Introduction to Information Retrieval and Text Mining Assignment 2

Roman Klinger

Institute for Natural Language Processing, University of Stuttgart

2017-11-30

Outline

Task 1 (Jaccard)

According to the Jaccard Index, which document d_i is more relevant to the given query q? What are the values for each comparison?

- q algorithm intersection
- d_1 the intersection algorithm for two documents is efficient
- d_2 intersection of two or more objects is another smaller object
- d_3 intersection algorithm

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

- $\textsc{Jaccard}(q, d_1) = \frac{2}{8}$
- $\textsc{Jaccard}(q, d_2) = \frac{1}{11}$
- $\textsc{Jaccard}(q, d_3) = \frac{2}{2} = 1$

$d_3 > d_1 > d_2$

Outline

Task 2 (TF·IDF)

d_1 the intersection algorithm for two documents is efficient

d_2 intersection of two or more objects is another smaller object

d_3 intersection algorithm

According to cosine similarity with tf·idf weights, which of the documents d_1, d_2, d_3 is more relevant to the query "algorithm intersection"? Explain your solution.

- $w_{t,d} = (1 + \log \mathrm{tf}_{t,d}) \cdot \log \frac{N}{\mathrm{df}_t}$ (if $\mathrm{tf}_{t,d} > 0$)
- $\cos(\vec{d}, \vec{q}) = \dfrac{\vec{d} \cdot \vec{q}}{|\vec{d}| \cdot |\vec{q}|}$
- Count/not log-weighted:

t	$\mathrm{tf}_{t,q}$	tf_{t,d_1}	tf_{t,d_2}	tf_{t,d_3}	idf_t
algorithm	1	1	0	1	3/2
intersection	1	1	1	1	3/3

Task 2 (TF·IDF) (2)

- $w_{t,d} = (1 + \log \mathrm{tf}_{t,d}) \cdot \log \frac{N}{\mathrm{df}_t}$ (if $\mathrm{tf}_{t,d} > 0$)
- $\cos(\vec{d}, \vec{q}) = \dfrac{\vec{d} \cdot \vec{q}}{|\vec{d}| \cdot |\vec{q}|}$
- Counts:

t	$\mathrm{tf}_{t,q}$	tf_{t,d_1}	tf_{t,d_2}	tf_{t,d_3}	idf_t
algorithm	1	1	0	1	3/2
intersection	1	1	1	1	3/3

- log-weights:

t	$\mathrm{tf}_{t,q}$	tf_{t,d_1}	tf_{t,d_2}	tf_{t,d_3}	idf_t
algorithm	1	1	n.d.(0)	1	0.18
intersection	1	1	1	1	0

- Vectors:

$$\vec{q} = \begin{pmatrix} 0.18 \\ 0.0 \end{pmatrix}, \vec{d_1} = \begin{pmatrix} 0.18 \\ 0.0 \end{pmatrix}, \vec{d_2} = \begin{pmatrix} 0.0 \\ 0.0 \end{pmatrix}, \vec{d_3} = \begin{pmatrix} 0.18 \\ 0.0 \end{pmatrix}$$

Task 2 (TF·IDF) (3)

- $\cos(\vec{d}, \vec{q}) = \dfrac{\vec{d} \cdot \vec{q}}{|\vec{d}| \cdot |\vec{q}|}$
- Vectors:
  $\vec{q} = \begin{pmatrix} 0.18 \\ 0.0 \end{pmatrix}, \vec{d_1} = \begin{pmatrix} 0.18 \\ 0.0 \end{pmatrix}, \vec{d_2} = \begin{pmatrix} 0.0 \\ 0.0 \end{pmatrix}, \vec{d_3} = \begin{pmatrix} 0.18 \\ 0.0 \end{pmatrix}$
- $\cos(\vec{d_1}, \vec{q}) = \cos(\vec{d_3}, \vec{q}) = \dfrac{0.0324}{0.18 \cdot 0.18} = 1.0$
- $\cos(\vec{d_2}, \vec{q})$ " $= 0.0$"
- $\Rightarrow d_1$ and d_3 are more similar to q than d_2

Note: We omitted terms in doc. here for simplicity of discussion.

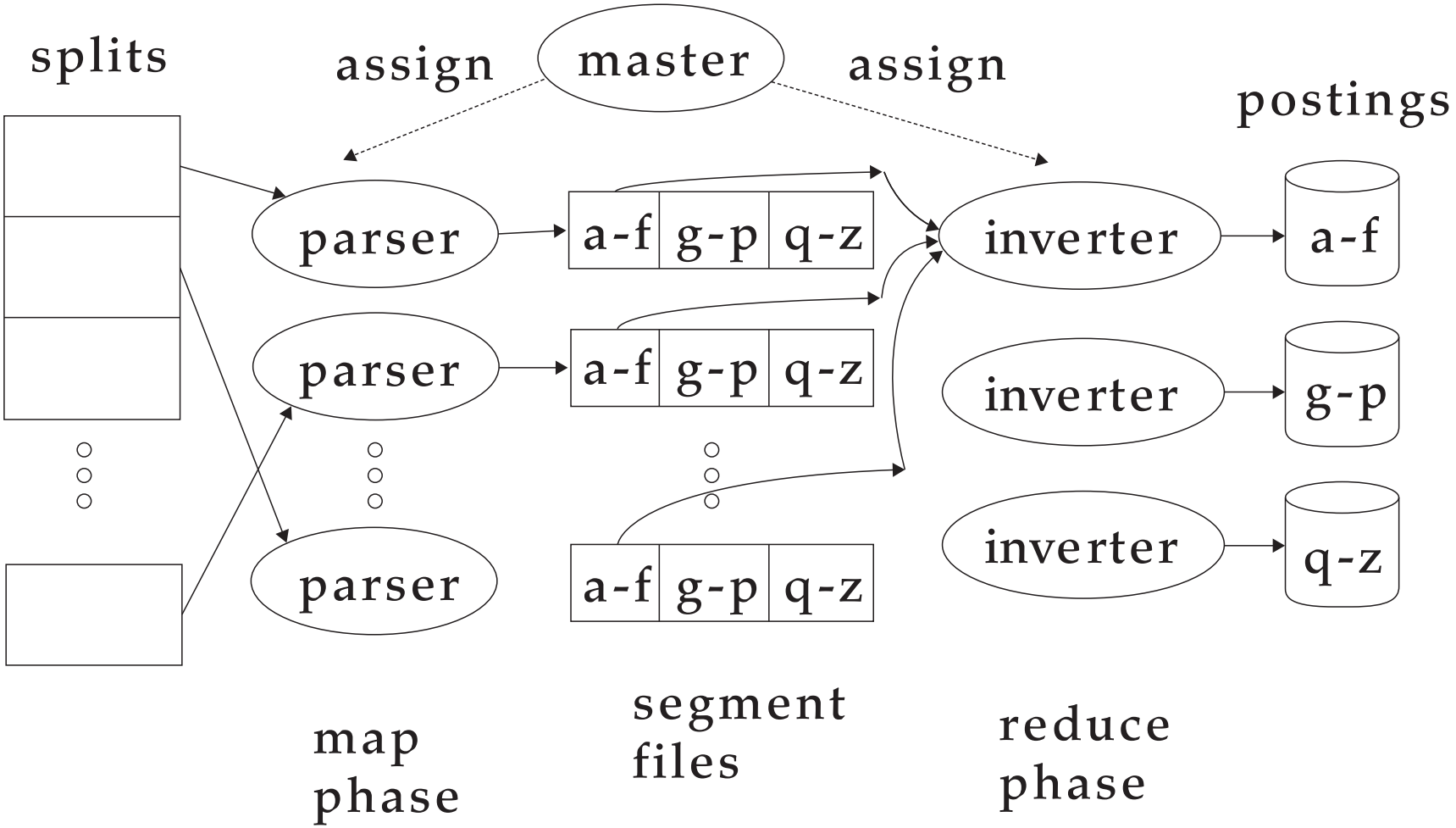
Outline

Task 3 (Distributed Indexing)

Answer the following questions about distributed indexing:

- What information does the task description contain that the master gives to a parser?
- What information does the parser report back to the master upon completion of the task?
- What information does the task description contain that the master gives to an inverter?
- What information does the inverter report back to the master upon completion of the task?

Task 3 (Distributed Indexing)



Task 3 (Distributed Indexing)

Answer the following questions about distributed indexing:

- What information does the task description contain that the master gives to a parser?
  set of document IDs, # of segments
- What information does the parser report back to the master upon completion of the task?
  position of prefix-based term partitions for the documents
- What information does the task description contain that the master gives to an inverter?
  (a prefix +) positions to all term partitions for a prefix
- What information does the inverter report back to the master upon completion of the task?
  position of the complete postings list for the prefix

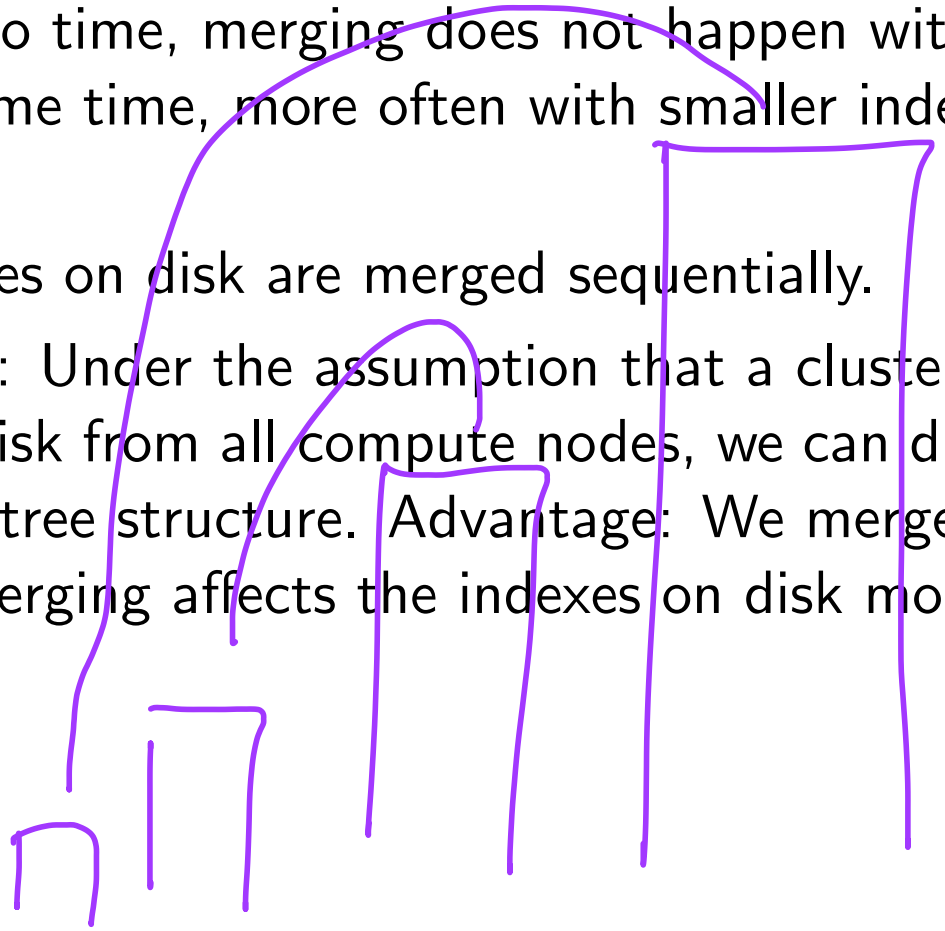
Outline

Task 4 (Distributed Logarithmic Merging)

Explain logarithmic merging in your own words. Include the motivation for this method in your explanation and make clear what advantages this method has in contrast to one auxiliary index and only one index on hard disk.
Please come up with an idea how logarithmic merging could be distributed by using a compute cluster. Can you resolve (at least partially) some disadvantages it has?

Task 4 (Distributed Logarithmic Merging)

- Idea 1: Use auxiliary index (fast index in memory)
- Idea 2: Keep indexes on disk in increasing order and merge them from time to time, merging does not happen with all indexes at the same time, more often with smaller indexes.
- Potential issue:
  Succeeding indexes on disk are merged sequentially.
- Idea to distribute: Under the assumption that a cluster can access the hard disk from all compute nodes, we can distribute the merging in a tree structure. Advantage: We merge faster. Disadvantage: Merging affects the indexes on disk more.



Task 4 (Distributed Logarithmic Merging)

- Idea 1: Use auxiliary index (fast index in memory)
- Idea 2: Keep indexes on disk in increasing order and merge them from time to time, merging does not happen with all indexes at the same time, more often with smaller indexes.
- Potential issue:
  Succeeding indexes on disk are merged sequentially.
- Idea to distribute: Under the assumption that a cluster can access the hard disk from all compute nodes, we can distribute the merging in a tree structure. Advantage: We merge faster. Disadvantage: Merging affects the indexes on disk more.
- Another idea: Make a copy of all indexes (if hard disk space allows) and use these for answering queries. Merge in parallel, when done, replace the copy by the merger.

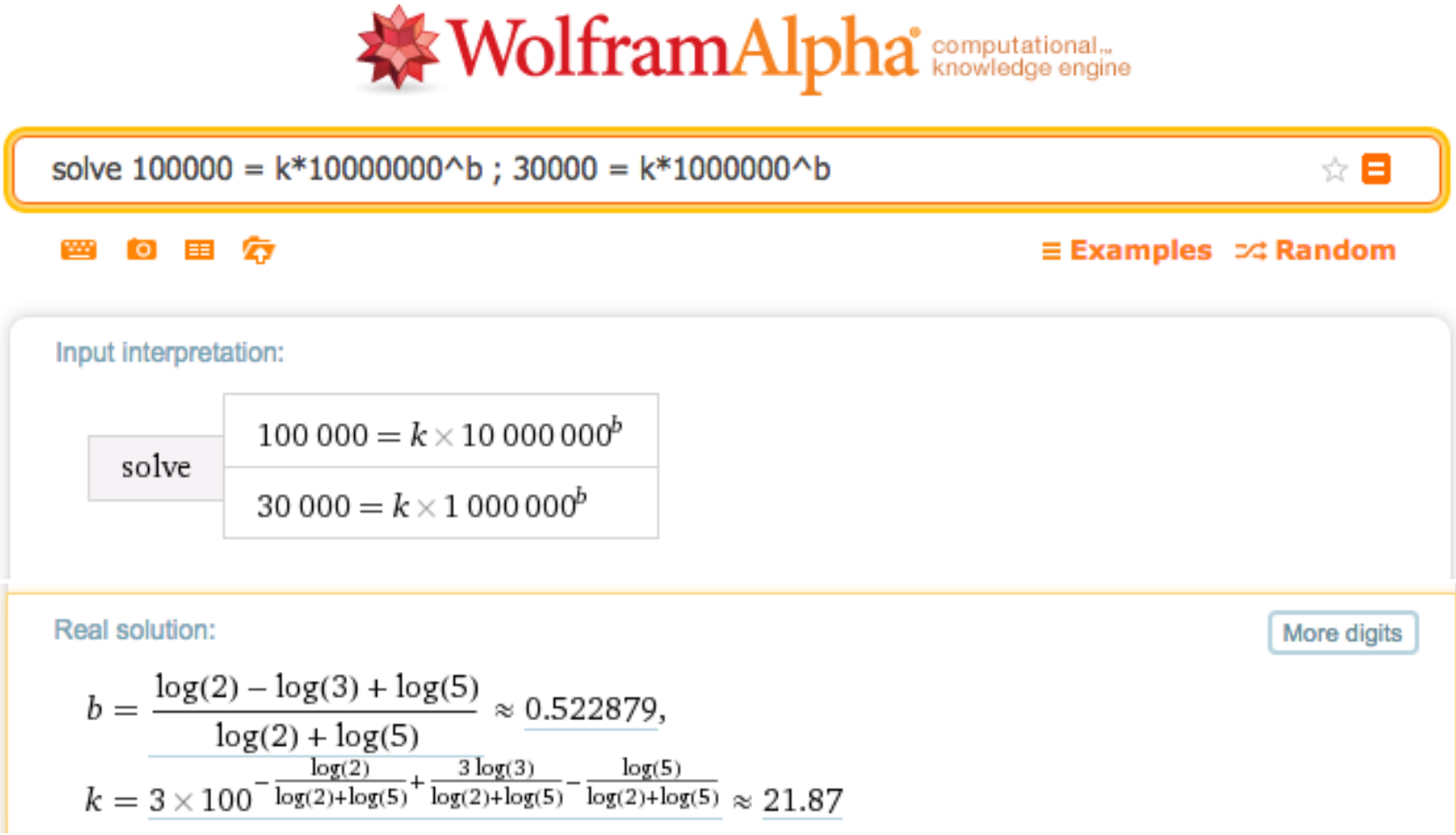
Outline

Task 5 (Heaps' Law)

dataset	collection size	vocabulary size
subset 1	10M	100K
subset 2	1M	30K

- Compute the coefficients k and b.
- Compute the expected vocabulary size for the complete collection (1G tokens).
- Heaps' Law: $M = kT^b$ (M: Vocabulary size, T: # Tokens)
- $100000 = k \cdot 10000000^b$
- $30000 = k \cdot 1000000^b$
- $b = 0.52287875$
- $k = 21.87$
- $21.87 \cdot 1000000000^{0.52287875} = 1111111$

Task 5 (Heaps' Law) II



Outline

Task 6 (Zipf's Law)

Given a collection that contains only four different words a, b, c, d. The frequency order is $a > b > c > d$. The total number of tokens in the collection is 6000. Assume that *Zipf's law* holds exactly for this collection. What are the frequencies of the four words?

- $a + b + c + d = 6000$
- $b = \frac{a}{2}$, $c = \frac{a}{3}$, $d = \frac{a}{4}$
- $a + \frac{a}{2} + \frac{a}{3} + \frac{a}{4} = 6000 \Leftrightarrow \frac{12a}{12} + \frac{6a}{12} + \frac{4a}{12} + \frac{3a}{12} = 6000$
  $\Leftrightarrow \frac{25}{12}a = 6000 \Leftrightarrow a = 2880$
- $\Rightarrow b = 1440,\ c = 960,\ d = 720$

Outline

Task 7 (VB Code, γ code)

Calculate the variable byte code and the gamma code for 217.

- VB:
  - $217_{10} = 11011001_2$
  - Two bytes needed:

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

- γ code:
  - Offset: 101 1001
  - Length: $7_{10} = 11111110_1$
  - γ code: 111111101011001

Outline

Task 8 (γ code)

From the following sequence of γ-coded gaps, reconstruct first the gap sequence and then the postings sequence:

- 111101000011111010101111000
- First length: $11110_1 = 4_{10}$; 1000: $11000_2 = 24_{10}$
- Next length: $0_1 = 0_{10}$; : $1_2 = 1_{10}$
- Next length: $111110_1 = 5_{10}$; 10101 : $110101_2 = 53_{10}$
- Next length: $110_1 = 2_{10}$; 00 : $100_2 = 4_{10}$
- Gap Sequence: $24 \rightarrow 1 \rightarrow 53 \rightarrow 4$
- Doc Sequence: $24 \rightarrow 25 \rightarrow 78 \rightarrow 82$

Overview

1. Task 1 (Jaccard)
2. Task 2 (TF·IDF) (1)
3. Task 3 (Distributed Indexing)
4. Task 4 (Distributed Logarithmic Merging)
5. Task 5 (Heaps' Law)
6. Task 6 (Zipf's Law)
7. Task 7 (VB Code, γ code)
8. Task 8 (γ code)