# IR tutorial

Oct 4th, 2021

# Topic I: Index Construction

## Exercise 4.6

Total index construction time in blocked sort-based indexing is broken down in Table 4.3. Fill out the time column of the table for Reuters-RCV1 assuming a system with the parameters given in Table 4.1.

| | |
|---|---|
| 1 | reading of collection (line 4) |
| 2 | 10 initial sorts of $10^7$ records each (line 5) |
| 3 | writing of 10 blocks (line 6) |
| 4 | total disk transfer time for merging (line 7) |
| 5 | time of actual merging (line 7) |

We mention the required system informations as well as important information for Reuters data in next page.

► **Table 4.1** Typical system parameters in 2007. The seek time is the time needed to position the disk head in a new position. The transfer time per byte is the rate of transfer from disk to memory when the head is in the right position.

| Symbol | Statistic | Value |
|---|---|---|
| $s$ | average seek time | $5\ \text{ms} = 5 \times 10^{-3}\ \text{s}$ |
| $b$ | transfer time per byte | $0.02\ \mu\text{s} = 2 \times 10^{-8}\ \text{s}$ |
| | processor's clock rate | $10^9\ \text{s}^{-1}$ |
| $p$ | lowlevel operation (e.g., compare & swap a word) | $0.01\ \mu\text{s} = 10^{-8}\ \text{s}$ |
| | size of main memory | several GB |
| | size of disk space | 1 TB or more |

**Reuters data**

Number of Document: 8,00,000
Number of token per doc: 200
Number of terms: 4,00,000
Number of tokens : 100,000,000
Byte per token: 6
Byte per term id / doc id: 4

Also assume we have 10 blocks, from where we read the data. Also we save postings lists into 10 blocks, each block having $10^7$ postings each. Assume number of postings equals number of tokens.

1)Reading the collection from disk = Number of token * Average bytes per token * Time to transfer each byte.+ disk seek time
2)Total sorting time = Number of blocks * (N log N) * time for low level operation,
N = Number of records in each block
3) Total writing time for writing sorted blocks back to disk = Number of blocks * time for writing single block
Time for writing single block = (Number of terms per block + Number of posting in each block ) * (size of term id / doc id) * time for transferring 1 byte data to disk
4) Disk transfer time for merging:
Assume we open a read buffer for each of the 10 blocks. Each buffer contains 1/10 th of the block's content.
Disk transfer time for merging = disk seek time + time for reading part of sorted posix lists from disk + time for writing final index
disk seek time = Number of blocks * Number of time required to read full content of the block given we can save only one-tenth of the block * disk seek time  * 2 (because we need 2 disk seeks)
total time for reading posix lists from disk = total time for writing final index = (total number of terms  + total number of posting across all block ) * (size of term id / doc id) * time for transferring single byte
5) Time for actual merging = disk transfer time for merging + processing time for merging.
Processing time for merging = time for low-level operations * number of low level operations
Number of low level operations = O(total number of posting) = total number of terms

**Exercise 4.3**

For $n = 15$ splits, $r = 10$ segments, and $j = 3$ term partitions, how long would distributed index creation take for Reuters-RCV1 in a MapReduce architecture? Base your assumptions about cluster machines on Table 4.1.

► **Table 4.1** Typical system parameters in 2007. The seek time is the time needed to position the disk head in a new position. The transfer time per byte is the rate of transfer from disk to memory when the head is in the right position.

| Symbol | Statistic | Value |
|---|---|---|
| $s$ | average seek time | $5\text{ ms} = 5 \times 10^{-3}$ s |
| $b$ | transfer time per byte | $0.02\ \mu\text{s} = 2 \times 10^{-8}$ s |
| | processor's clock rate | $10^9\ \text{s}^{-1}$ |
| $p$ | lowlevel operation | |
| | (e.g., compare & swap a word) | $0.01\ \mu\text{s} = 10^{-8}$ s |
| | size of main memory | several GB |
| | size of disk space | 1 TB or more |

Map reduce:

Map phase:

Time spent by a machine = Data in each splits in terms of bytes * (time required for reading each byte + time required for performing low level operation on each byte)

As 15 splits are there, map phase will be executed in two phases taking twice of the time spent by each machine.

Reduce phase:

Let we have 100 million postings, so 100/3 million posting per inverter.

Also number of terms per inverter = 4,00,000 / 3

For each inverter,
time in reading = (Number of terms per inverter + Number of posting per inverter)* (size of term id / doc_id in bytes)*transfer time for each byte.
time in sorting = (N log N)* time for low level operations, where N = Number of posting per inverter
time in writing = time in reading = (Number of terms per inverter + Number of posting per inverter)* (size of term id / doc_id in bytes)*transfer time for each byte.
total time in reduce phase = time for reading + time for writing + time for sorting.

# Topic II: Evaluation

# MAP

The following list of Rs and Ns represents relevant (R) and nonrelevant (N) returned documents in a ranked list of 20 documents retrieved in response to a query from a collection of 10,000 documents. The top of the ranked list (the document the system thinks is most likely to be relevant) is on the left of the list. This list shows 6 relevant documents. Assume that there are 8 relevant documents in total in the collection.

R R N N N  N N N R N  R N N N R  N N N N R

Q1. Assuming these 20 documents  to be complete set of result find out the MAP.

Q2. Assuming the system returns all 10,000 documents as results, find out the maximum and minimum possible MAP.

# Solution

Solution 1: ⅛ * (1+1+3/9+4/11+5/15+6/20) = 0.555

Solution 2 a: 1/8 * (1+1+3/9+4/11+5/15+6/20+7/21 + 8/22) = 0.503

Maximum MAP will be attained if both the relevant items are positioned right after the set of documents already shown i.e., at positions 21 and 22.

Solution 2 b: 1/8* (1+1+3/9+4/11+5/15+6/20+7/9999 + 8/10000) = 0.416

Minimum MAP will be attained if both the relevant items are positioned right at the end of all results i.e., at positions 9999 and 10000.

NOTE-- PRECISION WILL BE CALCULATED AT ALL POINTS OF RECALL, i.e., POSITIONS WHERE RELEVANT DOCUMENTS APPEAR.

# Kappa measure example

400 documents were marked Relevant / Non-relevant by 2 annotators. Table below shows the number of documents for all the four possible combinations of their decisions. Evaluate the Kappa between annotator 1 and annotator 2.

| **Number of documents** | **Annotator 1** | **Annotator 2** |
|---|---|---|
| 300 | Relevant | Relevant |
| 70 | Nonrelevant | Nonrelevant |
| 20 | Relevant | Nonrelevant |
| 10 | Nonrelevant | Relevant |

# Solution...

- Pr(a) = P(Agreement) = 370/400 = 0.925
- P(Non-relevant) = (10+20+70+70)/800 = 0.2125
- P(Relevant) = (10+20+300+300)/800 = 0.7878
- Pr (e) = P(Agree Chance) = 0.2125^2 + 0.7878^2 = 0.665
- **Kappa = (0.925 – 0.665)/(1-0.665) = 0.776**

| | Annotator 1: R | Annotator 1: N |
|---|---|---|
| Annotator 2: R | 300 | 10 |
| Annotator 2: N | 20 | 70 |

| Number of documents | Annotator 1 | Annotator 2 |
|---|---|---|
| 300 | Relevant | Relevant |
| 70 | Nonrelevant | Nonrelevant |
| 20 | Relevant | Nonrelevant |
| 10 | Nonrelevant | Relevant |

$$\kappa = \frac{\Pr(a) - \Pr(e)}{1 - \Pr(e)}$$

# Thank you