Advanced Algorithms 3º Project
Most Frequent Letters

Renan Ferreira 93168

***Abstract* – This report addresses the third project of the subject Advanced Algorithms of Masters in Informatics Engineering of University of Aveiro. It focuses on obtaining the most frequent letters of a certain set of books in order to investigate and analyse the performance of 3 count algorithms, the exact algorithm, which counts each element with certainty, the approximate algorithm, which contains some randomness in order to downsize the computer resources necessary for the assignment, and a data stream frequent-count algorithm, the Misre & Gries Algorithm.**

I. Introduction

This report addresses the 3rd and final project of the subject Advanced Algorithms, which investigates the performance of 3 algorithms to identify the most frequent letters in a textbook: Exact count Algorithm, Approximate Algorithm, and Misra-Gries Data Stream Algorithm.

Before we are able to execute the mentioned algorithms, we need to pass our textbooks through a processing of removing noisy and irrelevant data.

The first algorithm is the exact count, which uses a table(Dictionary) to store the frequency of each letter in the text, identifying with precision the most frequent letters.

The second one is an Approximate Count Algorithm, that returns approximate values for letters' occurrences, but that does not necessitate too much computer resources as the exact one. In our example, we increment the counter by a probability of $\frac{1}{2^k}$, with k the current value in the counter.

The third algorithm is the Misra-Gries, a data stream frequent count algorithm that identifies the k-1 most frequent letters by keeping k-1 counters.

We will execute those 3 algorithms in our dataset, from Gutenberg Project, and obtain the accuracy of the most frequent letters by each one, and comparing the efficiency of the approximate counter and Misra-Gries counter in comparison with the exact one, to check If it is able to present us with the accurate most frequent letters of each textbook. We also will compare the most frequent letters for the same literary work, *The Portrait of Dorian Grey*, in different languages, in order to check for the similarity.

II. Dataset

The dataset that we will use for the problem execution are literary works from Project Gutenberg. We chose The Portrait of Dorian Gray, in 5 different languages: English, German, Dutch, French, and Finnish. All the textbooks are text-plain encoded in utf8.

III. Text Processing

Before we can pass the textbooks for counting execution, we need to format them to eliminate noisy and irrelevant data present in the text, making it cleaner for a realistic and efficient count analysis.

First, we make sure to remove the header and the footer of each file. They mostly contain metadata about the literary work, and through the header, we can obtain information about the book's language. Then, we remove punctuation.

The third step is to remove stopwords from the textbook. Stopwords are words that have no synthetic relevance and can be removed for language processing. We downloaded stopwords for each language from *nltk* package, which can be increased based on the user intention, especially through output analysis of program *freq_words.py*. During this procedure, we also convert each character to Uppercase.

The fourth and final step converts non-ASCII words to their ASCII counterparts, to standardize the books to common Latin letters.

After the textbook processing, the outputted text obtained contained the following summary:

Language	Initial Size	Final Size	Percentage of Initial Size
English	448599	218796	48.77%
German	514006	263792	51.32%
Dutch	395210	195069	49.36%
French	481228	273087	56.75%
Finnish	479072	307532	64.19%

TABLE I
Textbook processing summary

IV. Count Algorithms

A. Exact Counter

Exact counters are the type of algorithm that gives you a precise frequency of letters in a text. it is highly efficient in situations where it is necessary to have an

exact solution to a count problem. In our case, it is necessary to properly evaluate the other algorithms and to compare the most frequent letters in literary works in different languages.

Even though it returns an exact count of letters, it is necessary to notice it demands a huge amount of memory to store those counters, in comparison to the other algorithms.

B. Approximate Counter

The approximate counter offers an approximate count of letters in a text, using some randomness factor for it. In our case, we used a decreasing probability factor of $\frac{1}{2^k}$, with k the current value in the counter.

Approximate counters can be very efficient in terms of space and memory use, but they cannot offer a precise answer, which can vary depending on the error rate.

C. Misra-Gries Counter

The Misra-Gries algorithm, also known as Frequent-Count, is a well-known method for finding the frequency of items in a data stream. It is a one-pass algorithm, meaning it only needs to process and run through the data stream once.

The variation of the algorithm that we are going to use contains the parameter k, which identifies the (k-1) most frequent elements in a data stream. We achieve this by maintaining (k - 1) counters (one for each of the k-1 most frequent items at a given time) and incrementing the corresponding counter whenever an item is processed. Here, the parameter k controls the quality of the results given. After all, we keep, at most, (k - 1) candidates at the same time, and we don't miss any item with (at least) frequency m / k. By allowing the detection of the (k - 1) most frequent items, the Misra-Gries algorithm with k provides more information about the distribution of elements in the stream.

One of the key advantages of the Misra-Gries algorithm is its efficiency. It can process large data streams in milliseconds, making it suitable for real-time applications. It is also relatively simple to implement, as it only requires a small number of counters and basic arithmetic operations.

However, it is an approximate algorithm, meaning that it provides an estimate of the frequency of items rather than an exact count. This trade-off is necessary to achieve high efficiency, but it may not be suitable for all applications that require precise counts.

V. Experimentation

To execute the counting algorithms and to provide an appropriate analysis of their performances, we must execute the program in the following order:

```
$ python3 textbook_processing.py
$ python3 counter.py
$ python3 analyse.py
```

The first command will do the text processing, the second, the execution of the 3 counting algorithm, and the last will analyse and produce a summary of the count performance.

A. Execution Time

Language	Execution time
English	0.0165
German	0.0189
Dutch	0.0159
French	0.0257
Finnish	0.0192

TABLE II
Exact Count execution time in seconds

Language	Execution time
English	0.1132
German	0.1222
Dutch	0.0996
French	0.1206
Finnish	0.1448

TABLE III
Approximate Count execution time in seconds (Average for 100 trials)

Language	Execution time			
	K=3	K=5	K=10	Mean
English	0.0538	0.0508	0.0464	0.0503
German	0.0547	0.0536	0.0533	0.0539
Dutch	0.0464	0.0393	0.0388	0.0415
French	0.0605	0.0561	0.0549	0.0571
Finnish	0.0583	0.056	0.0709	0.0617

TABLE IV
Misra-Gries Count execution time in seconds

The analysis of the previous tables shows that the execution time is neatly similar to all the algorithms, but the Exact counter and the Misra-Gries Algorithm perform better in comparison to the Approximate Counter. Considering that the data has a fixed size, the additional time on both Approximate and Misra-Gries algorithms can be attributed to the additional instructions in the loop of both.

B. Exact Count

The 10 most frequent letters by each languages are:

- English: E(27813), A(16230), R(15761), O(15203), N(15008), I(14961), S(14860), T(14621), L(12573), D(11593)
- German: E(43916), N(24788), R(19300), T(18340), S(17866), A(17018), I(15999), H(14330), L(13076), G(11011)

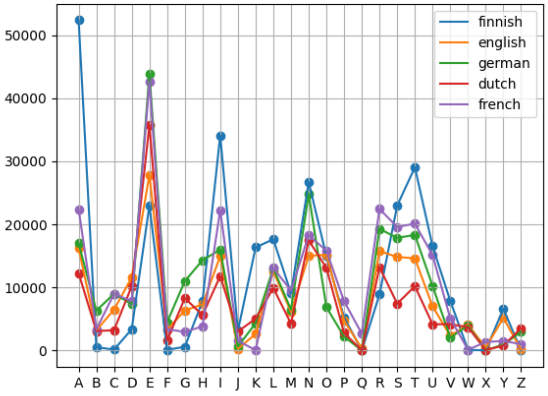

Fig. 1 - Frequency by each language.

- Dutch: E(35755), N(17474), R(13223), O(13120), A(12302), I(11812), T(10210), D(10203), L(9929), G(8321)
- French: E(42648), R(22444), A(22429), I(22238), T(20120), S(19611), N(18291), O(15844), U(15133), L(13103)
- Finnish: A(52377), I(34114), T(29056), N(26705), E(23018), S(22988), L(17648), U(16545), K(16362), O(14959)

The most frequent letter is E in all languages, except Finnish, which is A. Considering the 3 most frequent letters, most languages have either E, A, and R, or E, N, and R, except again finish, which are A, I, T.

By the top 5, E is in all languages, A, N and R are in 4 from 5 languages, T in 3 from 5, I and O in 2, and S in 1, german. Finally, considering all the 10 most frequent letters, the vowels appear A, E, I appear in all the languages, except O and U, which appear in 3 and 2, respectively. The consonants L, N and T are in the 10 most frequent letters in all languages, S and R appear in 4 out of 5, D and G in 2, and H and k in 1.

C. Approximate Counter

Like we said before, our experimentation involved running the approximate counter in 100 trials, and we obtained the following accuracy.

Language	Accuracy			
	Top 1	Top 3	Top 5	Top 10
English	54%	0%	0%	0%
German	59%	3%	0%	0%
Dutch	68%	1%	0%	0%
French	55%	1%	0%	0%
Finnish	63%	3%	0%	0%

TABLE V
APPROXIMATE COUNTER ACCURACY

The accuracy is pretty good for the most frequent letter for each language, but it falls for null or almost null values for the other orders.

D. Misra-Gries Counter

For the Misra-Gries we considered the parameter K as equal to 3, 5, and 10.

Language	Accuracy		
	K=3	K=5	K=10
English	0%	25%	55.56%
German	50%	25%	33.33%
Dutch	0%	25%	66.67%
French	0%	25%	44.44%
Finnish	50%	25%	55.56%

TABLE VI
MISRA-GRIES ACCURACY

For all the languages, the accuracy seems to increase with the increase of the parameter K, but still for not good enough values. It is interesting to note that for all languages, the Misra-Gries algorithm with K=5 gave 25% accuracy.

VI. CONCLUSION

In conclusion, we can see that the approximate count algorithm is very efficient to obtain the most frequent letter, but not a collection of the most frequent ones, which the Misra-Gries Algorithm can be very good at this. Anyways, both algorithms fail to give an approximate count values for all the letters present in the textbooks.