Advanced Algorithms Assignment 2

Diogo Bento 93391

***Resumo* – O presente artigo detalha um estudo do uso de contadores probabilísticos no contexto da Unidade Curricular de Algoritmos Avançados da Universidade de Aveiro.**

Este artigo faz uso de contadores com probabilidade fixa e dinâmica e analisa os resultados obtidos, que serão comparados aos valores obtidos por um contador determinístico.

***Abstract* – This paper details a study of the use of probabilistic counters, tackled within the context of the Advanced Algorithms Curricular Unit of Universidade de Aveiro.**

This article uses both fixed and dynamic probability counters and analyses the results obtained, using values obtained by a deterministic counter as a benchmark.

I. INTRODUCTION

In software counting event occurrences is commonplace.

Standard procedure is to increment the event's matching variable by 1 whenever the event occurs, but other alternatives may be appealing based on context.

Probabilistic counters have a non-100% chance to increment their variable, leading to approximate counter values that tend to occupy less memory and be quicker to process.

The chance to incremement the counter can either be **fixed** at a certain threshold or vary **dynamic**ally based on the counter's value.

II. ALGORITHMS

A. *Fixed Probability Counter*

We will be using a 1/16 chance to increment our counter, meaning that it will be incremented approximately 6.25% of the time and house values roughly 16 times smaller than the real ones.

In order to get approximate real values one just has to multiply the counter's values by 16.

B. *Dynamic Probability Counter*

We will be using a $2^{-\frac{k}{2}}$ chance to increment our counter, with k being the counter's current value.

This means that the counter's increment chance goes down exponentially.

Associating the obtained value to an approximation of the real one is trickier when taking this approach.

The counter's value is expected to be k after n events, with n being

$$n = 2^{\frac{k}{2}}\text{-}1$$

Which can be inverted to

$$k = 2 * \log_2(n+1)$$

Should we choose to analyze accuracy by downscaling data rather than upscaling it.

III. ANALYSIS

A. *On obtaining data and visualizations*

In order to obtain data the German, French, and English versions of 'A Christmas Carol' were downloaded from Project Gutenberg.

The texts were then converted to uppercase, had every non-alphabetic character removed, and had their headers and footers manually removed.

Both probabilistic counting processes were performed 1000 times for each file and the results were dumped into a text file for later analysis along with the values obtained from performing deterministic counting.

The values obtained were then upscaled via the inverse formulae described previously and compared to the real ones.

For each language,counter type, and a given sample size ranging from 1 to 1000 the minimum, maximum, and average deviation were calculated, both in relative and absolute terms.

Additionally the letters keys of each counter type were sorted based on their matching value and then a tally was taken of the number of item swaps required for a sorted array to match the real order.

Finally, given the deterministic values for each language we plotted the a bar chart comparing the frequencies of each letter in each language.

The obtained results can be found at *results.json* and were obtained by running *counters.py*.

The statistical calculations and plotting code are contained in *analysis.py* and running the program with the --**interactive** flag allows a user to interact with graphs rather than just dump them into an image file.

B. Results

For brevity's sake only a few of the generated graphs will be displayed.
All graphs will be provided in the /graphs directory.

A. Fixed Probability Counter

All texts saw the average deviation tend towards 0 as sample size increased.
English has significantly lower maximum deviation than the other texts (1800 vs 2500).
We are uncertain whether this is caused by a smaller alphabet, random chance, or another factor.

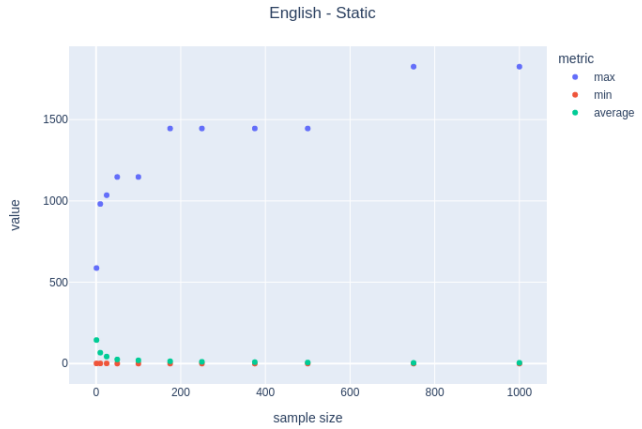

Fig. 3 – Static Counter Value Deviation

Likewise all relative deviations tend towards 0%.

Maximum percentual deviation varies from 1500% to 600%.

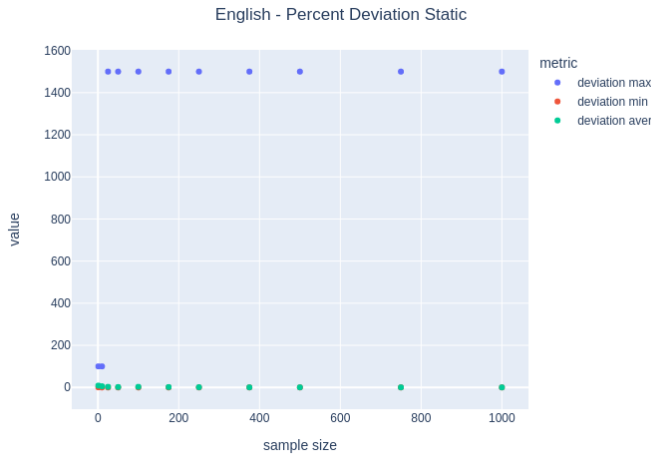

Fig. 4 -Static Counter Value % Deviation

B. Dynamic Probability Counter

All texts obtained roughly the same average and minimum deviation.
English has significantly lower maximum deviation than the other texts (12000 vs 20000), but it's unsure whether this is caused by a smaller alphabet, random chance, or another factor.

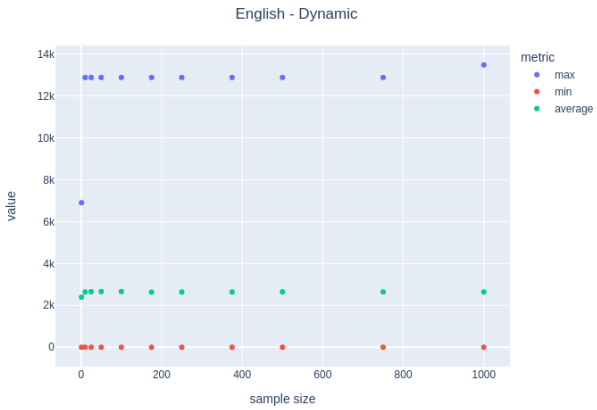

Fig. 3 – Dynamic Counter Value Deviation

For all languages the average relation deviation was around 60%, with the minimum approaching 0.

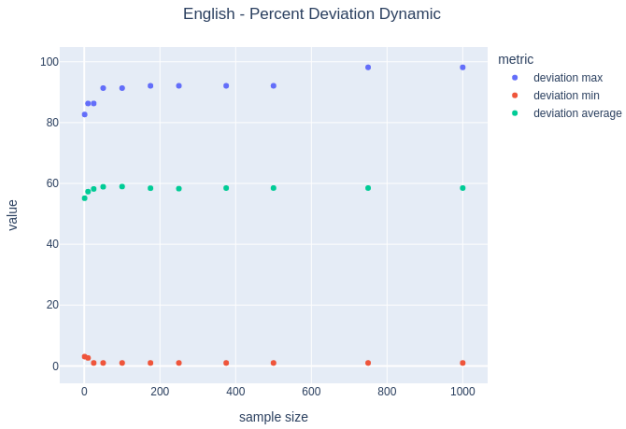

Fig. 4 - Dynamic Counter Value % Deviation

C. *Swaps*

The number of swaps required tends towards 0 as sample size grows, regardless of language.

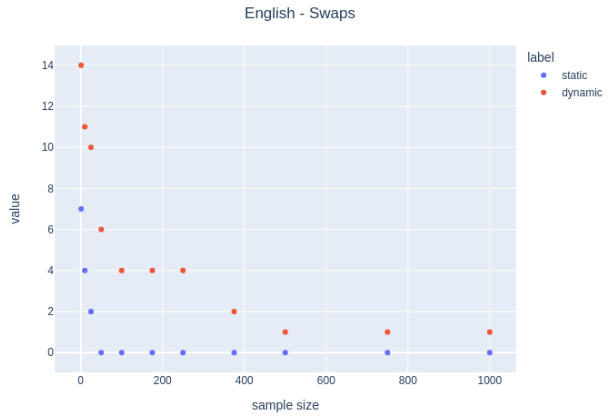

Fig. 5 - Swaps Performed

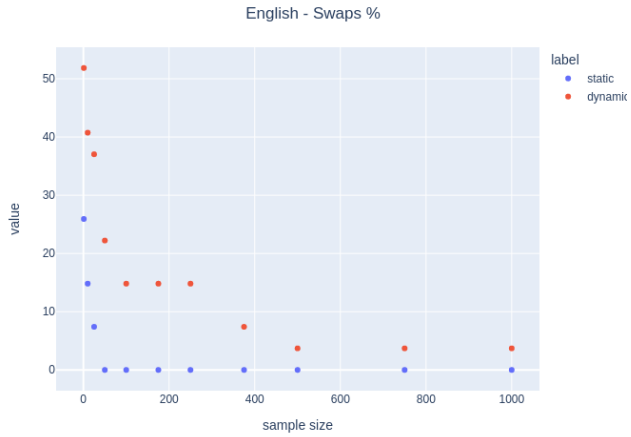

Fig. 6 - Swaps performed relative to alphabet size

D. *Letter Distribution*

Using the real counter values we can see that these 3 texts have different letter frequency distributions.

For example, the French text uses *U* much more often than the others and almost never uses *H*, and the German text uses the letter *E* 25000 times, a full 10000 more than English does.