Introduction to Information Retrieval and Text Mining Assignment 1

Roman Klinger

Institute for Natural Language Processing, University of Stuttgart

2017-11-14

Overview

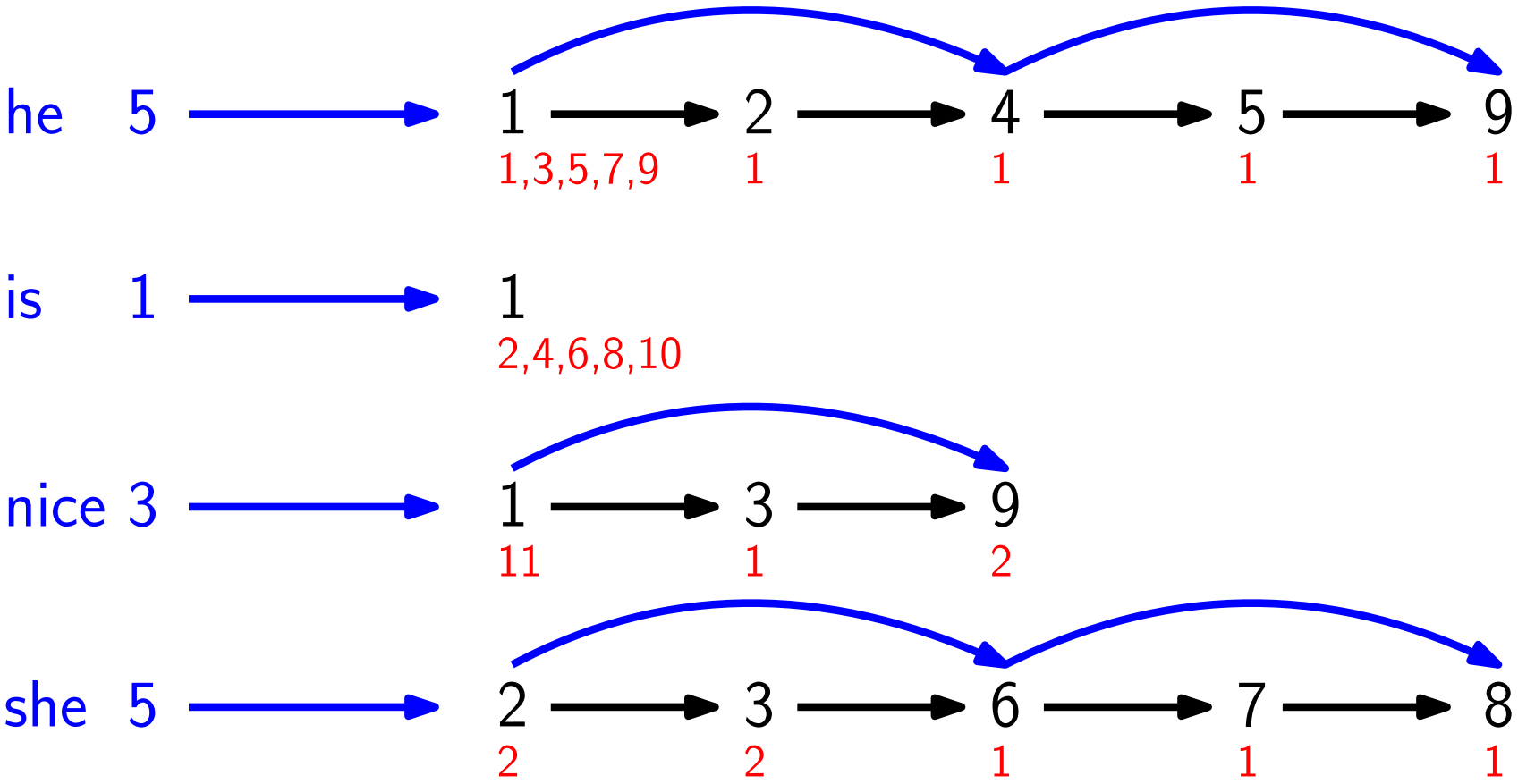
Task 1

Given the following documents:

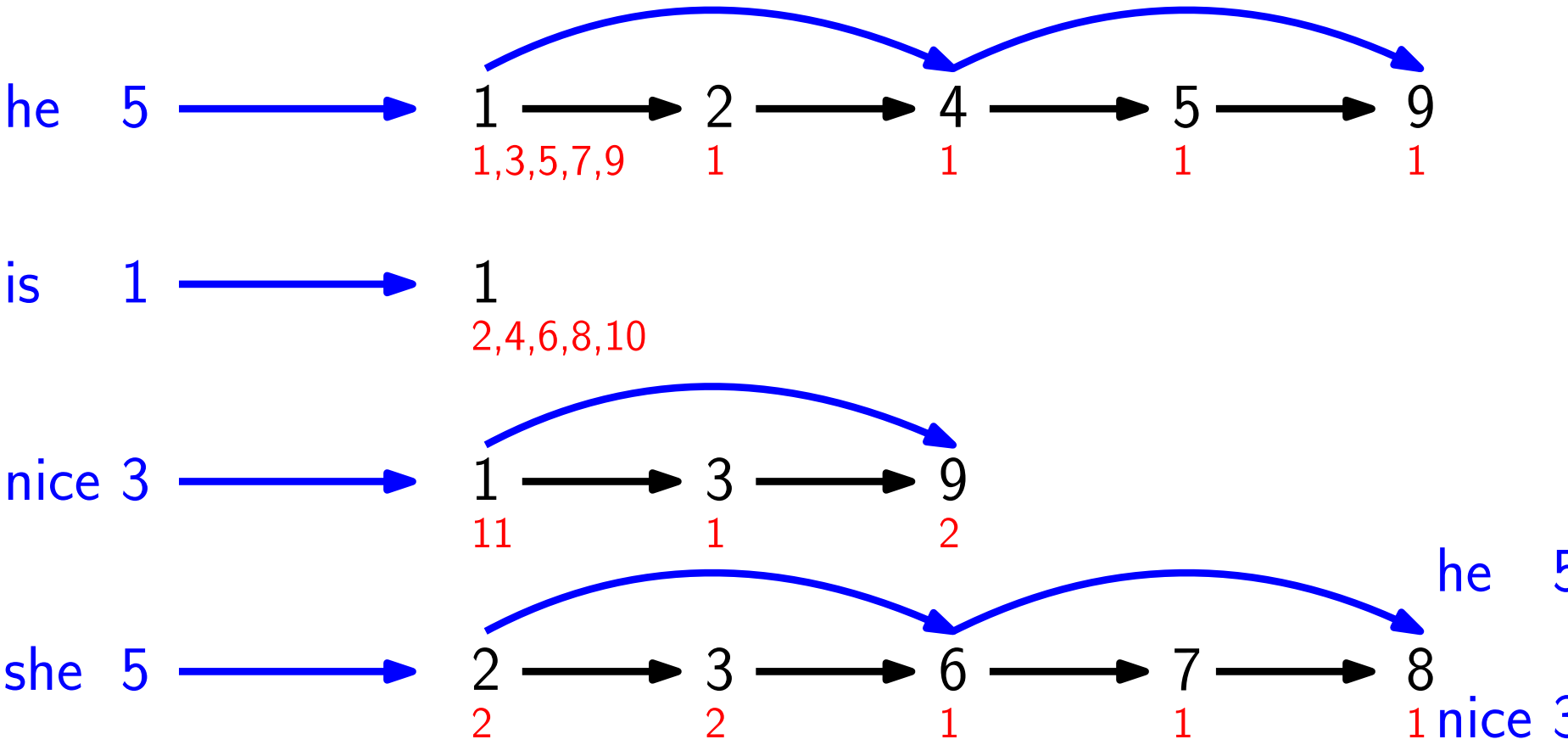
- Document 1: `he is he is he is he is he is nice`
- Document 2: `he she`
- Document 3: `she nice`
- Document 4: `he`
- Document 5: `he`
- Document 6: `she`
- Document 7: `she`
- Document 8: `she`
- Document 9: `he nice`

Build the full inverted *positional* index for these documents. Do not perform normalization.
Add skip pointers to the index from the previous task. Provide an example query which demonstrates the usefulness of skip pointers for your index and explain why this query can be answered in a more efficient way with skip pointers than without.

Task 1 (with heuristic of $\sqrt{\#doc}$)



Task 1



- Example query: "he AND nice"
- without skip pointers: 6 comparisons
  - 1-1, 2-3, 4-3, 4-9, 5-9, 9-9
- with skip pointers: 5 comparisons

Task 2

Are skip pointers helpful for queries in which two terms are connected by OR? Explain your answer!

Answers

- No, because all positions need to be printed anyway. Comparisons can be saved to ensure not printing results multiple times.

Task 3

- Which strings are stored in the permuterm index for the word `car`?
  - `car$ ar$c r$ca $car`
- How is this permuterm index queried when the user requests `c*r`?
  - `r$c*`
- Which of the strings in the permuterm is the one which answers the query?
  - `r$ca`

Task 4

Consider the following fragment of a positional index in the format
`word: document: <position, position, position, ...>; document: ...:`

```
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>;
```

Which comparisons are made when querying for
`Gates /2 Microsoft`?

Answer

```
1-1, |3-1|<3 -> OUTPUT
2-2, |6-1|<3, |6-21|<3
3-3, |2-3|<3, -> OUTPUT (|17-3|<3)
4-5
```

Task 5

The permuterm index and the k gram index are both approaches to make wildcard queries possible. Describe in your own words, what the advantage and disadvantages of each approach are. Explain!

Possible Answer

Permuterm

- Increases Size of Index
- No postfiltering needed

k gram

- Additional Index needed, but more space efficient
- Postfiltering might be necessary

Task 6

Calculate the Levenshtein distance for the two terms “flower” and “flour”. Draw the matrix and explain.

Solution

		f	l	o	w	e	r
	0	1	2	3	4	5	6
f	1	0	1	2	3	4	5
l	2	1	0	1	2	3	4
o	3	2	1	0	1	2	3
u	4	3	2	1	1	2	3
r	5	4	3	2	2	2	2

Task 7

Given the query `re*v*l`. One possibility to deal with multiple wildcards is to query for the margins: `l$re`. Does that work well? What are problems and how could these be addressed?

Possible Answer

- Problems: False positives (e.g. real)
- Possible Solution: Postprocessing necessary
- Another solution:
  Query for *v* and re*l and intersect.