Contagem Aproximada de Ocorrências – Cadeias de Caracteres Aleatórios

João Ferreira

***Resumo* – Este projeto foi desenvolvido de modo a contar o número de caracteres numa determinada sequência utilizando três tipos de contadores: contador exacto, contador com probabilidade fixa e contador com probabilidade decrescente logarítmica. As bases utilizadas para os últimos dois contadores foram 1/8 e raiz quadrada de 2, respetivamente. Inicialmente, foi feita a contagem de todos os caracteres distintos nas sequências geradas aleatoriamente e, posterimente, foram comparados os resultados do máximo erro relativo, mínimo erro relativo, média do erro relativo, desvio máximo, desvio médio absoluto, desvio padrão e variância.**

***Abstract* - This project was developed in order to count the number of characters in a given sequence using three types of counters: exact counter, counter with fixed probability and counter with logarithmic decreasing probability. The bases used for the last two counters were 1/8 and square root of 2, respectively. Initially, all the different characters in the randomly generated sequences were counted and, later, the results of the maximum relative error, minimum relative error, average of the relative error, maximum deviation, absolute mean deviation, standard deviation and variance were compared.**

I. Introdução

O projeto desenvolvido consiste na contagem de caracteres distintos numa sequência aleatória utilizando três contadores distintos: um contador exato, um contador com probabilidade fixa e um contador logarítmico. Pretende-se criar métodos que implementem estes contadores, utilizando a base 1/8 para o contador de probabilidade fixa e a base raiz quadrada de 2 para o contador logarítmico. As cadeias de texto utilizadas para as contagens são geradas escolhendo aleatoriamente caracteres da cadeia de caracteres que representa o meu nome completo – João Guilherme Mendonça Pimenta de Oliveira Ferreira – em que todos os caracteres são minúsculos e sem espaços. Para atingir os resultados pretendidos foram analisados os parâmetros sugeridos no enunciado (máximo erro relativo, mínimo erro relativo e média do erro relativo) e também parâmetros que considerei relevantes (desvio máximo, desvio médio absoluto, desvio padrão e variância). Os primeiros três parâmetros são executados comparando o contador exato com os dois restantes, individualmente. Os últimos quatro parâmetros são calculados individualmente para cada contador.

II. Sequência aleatória

De modo a implementar os contadores pretendidos é necessário gerar uma sequência de caracteres de tamanho fixo. Este processo é bastante simples e facilmente alcançável através de um ciclo *for*. A imagem seguinte evidencia como foi produzida essa sequência.

```
name='joãoguilhermemendoncapimentadeoliveiraferreira'
sequence=''

for i in range(size):
    sequence += random.choice(name)

print("Sequence size: "+str(len(sequence))+" chars \n")
```

Fig.1 – Gerar a sequência aleatória

III. Contador exato

Para a implementação do contador exato começo por criar um dicionário no qual irão ser guardadas as contagens de cada caractere no formato <caractere, contagem>. De seguida, itero sobre a sequência fornecida (caracterere a caractere) e, caso esse caractere ainda não se encontre no dicionário anteriormente criado, é adicionado ao mesmo com um inteiro equivalente a 1 associado. Caso esse caractere já se encontre no dicionário, o valor associado é incrementado uma unidade. A figura seguinte mostra como poderá ser implementado um contador exato.

```
def exact_counter(sequence):
    results = {}
    for char in sequence:
        if char in results:
            results[char]+=1
        else:
            results[char]=1
    return results
```

Fig.2 – Contador exato

IV. CONTADOR COM PROBABILIDADE FIXA

O contador com probabilidade segue a mesma lógica do contador exato, sendo que a diferença se encontra na probabilidade de se incrementar ou não o contador. Como indicado pelo Professor Joaquim Madeira, a probabilidade de incrementar seria igual a ⅛ e, consequentemente, a de não incrementar seria igual a ⅞ (1 – ⅛) . Para implementar esta diferença entre probabilidades optei por utilizar o módulo *random* da linguagem Python que retorna uma lista (neste caso, com um elemento) com o valor 0 ou 1, de forma aleatória. No fim, com o objetivo de obter os valores pretendidos para as estimativas, é necessário multiplicar os valores registados pelo denominador, neste caso, 8. A figura seguinte demonstra como foi implementado este contador.

```
def fixed_prob_counter(sequence):
    results = {}
    for char in sequence:
        if char in results:
            value = random.choices([0,1], weights = [7/8, 1/8])
            results[char]+=value[0]
        else:
            value = random.choices([0,1], weights = [7/8, 1/8])
            results[char]=value[0]
    for r in results:
        results[r]*=8

    return results
```

Fig.3 – Contador com probabilidade fixa

V. CONTADOR LOGARÍTMICO

Este contador é semelhante ao anterior, sendo que, neste caso, a probabilidade de incrementar o contador caso o caractere já tenha sido guardado no dicionário com os resultados seria 1/(*sqrt(2)**n)*, como acordado com o Professor. Logo, a probabilidade de não incrementar seria 1 – 1/(*sqrt(2)**n).*

Caso o caractere ainda não tenha sido guardado no dicionário, a probabilidade de incrementar o contador será igual a *1/(sqrt(2)**0)*, que equivale a 1 (um), ou seja, a primeira vez que o caractere for guardado será sempre com o valor 1.

No fim deste processo é necessário multiplicar os valores por *sqrt(2)**n* para obter os resultados esperados. A figura seguinte demonstra como foi implementado este contador.

```
def logarithmic_counter(sequence):
    results = {}
    for char in sequence:
        if char in results:
            value = random.choices([0,1], weights = [1-1/(math.sqrt(2)**results[char]), 1/(math.sqrt(2)**results[char])])
            results[char]+=value[0]
        else:
            value = random.choices([0,1], weights = [0, 1]) # 1 / math.sqrt(2)**0  = 1
            results[char]=value[0]

    for r in results:
        results[r] *= math.sqrt(2)**results[r]


    return results
```

Fig.4 – Contador com probabilidade decrescente logarítmica

VI. RESULTADOS

Para testar o algoritmo desenvolvido foram utilizadas três sequências de caracteres de tamanhos distintos: 100, 1000 e 10000.

Utilizando a sequência de tamanho 100, os resultados foram os exibidos na figura seguinte.

Exact	Fixed	Ambos
e	e	
o	d	
a	m	
d	j	
n	f	
i	r	
m	o	
l	a	
c	h	
r	t	
v	l	
t	c	
j	g	
g	n	
p	i	
ā	u	
h	p	
f	v	
u	ā	

Fig.5 – Comparação entre contador exato e contador de probabilidade fixa

Entre estes dois contadores a média do erro relativo foi 112.01%, o erro máximo foi 300% e o erro mínimo 14.29%.

Exact	Logarithmic	Ambos
e	e	
o	d	
a	m	
d	j	
n	f	
i	r	
m	o	
l	a	
c	h	
r	t	
v	l	
t	c	
j	g	
g	n	
p	i	
ā	u	
h	p	
f	v	
u	ā	

Fig. 6 – Comparação entre contador exacto e contador logarítmico

Entre estes dois contadores a média do erro relativo foi 114.95%, o erro máximo foi 619.96% e o erro mínimo 0%.

Para além destes três parâmetros foram ainda analisados o desvio máximo, o desvio médio absoluto, o desvio padrão e a variância para cada um dos contadores. Os valores são exibidos na figura seguinte.

		COUNTERS	
	EXACT	FIXED	LOG
MAX DEV	14,74	18,53	66,92
ABS MEAN DEV	2,65	5,76	9,9
STD DEV	4,06	6,89	17,03
VARIANCE	16,51	47,51	289,96

Fig. 7 – Parâmetros individuais de cada contador

De seguida, foram realizados testes para uma sequência com 1000 caracteres. Os resultados e respectivas comparações são exibidos nas figuras seguintes.

Exact	Fixed	Ambos
e	e	
r	r	
i	i	
m	a	
o	n	
a	d	
n	o	
d	u	
l	l	
u	m	
g	j	
f	v	
ā	g	
t	t	
p	p	
h	h	
j	f	
c	ā	
v	c	

Fig. 8 – Comparação entre contador exato e de probabilidade fixa

Entre estes dois contadores a média do erro relativo foi 30.08%, o erro máximo foi 88.24% e o erro mínimo 3.7%.

Exact	Logarithmic	Ambos
e	e	
r	m	
i	r	
m	n	
o	a	
a	i	
n	t	
d	d	
l	l	
u	g	
g	o	
f	u	
ā	p	
t	f	
p	c	
h	v	
j	h	
c	j	
v	ā	

Fig. 9 – Comparação entre contador exato e contador logarítmico

Entre estes dois contadores a média do erro relativo foi 264.86%, o erro máximo foi 954.12% e o erro mínimo foi 4.21%.

Para além destes três parâmetros foram ainda analisados o desvio máximo, o desvio médio absoluto, o desvio padrão e a variância para cada um dos contadores. Os valores são exibidos na figura seguinte.

		COUNTERS	
	EXACT	FIXED	LOG
MAX DEV	117,37	163,79	1531,05
ABS MEAN DEV	34,27	163,79	1531,05
STD DEV	41,79	49,8	411,98
VARIANCE	1746,13	2474,9	169731,07

Fig. 10 – Parâmetros individuais de cada contador

Finalmente, foram realizados testes para uma sequência com 10 000 caracteres. Os resultados obtidos para esta sequência foram os exibidos na figura seguinte.

Exact	Fixed	Ambos
e	e	
i	i	
r	r	
a	o	
o	a	
m	m	
n	n	
e	d	
d	l	
g	v	
f	j	
p	f	
c	c	
ā	t	
u	ā	
v	p	
t	g	
h	h	
j	u	

Fig. 11 – Comparação entre contador exato e contador de probabilidade fixa

Entre estes dois contadores a média do erro relativo foi 11.11%, o erro máximo foi 38.32% e o erro mínimo foi 0.35%.

Exact	Logarithmic	Ambos
e	e	
i	i	
r	r	
a	o	
o	n	
m	m	
n	l	
e	d	
d	h	
g	ā	
f	f	
p	j	
c	t	
ā	c	
u	a	
v	u	
t	v	
h	g	
j	p	

Fig.12 – Comparação entre contador exato e contador logarítmico

Entre estes dois contadores a média do erro relativo foi 417.16%, o erro máximo foi 787.01% e o erro mínimo foi 35.71%.

Para além destes três parâmetros foram ainda analisados o desvio máximo, o desvio médio absoluto, o desvio padrão e a variância para cada um dos contadores. Os valores são exibidos na figura seguinte.

		COUNTERS	
	EXACT	FIXED	LOG
MAX DEV	1258,68	1278,736	16785,55
ABS MEAN DEV	336,93	355,28	2364,27
STD DEV	415,68	428,2	3321,84
VARIANCE	16425,68	183358,4	11034590,89

Fig .13 – Parâmetros individuais de cada contador

IX. Conclusão

Analisando os resultados obtidos, podemos concluir que todos os caracteres foram identificados em todos os contadores para todos os contadores. É possível concluir também que a grandeza das estimativas do contador com probabilidade fixa e do contador logarítmico é a mesma do contador exato para as sequências de 100 e 1000 caracteres. Para a sequência de 10 000 caracteres, a grandeza do contador com probabilidade fixa é a mesma do contador exato, mas do contador logarítmico não.

Aprofundando a análise dos dados obtidos, podemos concluir também que, em relação ao contador de probabilidade fixa, quando o tamanho da sequência aleatória aumenta, a média do erro, máximo e mínimo descem e, em relação ao contador logarítmico, a média do erro e o erro mínimo sobem.

De modo a comprovar a afirmação do parágrafo anterior, optei por executar o programa para uma sequência de 100 000 000 caracteres. Para esta sequência, seria esperado que os valores do erro médio, máximo e mínimo fossem muito próximos de zero para o contador de probabilidade fixa e elevadíssimos para o contador logarítmico. O valor do erro médio do contador de probabilidade fixa foi 0.13%, o erro mínimo foi 0.02% e o erro máximo 0.36%, ou seja, muito próximos de zero. Para o contador logarítmico o erro médio foi 1599.24%, o erro mínimo foi 530.83% e o erro máximo foi 4338.02%. No directório raiz do projeto encontra-se um *printscreen* (cem_milhoes.png) com os resultados para esta sequência.

Finalmente, podemos concluir também que, quando o tamanho da sequência aumenta, o desvio máximo dos três contadores, o desvio médio absoluto, o desvio padrão e a variância aumentam também, como seria expectável.

VII. Estrutura

No diretório “relatório” encontra-se este relatório em versão PDF. No diretório “enunciado” é possível encontrar o enunciado que serviu de base para o desenvolvimento do programa. No mesmo nível destes dois directórios encontra-se o script a ser executado, *count_chars.py*.

VIII. Execução

Este projeto pode ser executado da seguinte forma:

python3 count_chars.py 100 (para uma sequência aleatória com 100 caracteres).

Se pretendermos testar o programa desenvolvido para diferentes sequências (tamanhos diferentes) podemos executá-lo da seguinte forma:

python3 count_chars.py 100 1000 10000 (para três sequências com tamanhos diferentes : 100, 1000 e 10000).

Links úteis

Para o desenvolvimento do código deste projeto foram utilizados diversos links, os quais posso destacar:

- Python weighted random choices to choose from the list with different probability - https://pynative.com/python-weighted-random-choices-with-probability/
- Python Random choices() Method - https://www.w3schools.com/python/ref_random_choices.asp
- How to get weighted random choice in Python? - https://www.geeksforgeeks.org/how-to-get-weighted-random-choice-in-python/
- Maximum deviation from numbers in a list - https://stackoverflow.com/questions/17915431/maximum-deviation-from-numbers-in-a-list
- Standard deviation of a list - https://stackoverflow.com/questions/15389768/standard-deviation-of-a-list
- Absolute Deviation and Absolute Mean Deviation using NumPy | Python - https://www.geeksforgeeks.org/absolute-deviation-and-absolute-mean-deviation-using-numpy-python/
- Machine Learning - Standard Deviation - https://www.w3schools.com/python/python_ml_standard_deviation.asp