Advanced Algorithms - Third Project
Most Frequent Letters

Gonçalo Machado Nmec 98359

Abstract **– This report studies different methods to identify the most frequent letters in text files. This problem is addressed by the course "Advanced Algorithms" at the University of Aveiro. The proposed approaches are Exact Counters, Approximate Counters, and the Misra-Gries algorithm, also known as Frequent-Count algorithm. Along with their implementation, we evaluate the quality of estimates regarding the exact counts and analyze the computational efficiency and limitations of the developed approaches. The chosen coding language was Python.**

Resumo **–Este relatório apresenta diferentes métodos de identificar as letras mais frequentes em ficheiros de texto. Este problema é abordado pelo curso "Algoritmos Avançados"na Universidade de Aveiro.Os métodos desenvolvidos são Contadores Exatos, Contadores Aproximados e o algoritmo Misra-Gries, também conhecido como algoritmo Frequent-Count .A linguagem de programação escolhida foi Python.**

Keywords **– Most Frequent Letters, Text Processing, Exact Counter, Approximate Counter, Misra-Gries Algorithm, Frequent-Count**

Palavras chave **– Letras Mais Frequentes, Processamento de Texto, Contador Exato, Contador Aproximado, Algoritmo Misra-Gries, Frequent-Count**

I. Introduction

This report is part of the third project of "Advanced Algorithms". The goal of the project was to identify the most frequent letters in text files (literary works) using different methods, and to evaluate the quality of estimates regarding the exact counts.

The following sections describe what each of the three methods used (Exact Counters, Approximate Counters, and the Misra-Gries algorithm, also known as Frequent-Count algorithm) are and how they work.

In each section pertaining a different approach, an analysis of the computation efficiency and limitations of the developed approach will be done in terms of absolute and relative errors, average values, among other relevant parameters.

A comparison with other relevant approaches will also be done to see if the same most frequent letters were identified and in the same relative order.We will also see if the most frequent letters are similar in text files of the same literary work, but in different languages.

Alongside the report there were also the files containing code for each of the algorithms developed, the code used for comparing the algorithms, the text files used and the results that were obtained from running the algorithms and performing the comparison between them. In order to run the code that generates the counters, the following commands should be used:

$ python exact_counter.py
$ python approximate_counter.py
$ python frequent_counter.py

For generating the files containing the comparison between counter, the following command should be used:

$ python compare_counters.py

II. Text Processing

For this project, as explained before, we were asked to identify the most frequent letters in text files of literary works, and to see if the same letters were the most frequent in different languages. For this reason, we picked three text files containing the literary work "The Odyssey" by Homer in English, Spanish and French. The files were obtained from the Project Gutenberg website. Project Gutenberg is a volunteer effort to digitize and archive cultural works, as well as to encourage the creation and distribution of eBooks.

Before the files could be used, they needed to be processed. The first step was to remove extra text in the text files so that only the literary work would be present. This meant removing the **Project Gutenberg** file headers (which are added to the beginning of each ebook to provide additional information about the book), but also removing text that was present in some languages but not others (such as prefaces or footnotes).

Although the file headers are easily spotted due to some keywords, the text that existed in some languages but not all were not as easy to remove. To remove them, we used some of the words in the first sentence of the literary work to mark the beginning and some words from the last sentence to mark the end, and removed everything that was not between the markers.

The next step of the processing was to remove all stop words and punctuation, as well as some keywords that were an indication of illustrations, thus not belonging to the literary work text. Stop words are common words that do not contribute much meaning to the text, such as articles, conjunctions, and prepositions. The stop words of each language were obtained from the **CountWordsFree** website. **CountWordsFree**

is a web-based service for content writers, web developers, and professionals who provide search ranking optimization services.

The final step was to remove all numbers and non-alphabetic characters, leaving only the letters of the alphabet, and converting every letter to uppercase, to ensure that the counters do not count a lower case letter as being different from the same letter in upper case.

Text Length Before and After Processing

Language	Initial Length	Final Length	Difference
English	698069	216616	481453
Spanish	1001728	367748	633980
French	668432	330184	338248

As we can see, after the processing of the literary works, the length was cut in more than half, with the length of the Spanish language text file ending up with a third of the original length.

III. Exact Counter

In the world of text analysis, exact counters, also known as exact frequency counters, play a vital role in quantifying the occurrences of letters within a text file. These tools are essential for revealing linguistic patterns and extracting valuable insights. Exact frequency counters are valuable for understanding the distribution of letters in a given text. They find applications in linguistic studies, language modeling, and cryptographic tasks.

Exact counters excel in accurately counting individual letter occurrences, allowing for nuanced linguistic analysis and providing insights into the structural aspects of language through the examination of letter frequencies. Researchers leverage the versatility of exact frequency counters not just for linguistic exploration but also as effective tools for deciphering encrypted messages using frequency analysis techniques.

However, it's worth noting that exact counters can be slower and require more memory to store the count of each item. It is a limitation that becomes more significant if the dataset is large or if the processing speed is a concern.

After executing the code that contains the exact counter algorithm for each of the three text files, we were able to identify the most frequent letters in each language:

1. English: 'E': 28671, 'S': 19490, 'A': 16673, 'R': 15432, 'N': 14411, 'O': 13811
2. Spanish: 'A': 44356, 'E': 42152, 'O': 31916, 'R': 28115, 'S': 27850, 'I': 23203
3. French: 'E': 41711, 'S': 28511, 'R': 27794, 'A': 25607, 'I': 25110, 'N': 22881

IV. Approximate Counter

Approximate counters are specialized data structures designed to provide an estimated count of item frequencies within a dataset. These counters prioritize efficiency over absolute accuracy, enabling them to process large datasets more swiftly than exact counters.

In the context of identifying letters in a text file, we can utilize approximate counters to gauge the frequency of each letter. This method serves practical purposes similar to exact counters, such as analyzing letter distributions and identifying common patterns in written communication. To implement an approximate counter for this task, we create a counter for each letter and increment the corresponding counter with a given probability when encountering a letter, which for this project was 1/4, or 25%. This meant that the counter would increment once every 4 encounters, approximately.

One advantage of using approximate counters is their faster processing speed and lower memory requirements compared to exact counters. However, it's important to note that approximate counters trade off some accuracy, providing an estimate of item frequencies rather than an exact count. This may be less suitable for applications requiring precise counts.

The following results were obtaining after using the approximate counters in 20000 trials.

Absolute and Relative Errors For English Version

	English
Expected value	54154.0
Variance	27077.0
Standard deviation	164.55090397806995
Mean absolute error	161.37945
Mean relative error	0.2980009786904006%
Mean accuracy ratio	100.00171797097168%
Smallest counter value	53244
Largest counter value	54921
Mean counter value	54154.93035
Mean absolute deviation	161.38131070000023
Standard deviation	202.20370990384353
Maximum deviation	910.9303500000024
Variance	40886.34029887771

Absolute and Relative Errors For Spanish Version

	Spanish
Expected value	91937.0
Variance	45968.5
Standard deviation	214.4026585656064
Mean absolute error	208.94865
Mean relative error	0.22727373092444111%
Mean accuracy ratio	100.0020301402047%
Smallest counter value	90879
Largest counter value	92936
Mean counter value	91938.86645
Mean absolute deviation	208.93907735999977
Standard deviation	262.5453762959799
Maximum deviation	1059.8664500000014
Variance	68930.07461439769

Absolute and Relative Errors For French Version

	French
Expected value	82546.0
Variance	41273.0
Standard deviation	203.15757431117353
Mean absolute error	197.9248
Mean relative error	0.23977515567077662%
Mean accuracy ratio	100.00142647735808%
Smallest counter value	81476
Largest counter value	83517
Mean counter value	82547.1775
Mean absolute deviation	197.91635449999842
Standard deviation	247.7787363632118
Maximum deviation	1071.1775000000052
Variance	61394.30219375002

Absolute and Relative Errors For Letter E (English)

	English
Expected value	7167.75
Variance	3583.875
Standard deviation	59.86547419005382
Mean absolute error	58.276225
Mean relative error	0.8130337274598115%
Mean accuracy ratio	99.98529873391232%
Smallest counter value	6894
Largest counter value	7470
Mean counter value	7166.69625
Mean absolute deviation	58.27169162499992
Standard deviation	73.11573897552825
Maximum deviation	303.30375000000004
Variance	5345.91128593758

Absolute and Relative Errors For Letter A (Spanish)

	English
Expected value	11089.0
Variance	5544.5
Standard deviation	74.46139939592862
Mean absolute error	72.92875
Mean relative error	0.657667508341596%
Mean accuracy ratio	100.00761971322933%
Smallest counter value	10703
Largest counter value	11423
Mean counter value	11089.84495
Mean absolute deviation	72.9274825749998
Standard deviation	91.34607823818999
Maximum deviation	386.8449500000006
Variance	8344.106009497526

The tables show statistical values for the different language versions (English, German, French). These values include measures of central tendency (mean, expected value), dispersion (variance, standard deviation, mean absolute deviation), and accuracy (mean absolute error, mean relative error, mean accuracy ratio).

Absolute and Relative Errors For Letter E (French)

	English
Expected value	10427.75
Variance	5213.875
Standard deviation	72.207167234285
Mean absolute error	70.12445
Mean relative error	0.6724792021289306%
Mean accuracy ratio	99.99845268634174%
Smallest counter value	10101
Largest counter value	10826
Mean counter value	10427.58865
Mean absolute deviation	70.12432091999963
Standard deviation	88.04118150716432
Maximum deviation	398.4113500000003
Variance	7751.249641177455

Accuracyfor Different Languages

Language (Accuracy)	English	Spanish	French
Letter Order	20.18%	0%	0.42%
Top 3 Letter Order	100.0%	100.0%	96.05%
Top 1 Letter	100.0%	100.0%	100.0%

In general, the results are that despite the randomness inherent of the approximate counters, the accuracy is extremely high in all languages, as proven by the mean relative error and mean absolute error, with the mean relative error being less than 1% in all languages.

As for identifying the order of the most frequent letters, we see that the algorithm has an 100% accuracy identifying the most frequent letter in all languages, and 100% accuracy identifying the three most frequent letter in order in nearly all languages, French being the exception. This exception is due to the fact that the second and third most frequent letters are really close in terms of counters, which makes it more vulnerable to the randomness of the algorithm.

The results were not as promising when identifying the order of the most frequent letters, with the English language having a 20% accuracy and the other two languages 0% accuracy. Although it is clearly possible to see that this algorithm is not good for identifying the order of most frequent letters, an explanation for the Spanish and French languages accuracy exists.

These languages have letters with accents and can include letters of other alphabets other than the Latin alphabet. This letters were a very small number of times when compared with the Latin alphabet letters and are very close between them in terms of counters, which makes them even more vulnerable to the randomness of the algorithm.

V. Misra-Gries Algorithm (Frequent-Count)

The Misra-Gries algorithm, also recognized as Frequent-Count, stands as a renowned approach for determining item frequencies in a data stream. Operating as a one-pass algorithm, it efficiently processes the data stream in a single run. Simultaneously, it

remains a straightforward yet effective algorithm applicable across diverse fields, such as natural language processing and database indexing.

A variant of the Misra-Gries algorithm is the version with a parameter k, targeting the identification of the (k-1) most frequent items in a data stream. This involves managing (k - 1) counters, each corresponding to the k-1 most frequent items at any given time. Incrementing the relevant counter upon processing an item is the core mechanism. The parameter k governs result accuracy, maintaining, at most, (k - 1) candidates simultaneously and ensuring the inclusion of items with a frequency of at least m / k.

By extending its capability to detect the (k - 1) most frequent items, the Misra-Gries algorithm with k offers enhanced insights into the element distribution in the stream.

The Misra-Gries algorithm excels in efficiency, processing large data streams within milliseconds, making it well-suited for real-time applications. Its implementation is straightforward, requiring a minimal number of counters and basic arithmetic operations.

Nonetheless, the Misra-Gries algorithm has its limitations. Being an approximate algorithm, it furnishes frequency estimates rather than precise counts. While this trade-off is crucial for achieving heightened efficiency, it may not be optimal for applications requiring exact counts.

For this project, we use this algorithm with k = 3, k = 5 and k = 10.

Accuracy for the Misra-Gries algorithm

Language	k	Correct Letters	Accuracy
English	3	1/2	50.00%
English	5	1/4	25.00%
English	10	4/9	44.44%
Spanish	3	0/2	0.00%
Spanish	5	2/4	50.00%
Spanish	10	6/9	66.67%
French	3	0/2	0.00%
French	5	1/4	25.00%
French	10	6/9	66.67%

As it is possible to see, the accuracy of the algorithm increases as k increases, although it is still low overall. For k = 3, only in the English language it had a non 0% accuracy. For k = 4, every language had at least one correct letter. For k = 10, we see a bigger accuracy, with most of the languages getting more than half the correct letters. This shows, as expected, that the accuracy of the algorithm is directly tied to the k value.

VI. Processing Time

By analysing the tables, we can see that the Frequent Count algorithm was the slowest of the three, with the approximate counter coming in second and the fastest overall being the exact counters. This is likely due to the extra operations that are done in the approximate counters (the randomness aspect) and the Misra-Gries algorithm (decreasing all the counters and deleting if count reaches 0).

Exact Counters Time

Language	Time
English	0.004995107650756836
Spanish	0.008999109268188477
French	0.010001182556152344

Approximate Counters (Average) Time

Language	Time
English	0.03644352322816849
Spanish	0.06214751232862473
French	0.05643431528806686

The Frequent Count Algorithm Time

Language	k	Time
English	3	0.09251284599304199
English	5	0.09551620483398438
English	10	0.08251547813415527
Spanish	3	0.15652132034301758
Spanish	5	0.1650235652923584
Spanish	10	0.136519193649292
French	3	0.1420297622680664
French	5	0.1465141773223877
French	10	0.12402200698852539

VII. Conclusion

In general, the choice between using an exact counter, an approximate counter, or the Misra-Gries algorithm depends on the specific requirements of the task at hand. If precise counts are necessary, exact counters are the better choice. However, if memory efficiency is more important, approximate counters or the Misra-Gries algorithm may be the better choice.

The Exact Counter was the best overall, in terms of speed and accuracy. Its simplicity and low operations allowed it to be superior than the other algorithms.

The Approximate Counter were very capable of getting the three most frequent letters in their relative order, and the results were very accurate, but it was slower than the exact counter and had very low accuracy when obtaining the order of the most frequent letters.

The Misra-Gries algorithm was generally the worst, both in terms of time and in terms of accuracy, although if a higher k is used the results would be better and it might make the algorithm worth choosing over other options.

References

[1] Wikipedia (2024). *Approximate Counting Algorithm* [Online]. Available: https://en.wikipedia.org/wiki/Approximate_counting_algorithm

[2] Wikipedia (2024) *Misra–Gries summary* [Online]. Available: https://en.wikipedia.org/wiki/Misra%E2%80%93Gries_summary

[3] Wikipedia (2024) *Stop words* [Online]. Available: `https://countwordsfree.com/stopwords`

[4] Gutenberg *Project Gutenberg* [Online]. Available: `https://www.gutenberg.org/`