Analyse

- **Computer specifications I have Used:**
  - Operating System: Windows 10 Enterprise 64-bit
  - CPU speed: Intel® Core ™ i7-6700 CPU @ 3.40GHz - 3.41GHz
  - Memory size: GB 8.00
  - Disk type: SSD
  - Number of processors: 8 processors

- **Time complexity while building the index**

  N = review numbers

  K = processors number

  M = average text length

  L = average word length

  I = size of integer

  R = average number of words starting with specific letter

  A = average appearance of words starting with specific letter

  C = number of reviews in chunk

I give to every processor in the pool task to read a review in chucks according to the memory that the OS gave to the program with consideration to the number of cores in that PC of the reviews file and then we start to process the chunk of data in this way, (if the chunk was too small to handle a single review, then an exception will be thrown).

The main process makes sure that each chunk starts and ends with a review, then the main process adds the task to the pool, as soon as the main process create a task and add it to the pool, the waiting the process starts to run their algorithm, which is as follow:

1. process the data in a way that separate the reviews by productID to create sub-index dictionary and sub analyzed dictionary and then split the text of the review and the details (score, length, etc.).

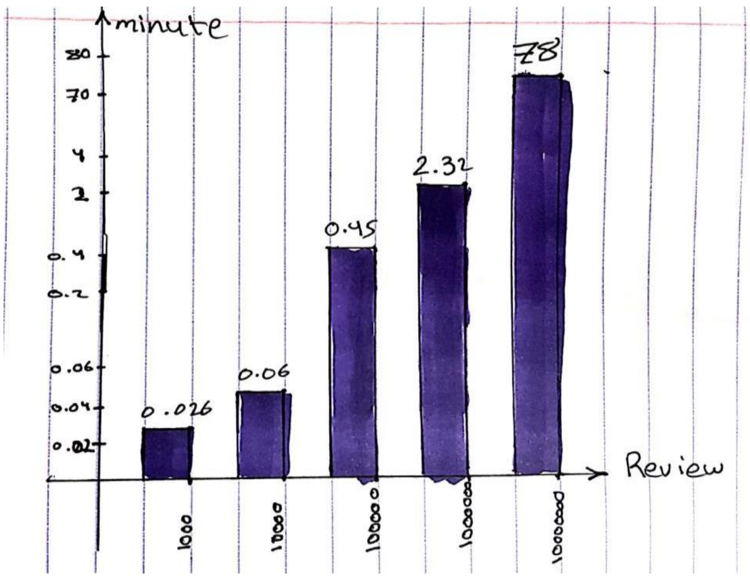
2. built the analyzed file (that contain the review no., productID, score, helpfulness, text length) ➔ O(C)

3. build a non-sorted index file in the process with the help, of the sub-index dictionary that was built in the process, mutex was used because each process writes on the sub-index file that he builds from the reviews he processed.

4. To build the index file I gave to every processor a mission to search for a word that start with a letter (from the alphabetic letters) and meanwhile write in whose reviews contain this word and how many times are contained in it and whenever a chunk is finished, he goes to write his data to a huge file. ➔ O(C)

5. Then we create another pool with other tasks, these tasks are to give the processes an ASCII value (starting with 97 which is 'a'). each process search in the non-sorted index file for all words start with that letter (from the alphabetic letters) and save all the line and sort everything, if duplicates were founded, then we merge the two file, otherwise we write to a directory of that letter all the sorted index when we finished building this huge index file we gave the mission to every processor to build smaller sorted indexes in a specific letter. ➔ $O((\frac{R*A}{S})*(K*n(1+\log n))$

Time complexity of hole algorithm is $O(\frac{N}{K}+\frac{R*A}{S})*(K*n(1+\log n))$



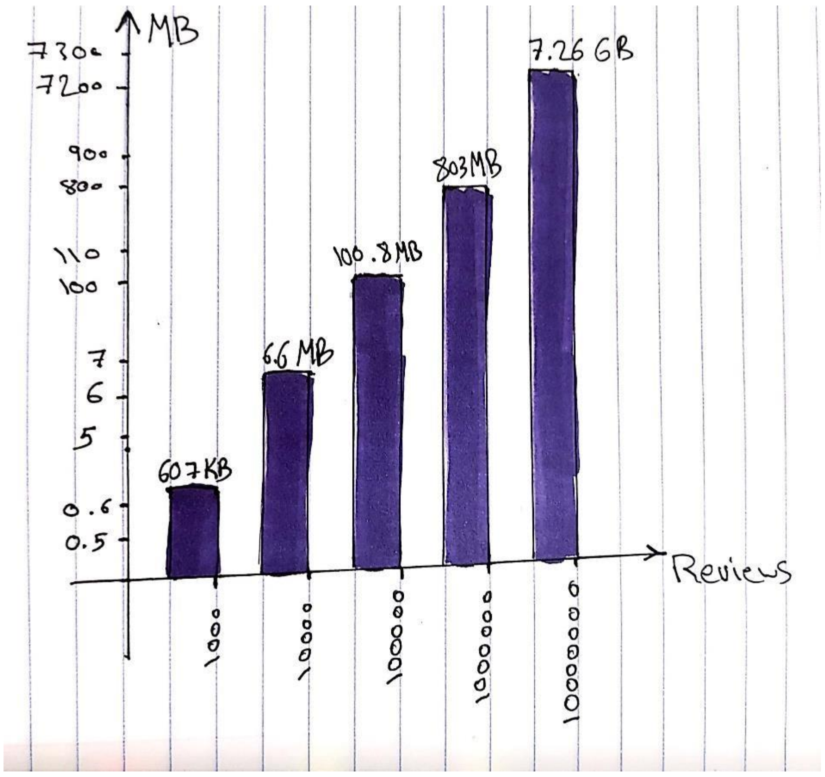
The size of the index on the disk:

- First file (analyser) as follow :
    - Review no: 4 bytes
    - productid: 10 bytes
    - score: 1 byte
    - helpfulness: 2 * (4 bytes)
    - text length: 4 bytes
    - \n : 1 byte

  Sum_0 = 28 + 0 (padding) = 28 bytes

  So for a file which has 10M review, then the size of this file will be 267.028 MB.
- Then 27 files → 26 for all the letters and 1 for all numbers → size as follow:
    - (processors number) * 2 * (size of integer) [The number of reviews and the number of impressions of the word] * (average text length) * (average word length) = L * M * 2I * K → its almost equal to 7GB

So, the size of the index will be → 7GB + 267.028MB = 7.27 G


MB
607KB
6.6 MB
100.8MB
803MB
7.26 GB
Reviews