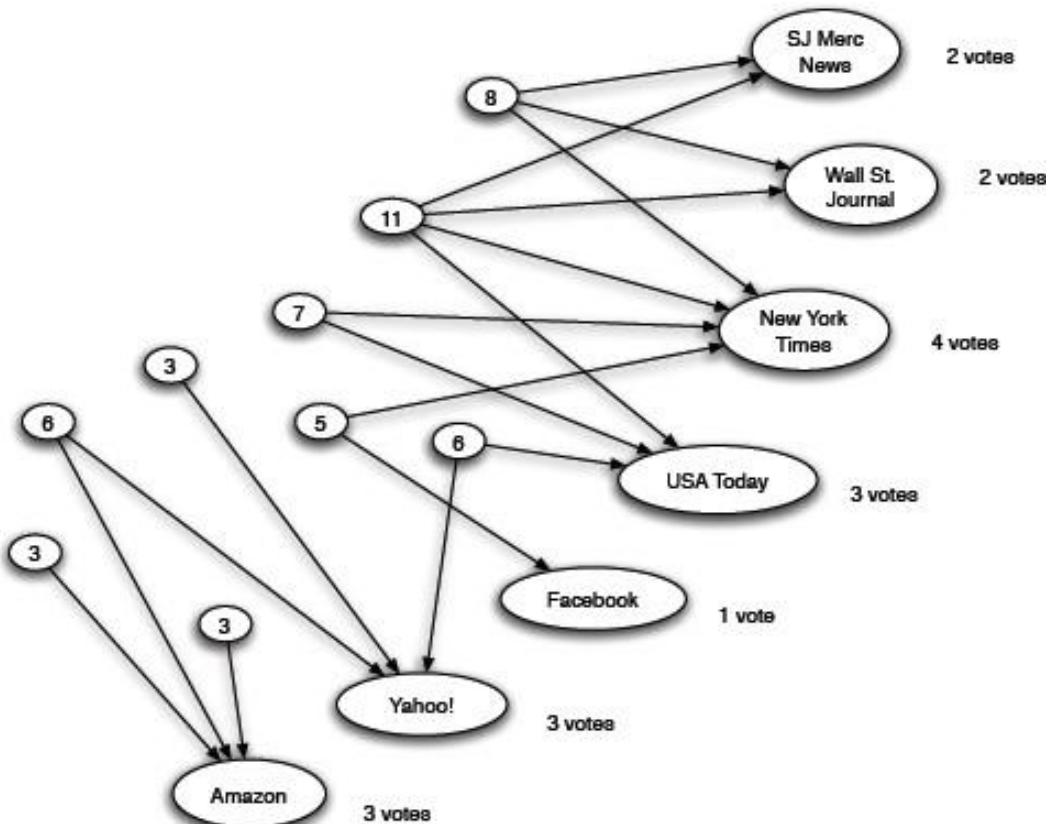
## Counting in-links to Pages For Query “Newspaper”

- In response to query “newspaper” a set of pages were collected
  - From these pages, we found links mostly to newspaper websites and few non-newspaper sites
  - Few sample pages voted for many of the pages that received lot of votes.
  - These sample pages have a good sense of what are the newspaper sites and are good candidates for list
  - A page’s value as list is equal to the sum of the votes received by all pages for which it voted !!



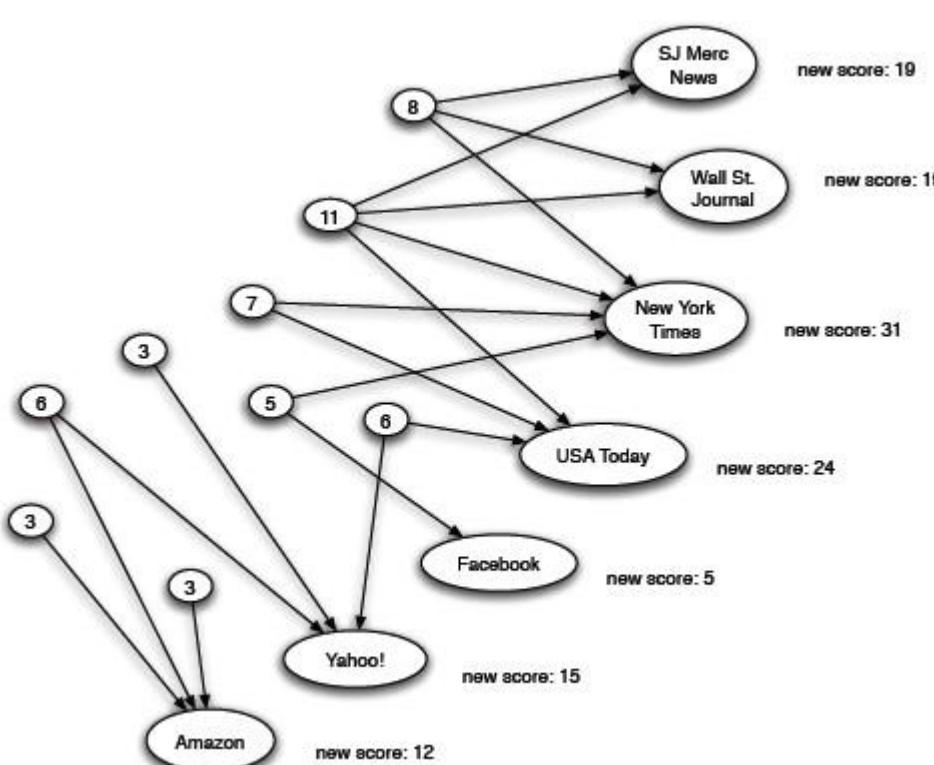
## Finding Good List For Query “Newspaper”

- Finding good lists ..



Each Labeled Page's New Score is Sum of Values of all Lists that Point to it

- If we believe that the pages scoring well as lists have a better sense of where the good results are , then we should **weigh their votes more heavily !**
- Give each page's vote a weight equal to its value as list
- **Principle of repeated improvement** : each refinement on one side enables a further refinement on the other side



## Hubs and Authorities

---

Interesting pages fall into two classes:

- **Authorities** are pages containing useful information (the prominent, highly endorsed answers to the queries)
  - Newspaper home pages
  - Course home pages
  - Home pages of auto manufacturers
- **Hubs** are pages that link to authorities (highly value lists)
  - List of newspapers
  - Course bulletin
  - List of US auto manufacturers

## Principle Of Repeated Improvement

---

Recompute the score of the **hubs** using the improved scores of the **authorities**!

Repeat .. Recompute the score of the **authorities** using the improved scores of the **hubs**!

Circular definition - will turn this into an iterative computation.

- ✓ A good hub links to many good authorities
- ✓ A good authority is linked from many good hubs

## Each Page has Two Scores

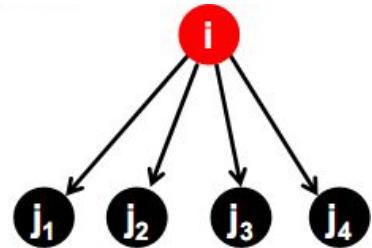
---

Each page  $p$ , has two scores

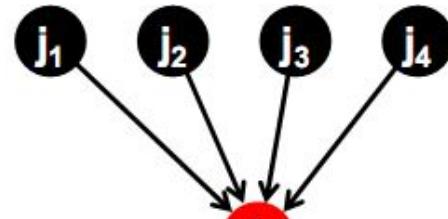
- A *hub score* ( $h$ ) quality as an expert    Total sum of votes of pages pointed to
- An *authority score* ( $a$ ) quality as content  
                    Total sum of votes of experts

## Iterative Update

---



$$h_i = \sum_{j \rightarrow i} a_j$$



$$a_i = \sum_{j \rightarrow i} h_j$$

**Authority Update Rule:** For each page  $i$ , update  $a(i)$  to be the sum of the hub scores of all pages that point to it.

**Hub Update Rule:** For each page  $i$ , update  $h(i)$  to be the sum of the authority scores of all pages that it points to.

## The HITS Algorithm

---

- Start with all hub scores and all authority scores *equal to 1*. Choose a number of *steps k*.
  - Perform a sequence of *k* hub-authority updates.
  - For each node:
    - First, apply the *Authority Update Rule* to the current set of scores.
    - Then, apply the *Hub Update Rule* to the resulting set of scores.
  - At the end, hub and authority scores may be very large.

Normalize: typically weights are normalized so that the squared sum for each type of weight is 1    i.e.  $\sum H_i^2 = \sum A_i^2 = 1$

## Scaling Factor ?

---

- To prevent the  $h()$  and  $a()$  values from getting too big, can scale down after each iteration.
- Scaling factor doesn't really matter ( for example, MAX\_NORM also can be done) :
- we only care about the *relative* values of the scores.

## How many iterations?

---

- Claim: relative values of scores will converge after a few iterations:
  - in fact, suitably scaled,  $h()$  and  $a()$  scores settle into a steady state!
  - proof of this comes later in spectral analysis.
- In practice, ~5 iterations get you close to stability.



**THANK YOU**

---

**Bhaskarjyoti Das**

Department of Computer Science Engineering



# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

## Hyperlink Induced Topic Search (HITS)

---

**Bhaskarjyoti Das**

Department of Computer Science Engineering

# ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

---

## HITS as an Eigen Vector Problem

Bhaskarjyoti Das

Department of Computer Science Engineering

## The HITS Algorithm

---

- Initialize all weights to 1.
- Repeat until convergence
  - **O** operation : hubs collect the weight of the authorities

$$h_i = \sum_{j:i \rightarrow j} a_j$$

– **I** operation: authorities collect the weight of the hubs

$$a_i = \sum_{j:j \rightarrow i} h_j$$

– Normalize weights under some norm

## HITS Implementation

---

- Executed in two steps
  - Sampling Step : to collect a set of relevant web pages given a topic
  - Iterative Step : To find the hubs and authorities using the information collected during sampling

The algorithm involves carrying out the sampling step by querying a search engine and then using the most highly ranked web pages retrieved for determining hubs and authorities

## HITS as Eigenvector Problem

---

- HITS algorithm in matrix notation. Iterate:
  - Compute  $h = Aa$
  - Compute  $a = A^T h$
- By substitution we get:  $h = AA^T h$  and  $a = A^T A a$
- Thus,  $h$  is an eigenvector of  $AA^T$  and  $a$  is an eigenvector of  $A^T A$ .
- So the HITS algorithm is actually a special case of the power method and hub and authority scores are eigenvector values.
- HITS and PageRank both formalize link analysis as eigenvector problems.

## Issues

---

1. **Hubs and authorities** : A clear cut distinction may not be appropriate always as some nodes are both hubs and authorities
2. **Topic Drift** : Search on “Jaguars” drifted to a football team called “Jaguars”. These documents in fact **may not be very relevant to query being posted.**
3. Auto generated links represent **no human judgements** but HITS still gives them **equal importance**

## PageRank vs HITS

---

### •PageRank

- Relatively simplistic view of the web based on the concept of popularity !  
Hubs and Authority takes a structured view of the web
- HITS came in 1997 by Kleinberg whereas PageRank came in 1998
- PageRank can be thought of as concentrating on one half of HITS i.e. the authority pages
- In practice, HITS is used on a result set ( of a query) but not on all pages in the web like PageRank
- Unlike PageRank, HITS renormalizes the score after every iteration as otherwise the score will grow too big !



**THANK YOU**

---

**Bhaskarjyoti Das**

Department of Computer Science Engineering