Exercise 1

Oleksandr Tarasov

2786693

Information Retrieval

Supervisors: Prof. Goran Glavas, Benedikt Ebing, Fabian David Schmidt

Graz, den 22.05.2024

1 Boolean retrieval - Inverted Index

Consider the following document collection:

- Doc1: Facebook's new tool is called Graph Search.
- Doc2: A new social graph for LinkedIn users.
- Doc3: Find friends using search on Facebook.
- Doc4: Google's Knowledge Graph lets you search for things, people, or places

1.1

Draw the term-document incidence matrix for this document collection.

Word	Doc1	Doc2	Doc3	Doc4
facebook	1	0	1	0
's	1	0	0	1
new	1	1	0	0
tool	1	0	0	0
called	1	0	0	0
graph	1	1	0	1
search	1	0	1	1
social	0	1	0	0
linkedin	0	1	0	0
users	0	1	0	0
find	0	0	1	0
friends	0	0	1	0
using	0	0	1	0
google	0	0	0	1
knowledge	0	0	0	1
lets	0	0	0	1
you	0	0	0	1
things	0	0	0	1
people	0	0	0	1
places	0	0	0	1

1.2

Draw the inverted index representation for this document collection, showing the dictionary and the postings.

- facebook: [1, 3]
- 's: [1, 4]
- new: [1, 2]

- tool: [1]
- called: [1]
- graph: [1, 2, 4]
- search: [1, 3, 4]
- social: [2]
- linkedin: [2]
- users: [2]
- find: [3]
- friends: [3]
- using: [3]
- google: [4]
- knowledge: [4]
- lets: [4]
- you: [4]
- things: [4]
- people: [4]
- places

1.3

Use both the term-document incidence matrix and the inverted index to compute the results return for the following queries:

- graph AND search

  for term-doc matrix:

Word	Doc1	Doc2	Doc3	Doc4
graph	1	1	0	1
search	1	0	1	1
graph AND search	1	0	0	1

for inverted index:
graph: [1, 2, 4]
search: [1, 3, 4]
graph AND search: [1, 4]

- graph AND NOT (google or facebook)

  for term-doc matrix:

Word	Doc1	Doc2	Doc3	Doc4
google	0	0	0	1
facebook	1	0	1	0
google OR facebook	1	0	1	1
NOT (google OR facebook)	0	1	0	1
graph	1	1	0	1
graph AND NOT (google or facebook)	0	1	0	0

for inverted index:
google: [4]
facebook: [1, 3]
google OR facebook: [1, 3, 4]
NOT (google or facebook): [2]
graph: [1, 2, 4]
graph AND NOT (google OR facebook)

1.4

For a conjunctive query, is processing posting lists in order of size guaranteed to be optimal? Explain why it is, or give an example where it isn't.

The ordering by the size of postings does not really matter for optimal processing. It is dependent on the posting intersection size. Imagine:

$$a < b < c$$
$$a \cap b = d$$
$$a \cap c = e$$
$$d > e$$
$$a\&b\&c = a + b + d + c > a + c + e + b = a\&c\&b$$

1.5

What is the complexity (in big O notation) for a query x AND y when the postings lists are sorted? What if they aren't?
O(x + y) if sorted, O(xy) if not

1.6

How should the Boolean query x AND NOT y be handled? Why is the naive evaluation of the query, i.e. evaluating NOT y first and then x AND NOT y, normally very expensive? How expensive is it (in big O notation)? How expensive is a Boolean query x OR y?

This operation consists of 2 operations: NOT for the linear O(Y) time and AND for the linear O(x + y) time. Making these operations together leads to an increasing constant near the y variable under the) notation. This problem can be treated better by changing the AND algorithm. We will need to incorporate 2 changes into the standard MERGE operation:

- if doc(x) == doc(y) then just step for forward with both pointers
- if doc(x) < doc(y) then add doc(x) into the result

This algorithm also works for the linear time O(x + y).

2 Boolean Retrieval - Skip Pointers

2.1

Why are skip pointers not useful for queries in the form x OR y?

Skip pointers are not useful for "x or y"queries because skip pointers are designed to optimize sequential access or traversal of data structures, while "x or y"queries require evaluating both x and y independently. Skip pointers cannot bypass the evaluation of both elements and therefore do not provide any advantage in this scenario. Other techniques, such as logical operations or indexing, are more suitable for optimizing "x or y"queries.

2.2

We have a two-word query. For one term the postings list consists of the following 16 entries:

$$[4, 6, 10, 12, 14, 16, 18, 20, 22, 32, 47, 81, 120, 122, 157, 180]$$

and for the other one it is the one-entry postings list:

$$[47]$$

Work out how many comparisons would be done to intersect the two postings lists with the following two strategies. Briefly justify your answers:

- using standard posting lists.
  There will be 11 comparisons. 10 elements of the first array will be checked, by every comparison we mean checking for equality, and then for the question is the element x bigger or smaller than element y from the 2nd array.

- Using postings lists stored with skip-pointers, with a skip length of $\sqrt{L}$, as suggested
  There will be done 6 comparisons in the second lecture (s.35).
  The skip pointers that we need will be done for the first list in this way:

  $$4 \rightarrow 14 \rightarrow 22 \rightarrow 120$$

  So, we will check pairs: (4, 47), (14, 47), (22, 47), (120, 57), (32, 47), (47, 47). It will be 6 comparisons at all.

2.3

Consider a postings intersection between this postings list, with skip pointers:

$$[3, 5, 9, 15, 24, 39, 60, 68, 75, 81, 84, 89, 92, 96, 97, 100, 115]$$
$$3 \rightarrow 24 \rightarrow 75 \rightarrow 92 \rightarrow 115$$

and the following intermediate result postings list (which hence has no skip pointers:

$$[3, 5, 89, 95, 97, 99, 100, 101]$$

- How often is a skip pointer followed?
  Only once, a skip pointer $24 \rightarrow 75$

- How many postings comparisons will be made by this algorithm while intersecting the two lists?

  18 comparisons at all.
  - (3, 3)
  - (5, 5)
  - (9, 89)

  - (15, 89)
  - (24, 89)
  - (75, 89)
  - (92, 89)
  - (81, 89)
  - (84, 89)
  - (89, 89)
  - (92, 95)
  - (115, 95)
  - (96, 95)
  - (96, 97)
  - (97, 97)
  - (100, 99)
  - (100, 100),
  - (115, 101)
- How many postings comparisons are made if the postings lists are intersected without the use of skip pointers?
  19 comparisons.

3 Phrase and Proximity Queries

3.1

Shown below is a portion of a positional index in the format: term: doc1: <position1, position2, ...>; doc2: <position1, position2, ...>; etc.

angels: 2: ⟨36,174,252,651⟩; 4: ⟨12,22,102,432⟩; 7: ⟨17⟩;
fools: 2: ⟨1,17,74,222⟩; 4: ⟨8,78,108,458⟩; 7: ⟨3,13,23,193⟩;
fear: 2: ⟨87,704,722,901⟩; 4: ⟨13,43,113,433⟩; 7: ⟨18,328,528⟩;
in: 2: ⟨3,37,76,444,851⟩; 4: ⟨10,20,110,470,500⟩; 7: ⟨5,15,25,195⟩;
rush: 2: ⟨2,66,194,321,702⟩ 4: ⟨9,69,149,429,569⟩; 7: ⟨4,14,404⟩;
to: 2: ⟨47,86,234,999⟩; 4: ⟨14,24,774,944⟩; 7: ⟨199,319,599,709⟩;
tread: 2: ⟨57,94,333⟩; 4: ⟨15,35,155⟩; 7: ⟨20,320⟩;
where: 2: ⟨67,124,393,1001⟩; 4: ⟨11,41,101,421,431⟩; 7: ⟨16,36,736⟩;

Which document(s), if any, meet each of the following queries, where each expression within quotes is a phrase query?

- "fools rush in"
  $[2, 4, 7]$

- "fools rush inÄND ängels fear to tread"
  $[4]$

3.2

Consider the following fragment of a positional index with the same format:

Gates: 1: ⟨3⟩; 2: ⟨6⟩; 3:⟨2,17⟩; 4: ⟨1⟩;
IBM: 4: ⟨3⟩; 7: ⟨14⟩;
Microsoft: 1: ⟨1⟩; 2: ⟨1,21⟩; 3: ⟨3⟩; 5: ⟨16,22,51⟩;

- Describe the set of documents that satisfy the query Gates /2 Microsoft
  $[1, 3]$

- Describe each set of values for k for which the query Gates /k Microsoft returns a different set of documents as the answer.

  - $k = 1, [1, 3]$
  - $1 < k < 5, [1, 2, 3]$
  - $k \geq 5, [1, 2, 3]$

4 Tolerant Retrieval

4.1

Write down the entries in the permuterm index dictionary that are generated by the term Retrieval.

- Retrieval$
- l$Retriva
- al$Retriv
- val$Retri
- ival$Retr
- rival$Ret
- trival$Re
- etrival$R

4.2

If you want to search for s*ng in a permuterm wildcard index, what key(s) would one do the lookup on?

ng$s*

4.3

Consider the following example of a postings list in a 3-gram index.

$$etr \rightarrow beetroot \rightarrow metric \rightarrow petrify \rightarrow retrieval$$

Why is it useful to have the vocabulary terms in the postings lexicographically ordered?

Intersecting sorted lists is faster and simpler during filtering dictionary candidates.

4.4

We want to compute the Levenshtein edit distance between Frodo and Gondor. Consider the sub-problem of computing the distance between G and Frod. What are the costs for insertion, deletion, and replacement respectively?

		F	r	o	d
	0	1	2	3	4
G	1	1	2	3	4

We will take care of the cell [G, d] in detail:

replacement = 4	insertion = 5
deleting = 4	minimum = min(4, 4, 5) = 4

- insertion: 5
- replacement: 4
- deleting: 4

4.5

Write down the full 6×5 array of distances between all prefixes as shown in lecture 3. What is the minimum edit distance between Frodo and Gondor?

		F	r	o	d	o
	0	1	2	3	4	5
G	1	1	2	3	4	5
o	2	2	2	2	3	4
n	3	3	3	3	3	4
d	4	4	4	4	3	4
o	5	5	5	4	4	3
r	6	6	5	5	5	4

So, the minimum distance is 4.

4.6

What is the Levenshtein-Damerau distance between hill and goblin? Write down the solution in the same tabular format from the previous task.

We will use the same table as in the previous task but also will consider the $d(i-2, j-2) + 1 * (a_{i-1} = b_j bj - 1 = a_i)$

		G	o	b	l	i	n
	0	1	2	3	4	5	6
h	1	1	2	3	4	5	6
i	2	2	2	3	4	4	5
l	3	3	3	3	3	4	5
l	4	4	4	4	3	4	5

The minimum edit distance is 5.

4.7

What is the Jaccard coefficient between the word bord and each of lord, morbid, and sordid when we treat them as bigrams?

bord = [bo, or, rd]

- lord = [lo, or, rd]
  Jaccard(bord, lord) = $\frac{2}{4}$

- morbid = [mo, or, rb, bi, id]
  Jaccard(bord, morbid) = $\frac{1}{7}$

- sordid = [so, or, rd, di, id]
  Jaccard(bord, sordid) = $\frac{2}{6}$