CS4611 Tuesday 17th September 2013

Information Retrieval.

Prof. M.P. Schellekens (TA: Ang Gao)
Slides: P. Nayak and P. Raghavan.

Lecture 1: Boolean Retrieval

◇ Information Retrieval (IR) is finding material (usually documents) of an unstructed nature (usually text) that satisfies an information need from within large collections (usually stored on computers)

◇ Corpora: large collection of documents.

Ranking: Two ways = System & Ask the user if it was helpfull.

◇ Precision: fraction or proportion of documents that came back are useful to me.

◇ Recall: Fraction of relevant docs in collection that are retrieved

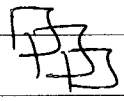
Terms for now we can refer to them as words in a document. This is expanded as the words are processed and we will end up with terms.

Variable-Size Posting lists

Dictionary ⇒ collection of all terms (processed words)

Sort by docID!

Documents to be indexed



Words (large caps)

Token Stream

Tokenizer

Modified Tokens

Linguistic Modules

friend roman countrymen (small caps) + remove phrase.

Indexer

friend ⇒ [2] → [4]

Inverted Index

etc

CS4611 Thursday 19th September 2015

Term frequency is handy for ranking and grouping results.

- Book: Introduction to Information Retrieval (http://informationretrieval.org/)

Lecture 2: The term vocabulary and postings lists.

Postings list: Document ID's.

◊ A term is a (normalized) word type, which is an entry in our IR system dictionary.

Normalization

C.A.T. → CAT → cat

So when you type C.A.T. into ~~set~~ IR search engine 'cat' will be at the top instead of Caterpillar. We need to ensure that C.A.T. shows up at the top.

Synonyms & Homonyms.

- Left (past of leave) } one word / two meanings
- Left (opposite of right) }

△ Porter Algorithm
- Used for stemming English
eg compress, compression & compressed
→ compress, compress & compress

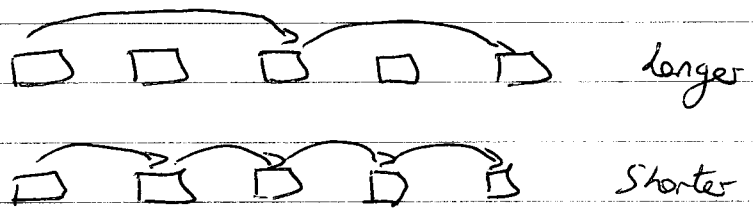
sses → ss
ies → i
ational → ate
tional → tion

~~Rules~~ ⇒
Other ⇒ Lovins Stemmer (approx 250 rules)

Rules are application & language specific

MIT.english
mit.german

Placing Skips = & use √L to evenly spaced skip pointers



Phrase Queries & Positional Indexes.

Search → Stanford University
Query → "I went to college at Stanford"

These are not a match.

Biwords

'friends romans' and 'romans countrymen'

each of the above biwords are not dictionary terms.

CS4611 Tuesday 24th September 2013

Dictionary (Dictionaries and tolerant Retrieval)

- Hashtables
- Trees

Hashtables — no prefix search → tolerant retrieval is a disadvantage

Tree → Binary Tree
→ B-Tree

mon* starts with mon
*mon ends with mon
to find if mon is in the middle then do an intersection of the two.

◇ Permuterm Index : permutation of a term
eg hello
hello$, ello$h, llohe, lohel
o$hell (where $ is a symbol)

Bigram - little bit similar to biterms
Store two consecutive letters
example
April
a, ap, ri, il, l

Spelling Correction

- Correcting the doc(s) being indexed
- Correcting user queries to retrieve "right" answers

Will not catch for example from / form
but if it is part of a phrase "I came form school"
clearly it should be "I came from school"
Google → Did you mean ...

- Lexicon

Alternatives
- Levenshtein Distance (Edit distance)
- Weighted Edit Distance
- n-gram overlap

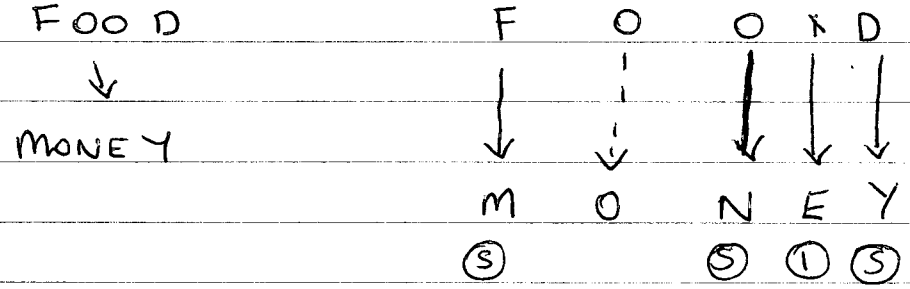
- Note : Must learn Levenshtein Distance algorithm

CS4611 Tuesday 1st October 2013

Edit Distance (Levensthein)

- Is the minimum number of letter insertions, letter deletions and letter substitutions to transform one word into another

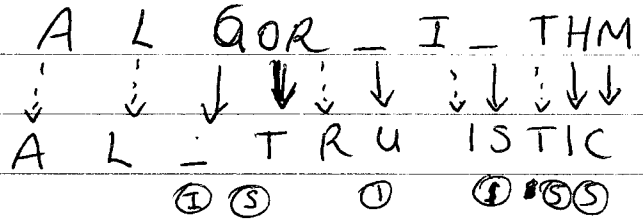
Example :- The distance between FOOD and MONEY



∴ The edit distance between MONEY and FOOD is 4

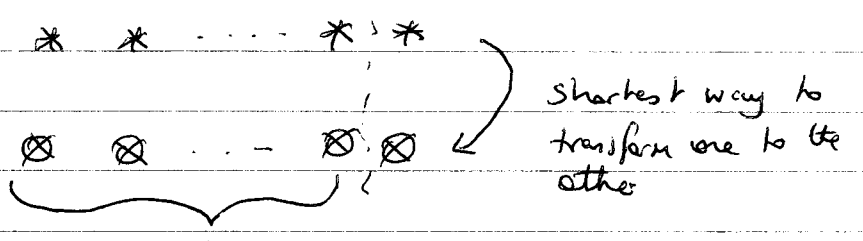
Edit Distance (MONEY, FOOD) ≤ 4

Example :-



6 operations - EditDistance = 6

Observation :-



↳ still shortest transformation ie a subsection. If you find the shortest way then it cannot

Recursive Definition of the Edit Distance between Two strings of A [1 ... m] and B [1 ... n]

Denoted by Edit (A [1 ... m], B [1 ... n])

If neither string is empty (ϕ)

then there are 3 possibilities for the last column, in the shortest edit sequence we are adding up.

3 Possibilities

- Insertion
- Substitution
- Deletion.

△ Insertion

The last entry in the top row is empty

A [1] ... A [m-1] ϕ) Insertion

B [1] ... B [n-1] B(n)

The edit distance = Edit ((A [1 ... m], B [1 ... n-1]) +

△ Deletion

The last entry in bottom row is empty.

A [1] ... A [m-1] A [M]) Deletion.

B [1] ... B [n-1] ϕ

The edit distance = EDIT ((A [1] ... A [M-1]) (B [n-1] B[n])

△ Substitution

Both rows have characters in the last column

If the characters are the same substitution is free (0 cost)

So the edit distance = edit ⟶

$\longrightarrow$ $Edit(A[1 \cdots m-1], B[1 \cdots n-1])$

If characters are different
$Edit(A[1 \cdots m-1], B[1 \cdots n-1]) + 1$

BASE-CASES

$EDIT(A[1 \cdots m], \phi) = m$ (Deletions)

$EDIT(\phi, B[1 \cdots n]) = n$ (insertions)

Simplify Notation

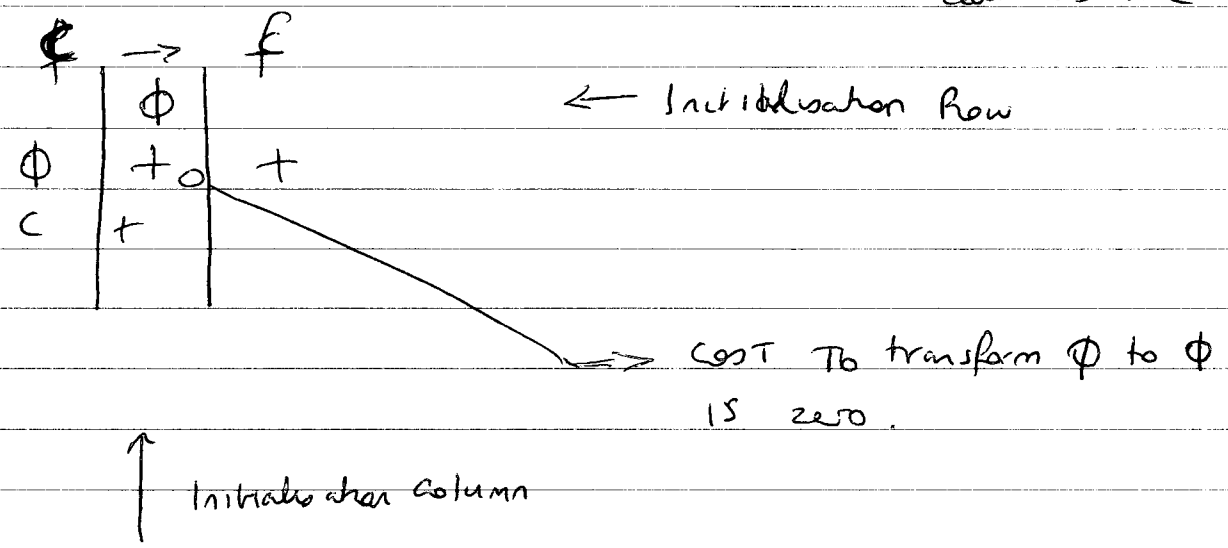
Recursive subproblems are always prefixes of the strings A and B, so we can simply use the length of the substrings.

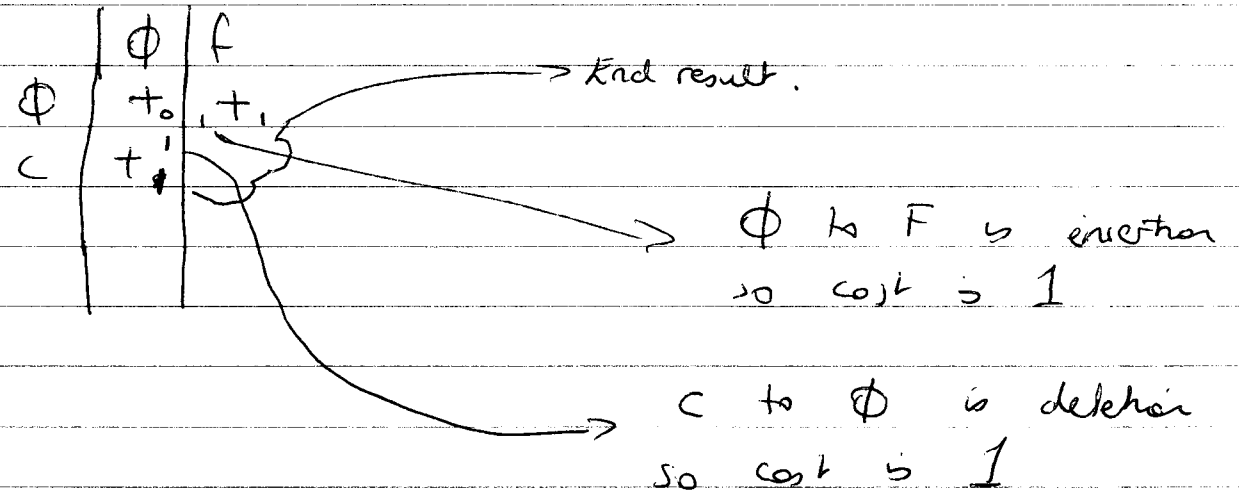
$EDIT(i, j)$ (shorthand for $edit(A[1 \cdots i], B[1 \cdots j])$

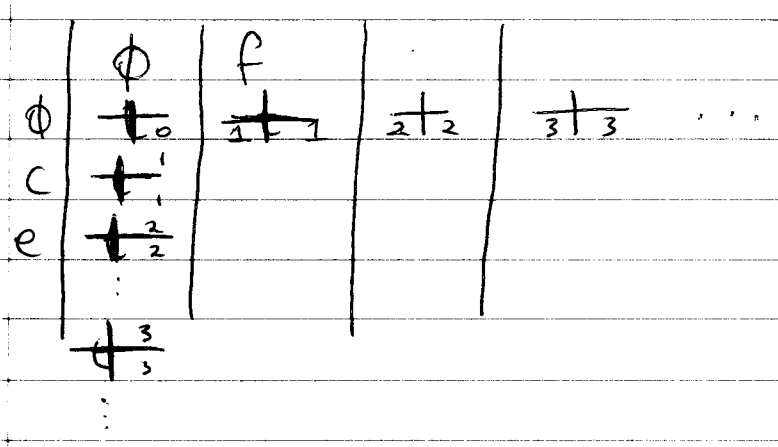
$$EDIT(i,j) = \begin{cases} i & \text{if } j = 0 \\ j & \text{if } i = 0 \\ \min \begin{cases} EDIT(i-1, j) + 1 \\ EDIT(i, j-1) + 1 \\ EDIT(i-1, j-1) + [A[i] \neq B[j]] \end{cases} \end{cases}$$

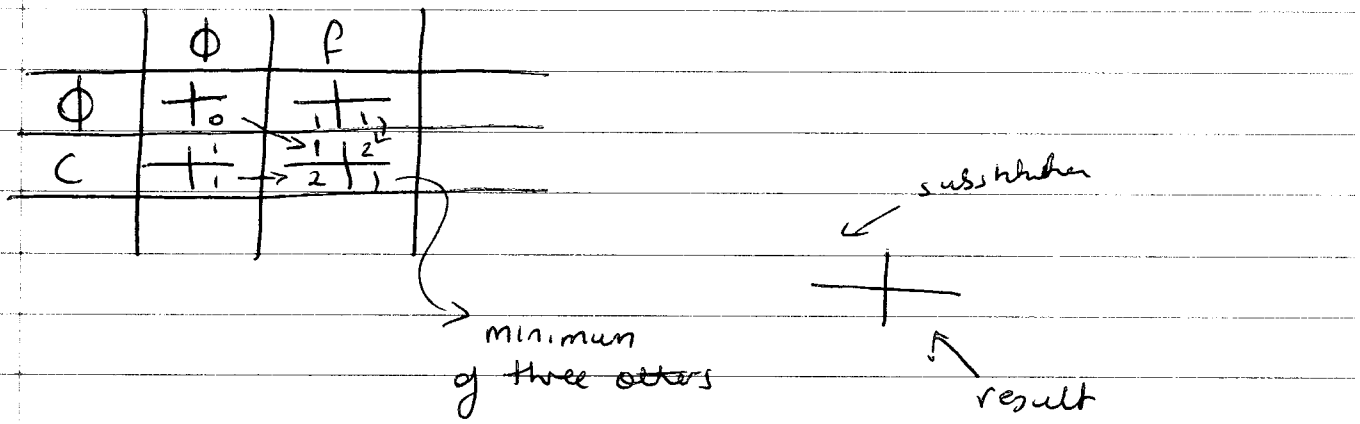
↳ is 1 if characters are different 0

SET UP A TABLE (Dynamic Programming) to compute the edit distance

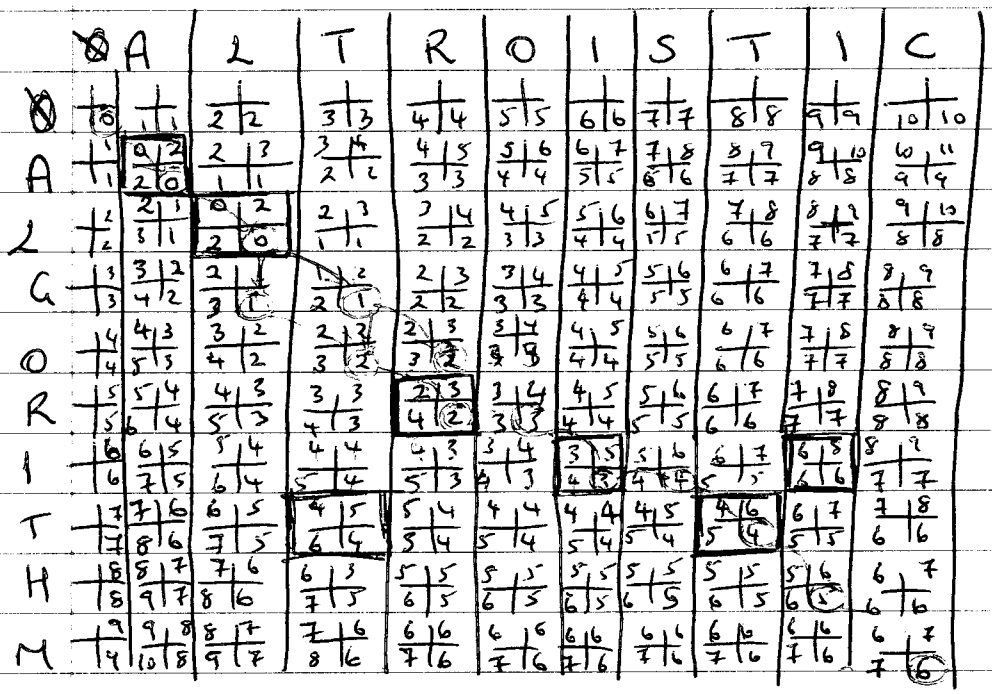







Φ
f
Φ
C
minimum
of three others
substitution
result

CS4611 Tuesday 8th October 2013



6 Operations

2. Distance

- There are multiple paths that could have been taken.

= where both letters are the same, this is where we have changes.

Assignment 1 (DANGER → BALTIMORE)

- Compute the Edit Distance
- Find all shortest Edit Sequences that transform "DANGER" to "BALTIMORE".

Assignment Due @ 15th October 2013 10am

One Option - Jaccard Coefficient

$$X \cap Y \,/\, X \cup Y$$

Count amount from intersection and divide by count of union.

Soundex

Class of heuristics to expand a query into phonetic equivalents

Convert similar sounding letters to numbers.

Compression

- Heap's Law is linear in log-log space
  It is the simplest possible relationship between collection size and vocab size in log-log space
  Empirical Law

CS4611 Thursday 10th October 2013

◇ Heaps Law : $M = kT^b$

M = No. of terms

T = No. of tokens

$\log_{10} M =$

$\log_{10}(kT^b)$

$\log_{10}(k) + \log(10)(T^b)$

✸ ~~$\frac{b}{0.5}\log_{10}(T) + \frac{\log_{10} k}{C}$~~

$$\frac{b}{0.5}\log_{10}(T) + \frac{\log_{10} k}{C}$$

b + k are based on experiments

▲ Zipfs Law

Frequent Terms - short Code

Infrequent Terms - long Code

Few frequent terms and many rare terms.

HEAPS LAW & ZIPFS LAW

Dictionary Compression

- Average length of a term is 8 characters (english)

- Entropy - enables one to compute the compressibility of data (kind of pre-scanning to see if it can be compressed)
  - ▲ Huffman Code : text file replace each char. with binary number, end result is one list of binary numbers

---

Huffmans Code

Characters A, ... F
Num. of Char 100,000

Freq.	A	B	C	D	E	F
	45	13	12	16	9	5

Fixed 000 001 010 011 100 101

Variable 0 101 100 111 1101 1100

CASE A → 300,000 (100,000 x 3 bits)
CASE A → ~~45~~,000 (45,000 x 1 bit)

CABEB (Encoding size) 224,000 bits (45*1 + 13*3 + 12*3 + 16*3 + 9*4 + 5*4 = 224)

Huffmans Code – gives the shortest code to the character that appears most frequently.

Two Codes

0 1 0 1 1 0 1 0

Prefix

one is a prefix of the other.

If you concatenate which do you decode? 0 1 0 1 1 0 1 0

Non-prefix code

A	B	C
1	00	001

⇓

1, 00, 00, 1

We want the Huffman Code to be leading, compare coding to entropy.

CS4611 Tuesday 15th October 2013

Entropy : A basic introduction

1) (See Slides)

Entropy enables one to compute ... (see slides)

Huffman Encoding Example

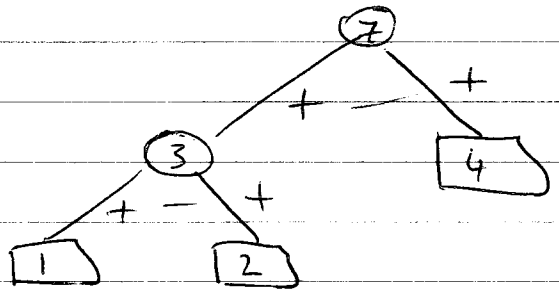
- Fair Dice

$p = 1/6$

$1/p = 1/(1/6) = 6$

$\log(1/p) = \log(6) = 2.59$

HUFFMANN CODES



Once at the top of the tree, this = the final cost of the encoding.

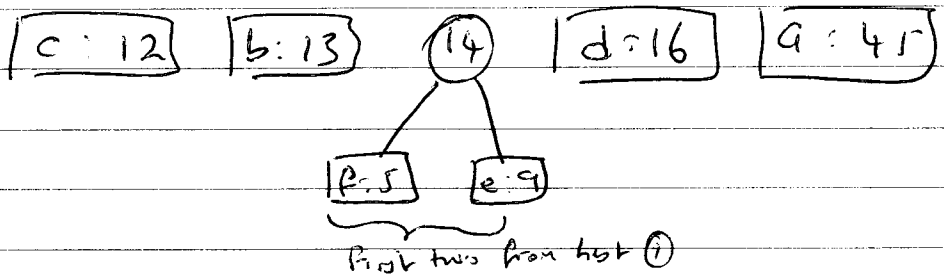
Sort the characters by increasing freq.

f : 5	e : 9	c : 12	b : 13	d : 16	a : 45

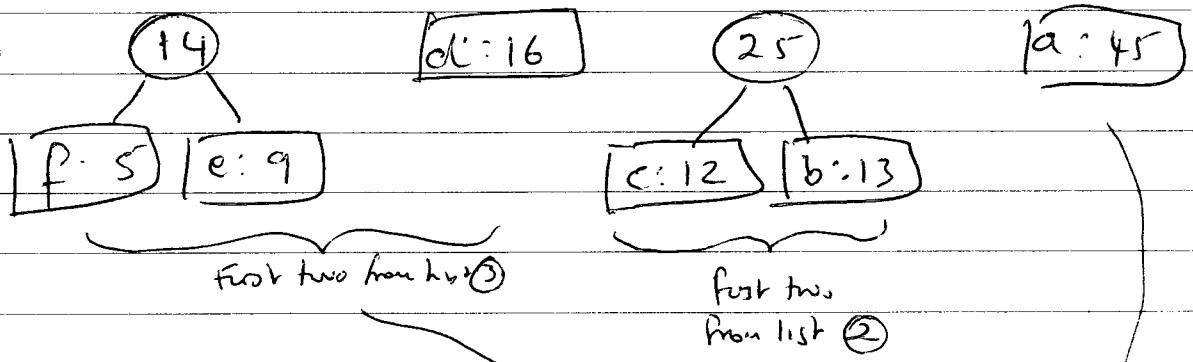
In order

12 → 13 → 14 → 16 → 45

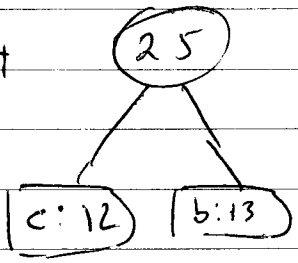
2 :



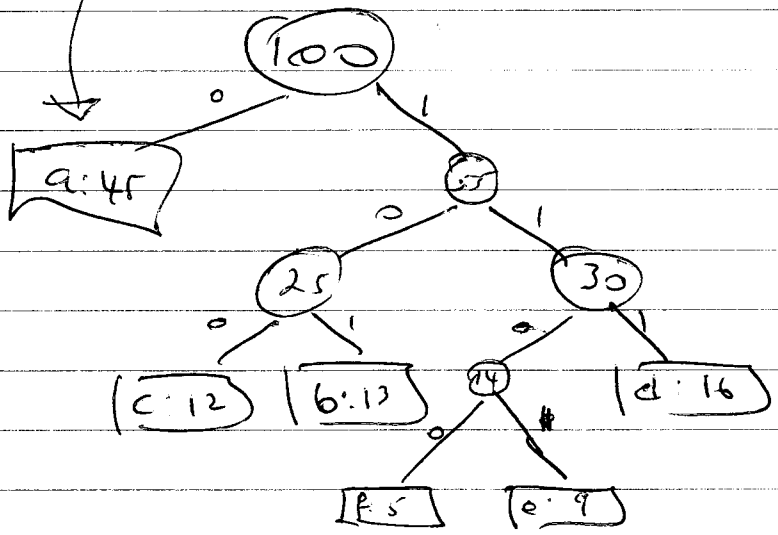
3



4



0 left
1 right



How would you get num. of bits (AVG BIT LENGTH)

1 → Compute total cost 224,000 bits (100 + 55 + 25 + 30 + 1
/ by number of character occurrences (100,000) (100)
= 2.24

$$\Rightarrow \frac{224,000}{100,000} = \boxed{2.24 \text{ bits}}$$

OR (longer way)

2 → $\left(\frac{45}{100} \times 1\right) + \left(\frac{12}{100} \times 3\right) + \left(\frac{13}{100} \times 3\right) + \left(\frac{5}{100} \times 4\right) + \left(\frac{9}{100} \times 4\right)$

$+ \left(\frac{16}{100} \times 3\right) = 224 / 100 = 2.24$

Compute the Entropy of the original file:

a	b	c	d	e	f
45	13	12	16	9	5

o Work in Log(2) (log base 2)

→ $\frac{45}{100}$ $\frac{13}{100}$ $\frac{12}{100}$ $\frac{16}{100}$ $\frac{9}{100}$ $\frac{5}{100}$

$H(p_1 \dots p_n) = p_1 \log(1/p1) + p_2 \log(1/p2) + \dots + p_n \log(1/pN)$

The above is the Entropy Expression
In our case $n = 6$ as there are only 6 characters.

$\log_2(k) = \ln(k) / \ln(2)$

(where "ln" is "log in base e")
This is if you do not have $\log_2$ on your calculator.

2.22

a) Compute the huffman encoding
b) Compute the cost of the encoding

a) (prefix) codes for char
a 0, b: 101, c 100, d: 111, e 1101, f 1100

b) $45 \times 1 + 13 * 3 + 12 \times 3 + 16 \times 3 + 9 * 4 + 5 * 4 = 224,000$ bits

c) $224,000 / 100,000 = 2.24$ average encoding length

e) Entropy

$H(45/100, 13/100, 12/100, 16/100, 9/100, 5/100) =$

$45/100 \log(45/100) - 13/100 \log(13/100)$
$- 12/100 \log(12/100) - 16/100 \log(16/100$
$- 9/100 \log(9/100) - 5/100 \log(5/100)$

$= 2.23$ Answer.

f) Conclusion
Excellent prediction of avg binary encoding length (some minor round off).

CS4611 Thursday 17th October 2013

- Calculator required for exam!!!

Postings Compression

Postings are much larger than Dictionary Compression. (approx. times 10)

Focus on storing GAPS instead of docID, for example THE

283043 283043 283044

The distance between THE occuring is quite frequent so the GAP is one so we store the first occurance plus 1.

- Dedicate one bit (high bit) to be a continuation bit c.

eg. 0 = Continue, 1 = Halt

Binary code of 824 is 1100111000

one indicates done if it was a zero it would mean there is more.

Other Way

Instead of bytes we can use a different "unit of alignment": 32 bit (words), 16 bits.

△ Gamma Codes - More compression — bit level compression.

* We need unary code to be able to use gamma codes
Unary code for 3 ... (see slides)

- First encode it in binary
- Then chop off the leading bit
  13 → 1101 → 101 = offset
- Length is length of offset
- 13 → 3
- Encode length in unary code: 1110
- Gamma = 1110101

→ Concatenation of length and offset.

1110101 = 111 0 101
length cont. offset.

★ Exercise

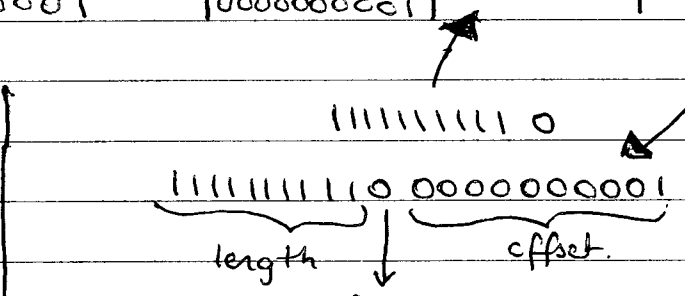
Gaps	ϕ
1	
2	
3	
4	
9	
13	
24	
511	
1025	

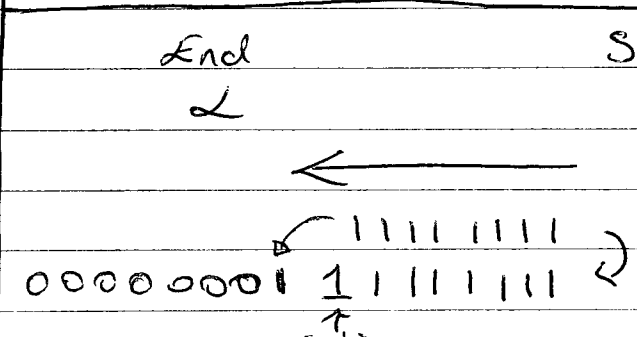
GAPS	BINARY
1	0001
2	0010
3	0011
4	0100
9	1001
13	1110
24	1111 1
511	
1025	

1	10000001
2	10000010
3	10000011
4	10000100
9	10001001
13	10001101
24	10010000
511	00000001 11111111

GAPS	BINARY	offset	length	γ CODE
1	1	∅	0	0
2	10	∅	10	100
3	11	1	10	101
4	100	00	110	11000
9	1001	001	1110	1110001
13	1101	101	1110	1110101
24	11000	1000	11110	11110 1000
511	111111111	11111111	111111110	111111110 11111111
1025	10000000001	0000000001		

GAPS	VB Code
1	10000001
2	10000010
3	10000011
4	10000100
9	10001001
13	10001101
24	10011000
511	00000011 11111111
1025	





130 = (2 to the power of 7) plus 2

Length of Gamma Code
- length of offset = $\lfloor \log_2 a \rfloor$ bits
- length of length = $\lfloor \log_2 a \rfloor + 1$ bits

RANKING

Scoring

Rank the documents by attaching a score to them.
Assign a score to each query-document (eg 0, 1)
The more frequently the term occurs the higher the score.

- Jaccard Coefficient

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

$(A \neq 0)\ (B \neq 0)$

Query "Ides of March"
Doc "Ceasar died in March"
Jaccard = 1/6

March = (1)
7 words but March repeated so (6)

CS4611 Tuesday 22nd October 2014

How to rank search results.

- Term Frequency
- Tf-idf Ranking
- Vector Space Model

Boolean retrieval – All or nothing, a massive amount of results or no results.
- Good for applications
- Law Collection of documents

Query – relevance → attach a number (eg 0,1)
score ↑

Take 1 – Jaccard Coefficient, can help us compute a score.
We don't measure how frequent a term occurs in a document using the Jaccard Coefficient - only that a term occurs in the document.

Vector ⇒ 

2D 3D

△ $w_{t,d} = \begin{cases} 1 + \log_{10} tf_{t,d} & \text{if } tf_{t,d} > 0 \\ 0 & \text{otherwise} \end{cases}$

$tf_{t,d} \rightarrow w_{t,d}$: 0 → 0, 1 → 1, 2 → 1.3, 10 → 2, 1000 → 4 etc.

Exercise

Compute the Jaccard matching score and the tf matching score for the following query-doc. pairs (1. Jaccard coefficient, 2. term frequency rate)

1 [information on cars] D1 : "all you've ever wanted to know about cars"

2 ["] D2 : "information on trucks, information on planes, information on trains".

3 [~~———~~] [red cars and red trucks] D3 : "cops stop red cars more often"

△ tf-matching-score $(q,d) = \sum_{t \in q \cap d} (1 + \log tf_{t,d})$

Ans $J_1 = \frac{1}{11} = \frac{1}{11}$

$J_2 = \frac{2}{6} = \frac{1}{3}$

$J_3 = \frac{2}{8} = \frac{1}{4}$

TF - Matching - Score (Q, D)

TF - Matching - score (Q1, D1) = 1
TF - " - " (Q2, D2) = 2.954
" - " - " (Q3, D3) = 3

---

tf-idf

tf = term frequency

We want high weights for Rare terms (Arachnocentric) (example)
For terms like (Good, Increase, line) we want positive weights

We are going to use document frequency for this.

idf - weight Definition = $idf_t = \log_{10} \frac{N}{df_1}$

t = term

We measure the informativeness of a term to the entire collection.

example : $idf_t = \log_{10} \frac{1000,000}{df_t}$

term	df_t	idf_t
animal	100	4

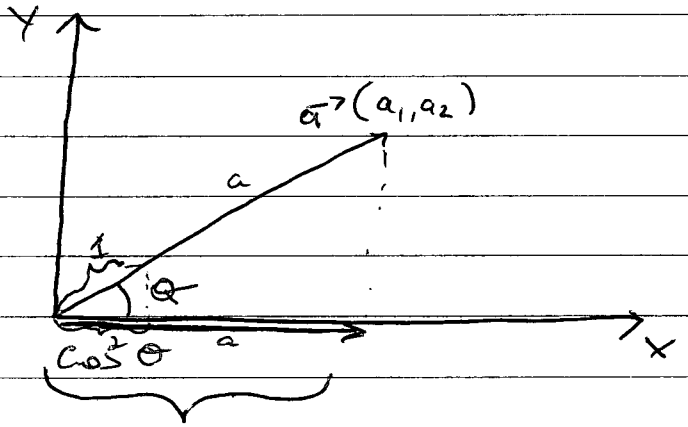
Word	collection freq.	document freq
Insurance	10440	3997
try	10422	8760

We don't measure collection freq as the count could be the same but the document freq could be extremely different.

tf-idf $w_{t,d} = (1 + \log tf_{t,d}) \cdot \log \frac{N}{df_t}$

Product of tf weight and idf weight

CS4611 Thursday 24th October 2013



Vector Spaces

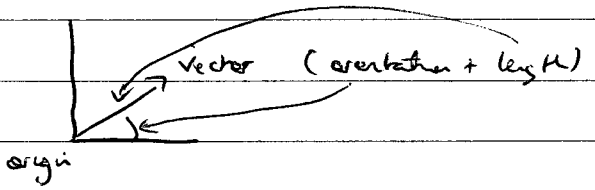
θ = theta

1 x a

2 x a

$\|\vec{a}\| \cos\theta$

$\Rightarrow$ length of Projection of $\vec{a}$ on x-axis

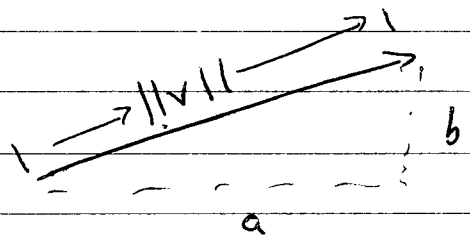


$$u \cdot v = u_1 v_1 + u_2 v_2 + u_3 v_3$$

$$u \cdot v = 1.5 + 2.6 + 3.7 = 38$$

Length of a vector $\Rightarrow$

$$\|v\| = \sqrt{a^2 + b^2} = \sqrt{v \cdot v}$$



Distance between two vectors

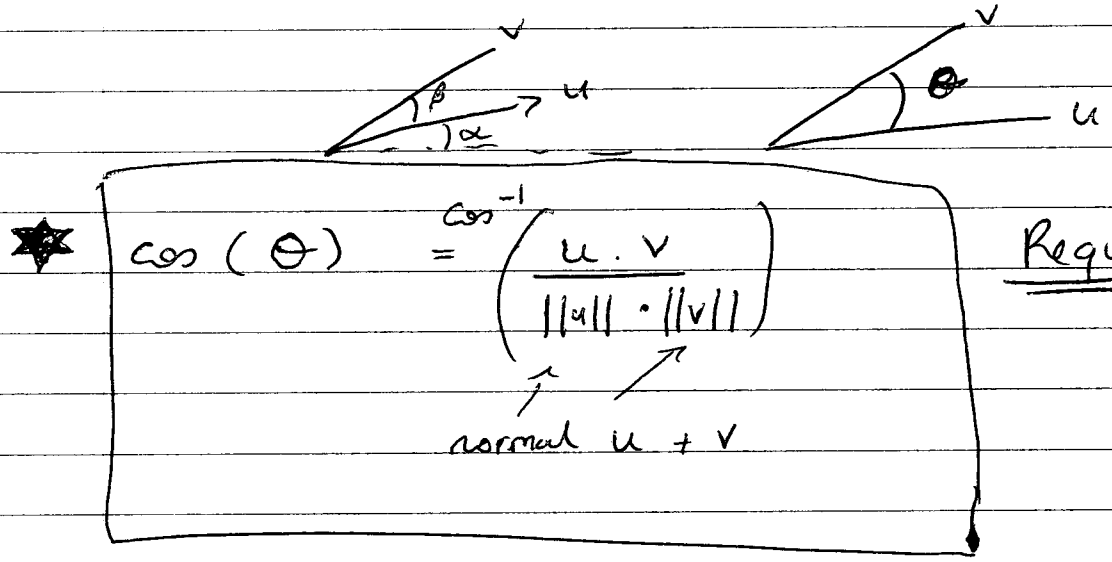
$$\text{Dist}(P_1, P_2) = \sqrt{V \cdot V} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (\alpha_1 - \alpha_2)}$$

α

Angel between two vectors

$u = (\|u\| \cos(\alpha), \|u\| \sin(\alpha))$ and
$v = (\|v\| \cos(\beta), \|v\| \sin(\beta))$

$$u \cdot v = \|u\| \, \|v\| \cos(\beta - \alpha)$$



$$\cos(\theta) = \cos^{-1}\left(\frac{u \cdot v}{\|u\| \cdot \|v\|}\right)$$

Required

normal $u + v$

Example:
Find the angle in degrees between the vectors
$u = (-2, 2, 1)$ and $v = (2, 3, 6)$

Solution

$$||u|| = \sqrt{4+4+1} = 3$$

$$||v|| = \sqrt{4+9+36} = 7$$

$3 \cdot 7 = 21$

$(-2 \cdot -2)\ (2 \cdot 2)\ (1 \cdot 1)$
$(2 \cdot 2)\ 3 \cdot 3)\ (6 \cdot 6)$

-2	2	1
2	3	6

$$\cos(\theta) = \frac{u \cdot v}{||u|| \cdot ||v||}$$

$$u \cdot v = -2 \cdot 2 + 2 \cdot 3 + 1 \cdot 6 = 8$$

$$\cos(\theta) = \frac{8}{21} = 67.6073^\circ$$

proximity = similarity

Distance is a bad idea as the distance could be very far apart, alternatively use the angle to measure the distance between two vectors.

Use angle instead of distance

example: d + d' are quite similar but using the Euclidean distance, the distance is quite far away. The angle is close. in fact angle = 0.

△ The closer things are together the bigger the cosine is going to be.

Length Normalization

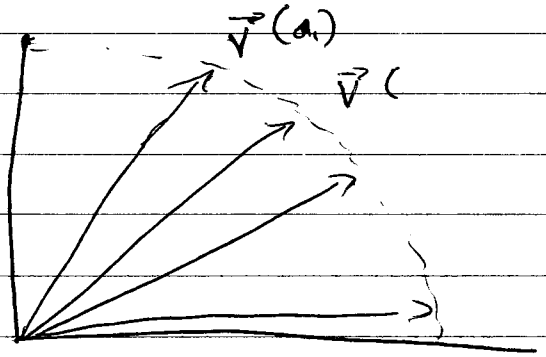
$$||x||_2 = \sqrt{\sum_i x_i^2}$$

This maps vectors onto the unit sphere

Query Vector Definition

$$\cos(\vec{q}, \vec{d}) = sim(\vec{q}, \vec{d}) = \frac{\vec{q} \cdot \vec{d}}{|\vec{q}||\vec{d}|} =$$

$$\frac{\sum_{i=1}^{|V|} q_i d_i}{\text{(see slides)}}$$



CS 4611 Thursday 31st October 2013

TERMS	Q	D_1	D_2	D_3	idf	Q tf idf	D_1 tf idf	D_2 tf idf	D_3 tf idf
A	0	1	1	1	0	0	0	0	0
Arrived	0	0	1	1	0.176	0	0	0.176	0.176
Damaged	0	1	0	0	0.477	0	0.477	0	0
Delivery	0	0	1	0	0.477	0	0	0.477	0
Fire	0	1	0	0	0.477	0	0.477	0	0
Gold	1	1	0	1	0.176	0.176	0.176	0	0.176
in	0	1	1	1	0	0	0	0	0
of	0	1	1	1	0	0	0	0	0
Silver	1	0	1	0	0.477	0.477	0	0.954	0
Shipment	0	1	0	1	0.176	0	0.176	0	0.176
Truck	1	0	1	1	0.176	0.176	0	0.176	0.176

$$Sim(Q, D_1) = \frac{\vec{Q} \cdot \vec{D}}{\|Q\| \cdot \|D\|}$$

$Sim(Q, D_2)$ = "

$Sim(Q, D_3)$ = "

Page Ranking

D_1 – Shipment of Gold Damaged in a fire

D_2 – Delivery of Silver arrived in a Silver truck

D_3 – Shipment of Gold arrived in a truck.

Q Gold Silver Truck.

$\rightarrow$ VECTOR (query)

$Q_* = (0, 0, 0, 0, 0, 0.176, 0, 0, 0.477, 0, 0.1761)$

Get this from table – idf column top → bottom.

Compute Similarity Between :

Q and D_1
Q and D_2
Q and D_3
} Largest Similarity best document Match with Q.

$$Sim(Q, D_i) = \frac{\sum_{j=1}^{3} w_{Q_1} w_{i,j}}{\sqrt{\sum_{j=1}^{3} w_{Q_j}^2} \sqrt{\sum_{j=1}^{3} w_{i,j}^2}}$$

$i = 1, 2, 3$

Vector (query) $\rightarrow ||Q|| = \sqrt{\quad}$

- Compute

(A) $\rightarrow ||Q|| = \sqrt{0^2 + 0^2 + 0^2 + 0^2 + 0^2 + (0.1761)^2 + 0^2 + 0^2 + (0.47$
$+ 0^2 + (0-1761)^2 =$

$\rightarrow ||\vec{D}_1||$
$||\vec{D}_2||$
$||\vec{D}_3||$
} same as $||Q||$ - take column and x^2 then $\sqrt[2]{x}$ the result of additions to get final result.

$\rightarrow ||Q|| = \sqrt{0.03101121 + 0.2276244 1 + 0.03101121}$

$= \sqrt{0.2896}$

$= \boxed{0.5382}$ (Calculator $\Rightarrow \sqrt[2]{x}$

$\rightarrow ||\vec{D}_1|| \sqrt{0.5173} = \boxed{0.7192}$

$\rightarrow ||\vec{D}_2|| = \sqrt{1.2001} = \boxed{1.0955}$

$\rightarrow ||\vec{D}_3|| = \sqrt{0.1240} = \boxed{0.3522}$

(B) $\vec{Q} \cdot \vec{D}_1$ = Could only one left, where ever you have a zero it disappear

so $(0.1761) \cdot (0.1761) = \boxed{0.0310}$

$\vec{Q} \cdot \vec{D}_2 = (0.4771) \cdot (0.9542) =$

$(0.1761) \cdot (0.1761) =$

$\vec{Q} \cdot \vec{D}_3 =$

$(0.4771 \times 0.9542) + (0.1761)^2$

$= \boxed{0.4862}$

$\vec{Q} \cdot \vec{D}_3 = (0.1761 \times 0.1761) + (0.1761 \times 0.1761)$

$= \boxed{0.062}$

(C) $\text{Sim}(\vec{Q}, \vec{D}_1) = \dfrac{\vec{Q} \cdot \vec{D}}{\|\vec{Q}\| \cdot \|\vec{D}_1\|} = \dfrac{0.0310}{0.5382 \times 0.7192}$

$= \boxed{0.0801}$

(B from above — How? — A . A — Q . D_1 on previous page)

$\text{Sim}(\vec{Q} \cdot \vec{D}_1) = \dfrac{\vec{Q} \cdot \vec{D}_2}{\|\vec{Q}\| \cdot \|D_2\|} = \dfrac{0.4862}{}$

$= \boxed{0.824}$

$\text{Sim}(\vec{Q} \cdot \vec{D}_2)$ ---> $\boxed{0.3271}$

From © which one will be presented?

$\vec{D}_1$ 0.0801
$\vec{D}_2$ 0.8246
$\vec{D}_3$ 0.3271

} $\vec{D}_2$ will be presented because it has the largest similarity value

---

Next we are going to try and reduce the amount of computation. We are going to do this by filtering out certain documents and then doing the computations.

Efficient Ranking

- Solve K-nearest neighbour problem for a query vector.

$SIM(\vec{Q}, \vec{D}_1)$ ... $SIM(\vec{Q}, \vec{D}_2)$

↘ ↙ We are looking for the largest.

CS4611 Thursday 7th November 2013

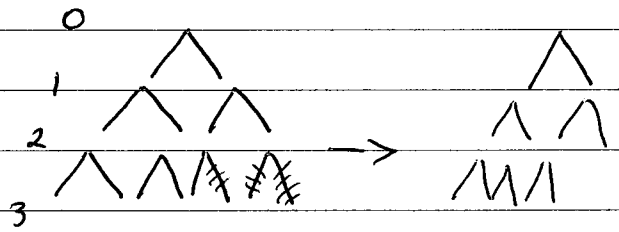
Heapify Exercise

[3, 1, 6, 2, 0, 9]

Heap: Complete Binary Tree, from L → R Order

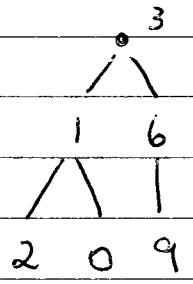
Create the heap produced for this list by the heapify algorithm.

A heap is a complete binary tree with leaves removed in Right to Left Order



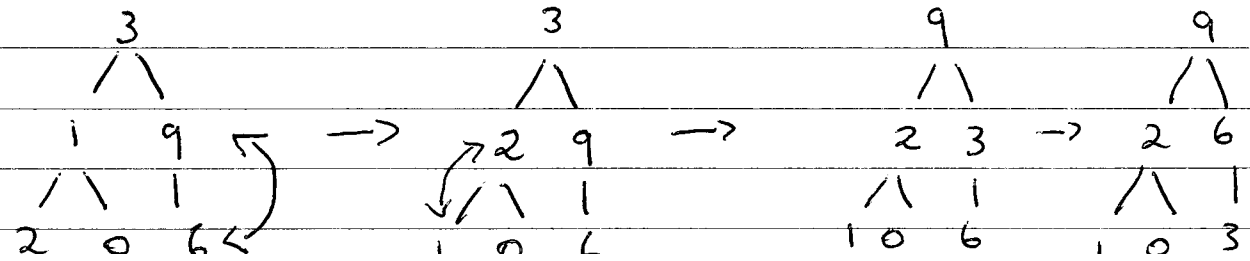
Label of Parent ⇒ labels of Children (in particular: largest label is at root)

A) Create nice tree labeled with List Element.

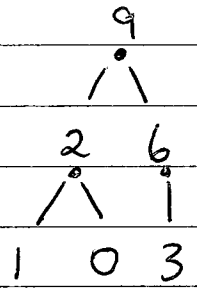


* Last Time / Near Heap

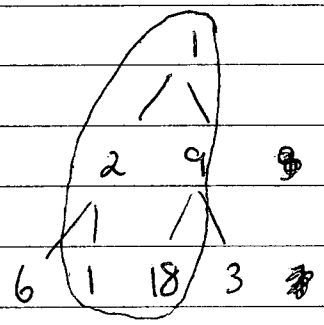
B) For Loop :



HEAP =>



Use Binary Tree - Heap for selecting top K



Generic Approach — Find set A of candidates
$K < |A| << N$

- Static Quality Scores
  - Top ranking documents to be both relevant and authoritative.

Relevance — Cosine Scores

Authority — query-independent property of a document
eg Wikipedia / Newspapers / papers citations.

Net Score

$$net\text{-}score(q, d) = g(d) + cosine(q, d)$$

Top K by Net Score

1/ Order all postings by $g(d)$

CS4611 Tuesday 12th November 2013

Assignment 2 : Cosine Similarity (See Notes)
Term Freq. (don't use log)
Find Closest match. term freq

Measures for a search Engine
- Measurable
- User Happiness - Can we measure this?
- Free
- Who is the user?

- Measure
  - Click-through rate
  - Rate of return to this search
  - Time to purchase, fraction of "conversions" of searches to buyers

User Happiness → feedback is relevant.
How to you measure relevance? ←
(proximity - cosine similarity)

Relevance to what?
- Information need.
- Query q
- Some results returned may match the query but is not what we are looking for.

User happiness can only be measured by relevance to the information retrieved.

$$\Delta \text{ precision} = \frac{\#(\text{relevant items retrieved})}{\#(\text{retrieved items})} = P(\text{relevant} \mid \text{retrieved})$$

$$\Delta \text{ recall} = \frac{\#(\text{relevant items retrieved})}{\#(\text{relevant items})} = R(\text{retrieved} \mid \text{relevant})$$

Accuracy
This is the fraction of decisions (relevant | nonrelevant) that are correct.

$$\text{accuracy} = (TP + TN)/(TP + FP + FN + TN)$$

=> roughly = 1, close to 100%
=> Not right way to go about it.

FP = False Positive - Small - Retrives Nothing - zero
FN = False Negative - Small
TP = True Positive - Small } relevance
TN = True Negative - Large }

$$\frac{1}{2}P + \frac{1}{2}R = \text{Arithmetic Mean.}$$
(Harmonic mean)

Returns everything - answer is there for sure - 50% accuracy this is still very bad.

$$\frac{1}{2}P + \frac{1}{2}R = \frac{1}{2}\,20/1{,}000{,}120 + \frac{1}{2}\,20/20 \approx \frac{1}{2}$$

F: Harmonic Mean

We can view the HM as a kind of smooth minimum.

$$F = \frac{1}{\frac{1}{2}\frac{1}{P} + \frac{1}{2}\frac{1}{R}}$$

$$F_1 = \frac{2PR}{P+R}$$

FORMULA

$$= \frac{1}{\frac{1}{2P} + \frac{1}{2R}}$$

$$= \frac{1}{\frac{2P+2R}{2P2R}} = \frac{4PR}{2P+2R}$$

What we need for a benchmark

- Collection of documents
- Collection of information needs (i.e. queries)
- Human relevance assessments

Benchmarks

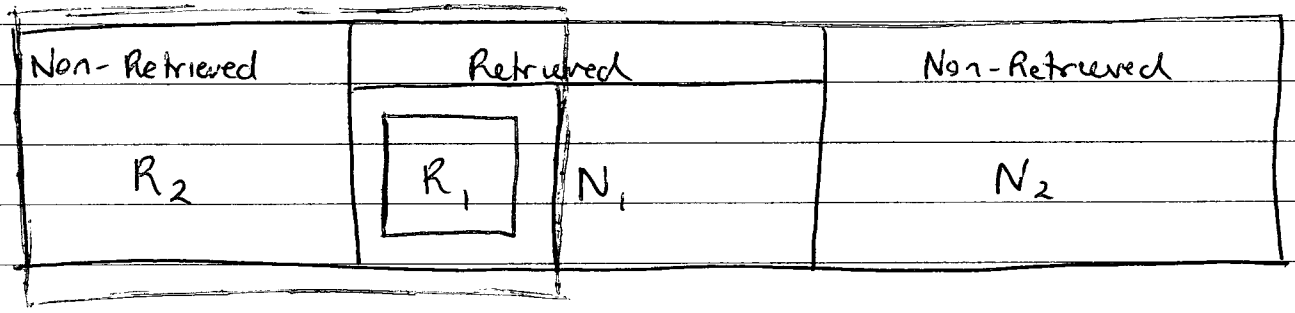
- Cranfield (quite old, too small & too untypical for today
- TREC (Text Retrieval Conference) - Too expensive
- GOV2
- NTCIR } others
- CLEF

CS4611 Thursday 14th November 2013

Precision (P) = $\frac{\#\text{ relevant items retrieved}}{\#\text{ retrieved items}}$

Recall (R) = $\frac{\#\text{ relevant items retrieved}}{\#\text{ relevant items}}$

Enumerator Same



Not Relevant = N

Relevant = R

$P = \frac{\#R_1}{(\#R_1 + \#N_1)}$ ↑ Red Box

$R = \frac{\#R_1}{(\#R_1 + \#R_2)}$ ↑ Green Box

△ Aim: To represent P and R by via single number

✱ Decided: Use something like Min (P, R)

System 1 : $P = \frac{1}{8}$ $R = \frac{5}{9}$

System 2 : $P = \frac{1}{8}$ $R = \frac{7}{9}$

S2 choose because it has slightly higher recall

M (P, R) for S1 = M (P, R) for S2 = Same $\left(\frac{1}{8}\right)$

Exercise

What happens with our measure F (Harmonic Mean),

Say $\alpha = \frac{1}{2}$, $F = \frac{1}{\frac{1}{2}\frac{1}{P} + \frac{1}{2}\frac{1}{R}}$

$$\frac{1}{\frac{2P+2R}{4PR}} = \boxed{\frac{2PR}{P+R} = F}$$

F (System 1) = 0.204 = 0.2

F (" 2) = 0.215 = 0.22 ✓ Better

KAPPA Measure

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

P(A) = proportion of time judges agree

P(E) = what agreement would we get by chance

Example:

$$\frac{(300+70)}{400} = \frac{370}{400} = 0.925$$

P (non-rel) - (80+90) / (400+400) : $\frac{170}{800}$ = 0.2125

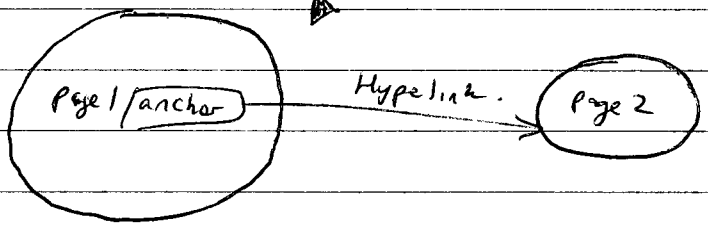
P (relevant) - (320+310) / (400+400) → $\frac{630}{800}$ = 0.7878

Summary

- Static Summary (subset of document, eg 1st 50 words)
- Dynamic Summary (snippets of info, that contain a no. of query elements - also occur together "the ball", hit "the ball")

- Evaluating webpages - links that go to them. Google Page Ranking.

- Citation Analysis
- Anchor Text
- Page Rank
- Hubs



Anchor Text is often a better description than the text itself.

CS4611 Thursday 21st November 2013

Matrix Multiplication.

$$\begin{bmatrix} 1 & 3 & 2 \\ 0 & 4 & 0 \end{bmatrix} \times \begin{bmatrix} 1 \\ 6 \\ 2 \end{bmatrix} = \begin{bmatrix} (23) \\ (24) \end{bmatrix}$$

2×3 (rows × cols) $\quad 3 \times 1$

row 1 × col 1
$1 \times 1 = 1$
$3 \times 6 = 18$ add up $= (23)$
$2 \times 2 = 4$

row 2 × col 1
$0 \times 1 = 0$
$4 \times 6 = 24 \rightarrow (24)$
$0 \times 2 = 0$

Page Ranking
(Compute importance of webpage - links there are)

Consider $N \times N$ matrices store probability of moving from page i to page j

$$\begin{matrix} N \\ \\ \\ \\ \end{matrix} \begin{bmatrix} X & P_{12} & P_{12} & P_{13} \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ P_{N1} & & & P_{NN} \end{bmatrix}$$

N

P = Probability that user will jump from Page 1 to Page N
X = Does not jump - stays on page.

Slides : Example Web Graph

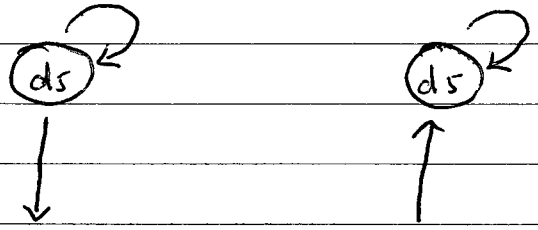
High Probability Page is d_3 as all others can reach it through links. d_5 is isolated.

d_0 = no link so 0

d_2 = link to itself so 1

	d_0	d_1	d_2
d_0	0	·	·
d_1	·	·	·
d_2	·	·	1

* The web graph must corr. with ergodic Markov chain
The web graph must not contain any dead ends.



No dead end Dead End.

If you reach a dead-end the probability to jump to another page is equal. (1/N)

A non-dead-end – probability 10% will jump to a random webpage. (0.1/N)

90% Keep going (0.9/N) ie following links.

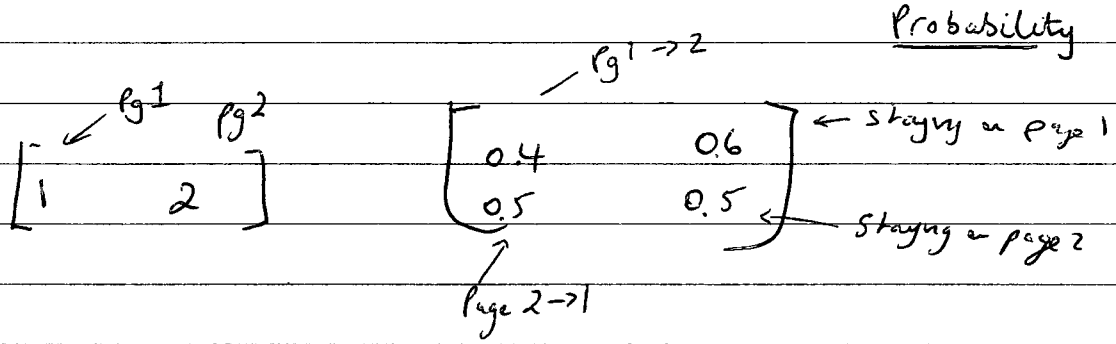
△ 10% is a parameter, the Teleportation Rate (TR)
★ "Jumping" from a dead end is independent from the TR

* Irreducibility } check slides - Intro. to Inf. Ret.
* Aperiodicity }

CS4611 Tuesday 26th November 2013

Formalisation of "visit"

How Random!



As X_m grows they will always stabilise - go to a steady state, ie when it reaches 1,000 or 1 million we will stop here.

Pg 170 Slides

$$P = \begin{bmatrix} 0.1 & 0.9 \\ 0.3 & 0.7 \end{bmatrix} \leftarrow \text{stay on Pg2}$$

Only give the bottom values - 0.3 0.7

↓

0.25 0.75

chances of landing on page 1 + 2

to 0 1

Random choice will always give the same outcome

0.75 high probability of moving to Pg2 and staying on Pg2.

$\vec{\pi} = (x_1, x_2)$

$\vec{\pi} = \vec{\pi} P$

$(x_1, x_2) = (x_1, x_2) P$ $\qquad P = \begin{bmatrix} 0.1 & 0.9 \\ 0.3 & 0.7 \end{bmatrix}$

$(x_1, x_2) = 0.1 x_1 + 0.3 x_2$

$\otimes 0.9 x_1 + 0.7 x_2$

$x_1 = 0.1 x_1 + 0.3 x_2$

$x_2 = 0.9 x_1 + 0.7 x_2$

Solve system of simultaneous equation
- Should give the same answer $(0.25, 0.75)$
- There are two ways to do this.

Starting Vector $(x_1, x_2) = (0, 1)$

Step 1 : Solve Power method

Step 2 : Solve $\vec{\pi} = \vec{\pi} P$ $\quad (\vec{\pi} = (x_1, x_2))$

Solution on pg 185 of slides

$\vec{\pi} P = (0.7 x_1 + 0.2 x_2, \; 0.3 x_1 + 0.8 x_2)$

$x_1 = 0.7 x_1 + 0.2 x_2$

$x_2 = 0.3 x_1 + 0.8 x_2$ $\quad$ SOLVE

$(x_1, x_2) = 0.4, 0.6$