Seminar 2

Algorithm 1 (Soundex Code)
Transformation of a string to a 4-character soundex code

1. *Keep the first character*
2. *Rewrite* $\{A, E, I, O, U, H, W, Y\}$ *to 0*
3. *Rewrite characters*
   (a) $\{B, F, P, V\}$ *to 1*
   (b) $\{C, G, J, K, Q, S, X, Z\}$ *to 2*
   (c) $\{D, T\}$ *to 3*
   (d) $\{L\}$ *to 4*
   (e) $\{M, N\}$ *to 5*
   (f) $\{R\}$ *to 6*
4. *Remove duplicities*
5. *Remove zeros*
6. *Change to length 4 (truncate or add trailing zeros)*

Algorithm 2 (Querying in Permuterm Index)
For query q, find keys according to the following scheme:

- *for* $q = X$*, find keys in the form* $X\$$
- *for* $q = X^*$*, find keys in the form* $\$X^*$
- *for* $q = {}^*X$*, find keys in the form* $X\$^*$
- *for* $q = {}^*X^*$*, find keys in the form* X^*
- *for* $q = X^*Y$*, find keys in the form* $Y\$X^*$

Exercise 2/1

Below is a part of index with positions in the form
doc1: $\langle pos1, pos2, pos3, \ldots\rangle$; doc2: $\langle pos1, pos2, \ldots\rangle$; ...

- angels: $2 : \langle 36, 174, 252, 651\rangle$; $4 : \langle 12, 22, 102, 432\rangle$; $7 : \langle 17\rangle$;
- fools: $2 : \langle 1, 17, 74, 222\rangle$; $4 : \langle 8, 78, 108, 458\rangle$; $7 : \langle 3, 13, 23, 193\rangle$;
- fear: $2 : \langle 87, 704, 722, 901\rangle$; $4 : \langle 13, 43, 113, 433\rangle$; $7 : \langle 18, 328, 528\rangle$;
- in: $2 : \langle 3, 37, 76, 444, 851\rangle$; $4 : \langle 10, 20, 110, 470, 500\rangle$; $7 : \langle 5, 15, 25, 195\rangle$;
- rush: $2 : \langle 2, 66, 194, 321, 702\rangle$; $4 : \langle 9, 69, 149, 429, 569\rangle$; $7 : \langle 4, 14, 404\rangle$;
- to: $2 : \langle 47, 86, 234, 999\rangle$; $4 : \langle 14, 24, 774, 944\rangle$; $7 : \langle 19, 319, 599, 709\rangle$;
- tread: $2 : \langle 57, 94, 333\rangle$; $4 : \langle 15, 35, 155\rangle$; $7 : \langle 20, 320\rangle$;
- where: $2 : \langle 67, 124, 393, 1001\rangle$; $4 : \langle 11, 41, 101, 421, 431\rangle$; $7 : \langle 15, 35, 735\rangle$;

The following terms are phrase queries. Which documents correspond to the following queries and on which positions?

a) *fools rush in*

b) *fools rush in* AND *angels fear to tread.*

The index is incorrect. How?

In order to retrieve the query it is necessary that the words are in a sequence. That is, if the word *angels* is in **doc2** on position 36, then the word *fear* has to be in the same document on the position 37 and so on.

For the exercise **a)** we calculate all possible positions of the phrase. Word *fools* appears in **doc2** on positions $\langle 1, 17, 74, 222\rangle$. That means that the word *rush* has to appear on positions $\langle 2, 18, 75, 223\rangle$ and the word *in* on positions $\langle 3, 19, 76, 224\rangle$. Similar process is applied on **doc4** and **doc7** which retrieves the requested results.

doc2 $\langle 1, 2, 3\rangle$, $\langle 17, 18, 19\rangle$, $\langle 74, 75, 76\rangle$, $\langle 222, 223, 224\rangle$

doc4 $\langle 8, 9, 10\rangle$, $\langle 78, 79, 80\rangle$, $\langle 108, 109, 110\rangle$, $\langle 458, 459, 460\rangle$

doc7 $\langle 3, 4, 5\rangle$, $\langle 13, 14, 15\rangle$, $\langle 23, 24, 25\rangle$, $\langle 193, 194, 195\rangle$

Now we look at the original position index and search for whether there is a conjunction between requested and real positions. Take **doc2** and check whether the words *fools*, *rush* and *in* are in a sequence on positions $\langle 1, 2, 3\rangle$. Since yes, the system returns **doc2** as relevant to our query. Same analogy is used for the remaining documents for which we get the result **doc2**: $\{\langle 1, 2, 3\rangle\}$; **doc4**: $\{\langle 8, 9, 10\rangle\}$; and **doc7**: $\{\langle 3, 4, 5\rangle, \langle 13, 14, 15\rangle\}$.

For the exercise **b)** we find the requested positions for also the term *angels fear to tread.*

doc2 $\langle 36, 37, 38, 39\rangle$, $\langle 174, 175, 176, 177\rangle$, $\langle 252, 253, 254, 255\rangle$, $\langle 651, 652, 653, 654\rangle$

doc4 $\langle 12, 13, 14, 15\rangle$, $\langle 22, 23, 24, 25\rangle$, $\langle 102, 103, 104, 105\rangle$, $\langle 432, 433, 434, 435\rangle$

doc7 $\langle 17, 18, 19, 20\rangle$

They appear in the correct order in **doc4**: $\{\langle 12, 13, 14, 15\rangle\}$ and in **doc7**: $\{\langle 17, 18, 19, 20\rangle\}$. Taking the first part from **a)**, we only check whether the results overlap {**doc2**(1), **doc4**(8), **doc7**(3), **doc7**(13)} $\cap$ {**doc4**(12), **doc7**(17)} = {**doc4**(8,12), **doc7**(3,17), **doc7**(13,17)}.

The index is incorrect. We need to have a look into **doc7**, where on position 15 are two terms *in* and *where*.

Exercise 2/2

Below is a part of index with positions in the form
doc1: $\langle pos1, pos2, pos3, \ldots\rangle$; doc2: $\langle pos1, pos2, \ldots\rangle$; ...

- ostrich: 1 : <1,7>; 2 : <4,5>;
- hippo: 1 : <5,8,9>; 3 : <6,9>;
- lion: 1 : <3,6>; 2 : <3,7>;
- giraffe: 1 : <2,4>; 2 : <1,2,8>;

Which documents correspond to the phrase query *lion giraffe hippo* and on which positions? Include intermediate results.

Candidates:

doc1 $\langle 3, 4, 5\rangle$, $\langle 6, 7, 8\rangle$

doc2 $\langle 3, 4, 5\rangle$, $\langle 7, 8, 9\rangle$

Result:

doc1 $\langle 3, 4, 5\rangle$

Exercise 2/3

Consider a query composed of two terms. Non-positional postings list of one term is composed of 16 items $P = [4, 6, 10, 12, 14, 16, 18, 20, 22, 32, 47, 81, 120, 215, 300, 500]$ and the second term has the postings list of only a single element $R = [47]$. Find out how many comparisons (and why) are necessary to find out the intersection of the lists that are organized as follows:

a) standard postings lists

b) postings lists with skip pointers of skip frequency $\sqrt{|P|}$

---

a) A naive algorithm compares each element from P with each element from R, which is 11.

b) With skip pointers of frequency $\sqrt{|P|} = 4$ we reduce the number of comparisons. Instead of the next element $i + 1$ only, the pointer goes directly to $i + \sqrt{|P|}$ until the referred value is larger than the searched value, after which it jumps back and then step forward by 1. Starting at position 0, the algorithm proceeds as follows: compare 4:47, jump to 14, compare 14:47, jump to 22, compare 22:47, jump to 120, compare 120:47, jump back to 22, step to 32, compare 32:47, step to 47, compare 47:47, done. The number of compare operations is 6.

Exercise 2/4

Consider a query composed of two terms. Non-positional postings list with skip pointers of one term is composed of 16 items $P_1 = [4, 6, 10, 12, 14, 16, 18, 20, 22, 32, 47, 81, 120, 215, 300, 500]$ with skip frequency of square root of its length and the second term has the standard postings list $P_2 = [18, 32, 60]$. How many comparisons are necessary to find out the intersection of the lists?

$(4, 18)$, $(14, 18)$, $(22, 18)$, $(16, 18)$, $(18, 18)$, $(20, 32)$, $(22, 32)$, $(120, 32)$, $(32, 32)$, $(47, 60)$, $(81, 60)$.

Exercise 2/5

List the comparisons performed to intersect the following sorted non-positional postings lists with skip pointers of frequency 5.

$$P_1 = [2, 10, 12, 16] \quad \text{and} \quad P_2 = [1, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]$$

$(2,1)$, $(2,7)$, $(2,3)$, $(10,3)$, $(10,4)$, $(10,5)$, $(10,6)$, $(10,7)$, $(10,12)$, $(10,8)$, $(10,9)$, $(10,10)$, $(12,11)$, $(12,12)$, $(16,13)$, $(16,14)$, $(16,15)$.

Exercise 2/6

List the comparisons performed to intersect the following sorted non-positional postings lists with skip pointers of frequency 5.

$$P_1 = [4, 5, 6, 7, 8, 9, 10, 13, 14, 15] \quad \text{and} \quad P_2 = [1, 2, 3, 4, 5, 10, 11, 15, 16]$$

$(4,1)$, $(4,10)$, $(4,2)$, $(4,3)$, $(4,4)$, $(5,5)$, $(6,10)$, $(7,10)$, $(8,10)$, $(9,10)$, $(10,10)$, $(13,11)$, $(13,15)$, $(14,15)$, $(15,15)$.

Exercise 2/7

a) Find two different words of the same soundex code.

b) Find two phonetically similar words of different soundex codes.

---

a) *sword* and *short* have codes S630

b) *fog* and *thug* have codes F200 and T200

Exercise 2/8

Write elements in a dictionary of the permuterm index generated by the term *mama*.

---

mama\$, *ama*\$*m*, *ma*\$*ma*, *a*\$*mam*, \$*mama*.

Exercise 2/9

Which keys are usable for finding the term *s*ng* in a permuterm wildcard index?

---

ng\$*s**

Exercise 2/10

What is the complexity of intersection of two un-ordered posting lists of lengths m and n?

$\mathcal{O}(m \log m + n \log n)$

Exercise 2/11

What is the complexity (in $\mathcal{O}$-notation) of intersecting of two ordered posting lists of lengths m and n?

$\mathcal{O}(m + n)$

Exercise 2/12

What is the worst-case complexity of searching in hash tables?

Linear.