Contagem de palavras com recurso a Count-Min Sketch

Miguel Filipe R. A. M. Fazenda - N. 110877 - miguel.fazenda@ua.pt

Universidade de Aveiro

9 de fevereiro de 2022

Resumo

A Count-Min Sketch é uma estrutura de dados que permite contar as frequências de um certo item pertencente a uma "data stream". Em comparação com contadores exatos, esta fornece um armazenamento sub-linear em troca de uma chance muito baixa de dar valores relativamente acima dos exatos. [1]

O Objetivo desta trabalho é comparar o desempenho entre um contador exato e a Count-Min Sketch através da contagem de palavras de uma obra literária. O desempenho será avaliado em várias versões da mesma obra literária.

1 Introdução

Uma *data stream* é um fluxo informação contínuo e infinito. Pode ser definido de várias maneiras visto ser um termo que se aplica a qualquer situação que possua uma produção de informação constante. [2]

De maneira a lidar com esta escala de produção de informação, foram criados *streaming models*, modelos de processamento de informação que processam a informação logo após a criação da mesma. [3]

Muitas das abordagens fazem uso de várias estruturas de dados diferentes, porém o foco deste trabalho é na Count-Min Sketch. A Count-Min Sketch é uma estrutura de dados que fornece estimativas certas ou acima das contagens exatas.

2 Objetivos

Tendo em conta o que já foi abordado na introdução, os objetivos deste trabalho podem-se resumir a:

- escolher e processar uma obra literária como se fosse uma data stream;
- contar palavras com recurso a um contador exato e uma Count-Min Sketch;
- recolher resultados obtidos pelos contadores, nas várias versões da obra;
- comparar resultados obtidos com os previstos inicialmente, para cada versão;

3 Ferramentas e Bibliotecas

Para a realização deste projeto foram utilizadas as seguintes ferramentas:

- ***PyCharm***, IDE para a linguagem de programação Python; [4]
- ***"Le trois mousquetaires"***, versão inglesa e francesa por Alexandre Dumas; [5] e [6]

4 Count-Min Sketch

Neste capítulo é abordada a estrutura de dados Count-Min SKetch de maneira a dar a entender o funcionamento da estrutura.

A estrutura é composta por um conjunto de arrays. As dimensões da estrutura são definidas pela relação **m** * **d**, onde **m** é o tamanho dos arrays e **d** a quantidade de arrays com função de hash única. Cada elemento do array é um contador.

A estrutura possui 2 métodos que definem o seu funcionamento:

- **update**, método que permite atualizar os valores dos contadores. É feito um hashing do elemento a atualizar na estrutura, em cada um dos arrays;
- **query**, método que permite obter uma estimativa do elemento a pesquisar. A estimativa é feita através da seleção do menor elemento obtido de todas as queries feitas pelas funções de hash nos seus respetivos arrays;

Ambas as caraterísticas possuem o seu impacto nos resultados obtidos. Enquanto que o **m** afeta as colisões nos arrays criadas pelo hashing dos elementos, o **d** afeta a certeza dos resultados devolvidos pela query (quanto menos queries, maior a chance de obter um resultado sobre estimado).

5 Implementação

Neste capítulo é abordado o código implementado. O código apresenta uma estrutura modular pelo que foram criados 2 módulos para as funcionalidades necessárias, sendo elas:

- conversor de ficheiros, prepara os ficheiros para serem percorridos;
- contador de palavras, percorre as palavras, conta-as e cria grafos com a evolução das contagens;

5.1 Conversor de Ficheiros

Este módulo é responsável por converter todas as palavras da obra literária para "*lowercase*"e remover quaisquer caractéres especiais.

Na figura 1 podemos observar o código responsável por converter as:

```python
def big2small(self):
    with open(self.path_to_files + "/3_musk_en.txt", "r", encoding="utf-8") as en:
        with open("livros/3_musk_en_lower.txt", "w", encoding="utf-8") as EN:
            for line in en:
                strip = line.rstrip("\n")
                split = strip.split(" ")
                new_line = ""
                for word in split:
                    new_line += re.sub('[^A-Za-z0-9]+', '', word)
                    new_line += " "
                EN.write(new_line.lower() + "\n")

    with open(self.path_to_files + "/3_musk_fr.txt", "r", encoding="utf-8") as fr:
        with open("livros/3_musk_fr_lower.txt", "w", encoding="utf-8") as FR:
            for line in fr:
                strip = line.rstrip("\n")
                split = strip.split(" ")
                new_line = ""
                for word in split:
                    new_line += re.sub('[^A-Za-z0-9]+', '', word)
                    new_line += " "
                FR.write(new_line.lower() + "\n")
```

Figura 1: código do conversor

5.2 Contador de palavras

Este módulo é responsável por contar todas as palavras. As contagens são feitas por um dicionário (Hash-table da linguagem Python) e uma Count-Min Sketch.

Segue-se um *snippet* do código responsável por contar as palavras:

```python
def count_en(exact, cms):
    with open("./livros/3_musk_en_lower.txt", "r", encoding="utf-8") as file:
        for line in file:
            strip = line.rstrip("\n")
            words = strip.split(" ")
            for word in words:
                if word != "":
                    if word in exact:
                        exact[word] += 1
                    else:
                        exact[word] = 1

                    cms.update(word, 1)
```

Figura 2: código do contador

O contador também permite criar grafos com a variação das contagens segundo **m** ou **d**. Segue-se os *snippets* que permitem criar os grafos coma informação:

Segue-se também o *snippet* de código responsável pela lógica entre a contagem das palavras e a criação dos grafos:

6 Resultados

Como já foi abordado no capítulo 4, conseguimos verificar através dos grafos os impactos esperados da

```python
def create_m_variation_graphs(variable_attribute_results, cms, language):
    # count variation
    figure = plt.figure(num=1, figsize=(16, 9))
    plt.xlabel("m")
    plt.ylabel("Contagens")
    plt.title(f"Variação das contagens consoante m, para d={cms.d}")

    m_variations = []
    contagens = []
    for x in variable_attribute_results:
        m_variations.append(x[0])  # m(valores do eixo x)
        contagens.append(x[5])  # cms_counts(valores do eixo y)

    plt.bar(m_variations, contagens, color="b", width=0.4, edgecolor="grey", label="Quantidade de contagens do Count-Min Sketch")
    plt.axhline(y=variable_attribute_results[0][4], color="r", linestyle="-", label="Quantidade de contagens exatas")

    # plt.show()
    plt.savefig("./results/" + language + "_m_count_variation.png")
    plt.close(figure)

    # median variation
    figure = plt.figure(num=2, figsize=(16, 9))
    plt.xlabel("m")
    plt.ylabel("Médias")
    plt.title(f"Variação das médias das sobre estimativas consoante m, para d={cms.d}")

    medias = []
    for x in variable_attribute_results:
        medias.append(x[7])

    plt.bar(m_variations, medias, color="g", width=0.4, edgecolor="grey", label="Médias das sobre estimativas")

    plt.savefig("./results/" + language + "_m_mean_variation.png")
    plt.close(figure)
```

Figura 3: criação dos grafos em função de m

```python
def create_d_variation_graphs(variable_attribute_results, cms, language):
    # count variation
    figure = plt.figure(num=1, figsize=(16, 9))
    plt.xlabel("d")
    plt.ylabel("Contagens")
    plt.title(f"Variação das contagens consoante d, para m={cms.m}")

    d_variations = []
    contagens = []
    for x in variable_attribute_results:
        d_variations.append(x[1])  # d(valores do eixo x)
        contagens.append(x[5])  # cms_counts(valores do eixo y)

    plt.bar(d_variations, contagens, color="b", width=0.1, edgecolor="grey",
            label="Quantidade de contagens do Count-Min Sketch")
    plt.axhline(y=variable_attribute_results[0][4], color="r", linestyle="-", label="Quantidade de contagens exatas")

    # plt.legend("d:", cms.d)

    # plt.show()
    plt.savefig("./results/" + language + "_d_count_variation.png")
    plt.close(figure)

    # median variation
    figure = plt.figure(num=2, figsize=(16, 9))
    plt.xlabel("d")
    plt.ylabel("Médias")
    plt.title(f"Variação das médias das sobre estimativas consoante m, para d={cms.d}")

    medias = []
    for x in variable_attribute_results:
        medias.append(x[7])

    plt.bar(d_variations, medias, color="g", width=0.1, edgecolor="grey", label="Médias das sobre estimativas")

    plt.savefig("./results/" + language + "_d_mean_variation.png")
    plt.close(figure)
```

Figura 4: criação dos grafos em função de d

```python
def count_words(min_m=None, max_m=None, min_d=None, max_d=None):
    print("A contar palavras...")

    # cria contadores
    en_exact_counter = {}
    fr_exact_counter = {}

    # resultados da variação do m, com d=min_d
    var_m_en_res = []  # versão inglesa
    var_m_fr_res = []  # versão francesa
    for m in range(min_m, max_m+1, 1000):
        en_cms = CountMinSketch(m=m, d=min_d)
        fr_cms = CountMinSketch(m=m, d=min_d)
        count_en(en_exact_counter, en_cms)
        count_fr(fr_exact_counter, fr_cms)
        var_m_en_res.append(calculate_result(en_exact_counter, en_cms, "EN"))
        var_m_fr_res.append(calculate_result(fr_exact_counter, fr_cms, "FR"))
        en_exact_counter.clear()
        fr_exact_counter.clear()

    create_m_variation_graphs(var_m_en_res, en_cms, "EN")
    create_m_variation_graphs(var_m_fr_res, fr_cms, "FR")

    # resultados da variação do d, com m=min_m
    var_d_en_res = []
    var_d_fr_res = []
    for d in range(min_d, max_d+1, 1):
        en_cms = CountMinSketch(m=min_m, d=d)
        fr_cms = CountMinSketch(m=min_m, d=d)
        count_en(en_exact_counter, en_cms)
        count_fr(fr_exact_counter, fr_cms)
        var_d_en_res.append(calculate_result(en_exact_counter, en_cms, "EN"))
        var_d_fr_res.append(calculate_result(fr_exact_counter, fr_cms, "FR"))
        en_exact_counter.clear()
        fr_exact_counter.clear()

    create_d_variation_graphs(var_d_en_res, en_cms, "EN")
    create_d_variation_graphs(var_d_fr_res, fr_cms, "FR")
```

Figura 5: lógica entre contagem e criação

variação tanto de **m** como de **d** nas contagens efetuadas. Os resultados de ambas as versões demonstram o mesmo comportamento, um impacto notável no início da variação, que vai diminuindo com o aumento da variação.

M demonstra um maior impacto a combater tanto a quantidade de sobre estimativas assim como a média das sobre estimativas, porém necessita de uma variação muito maior do que **d**.

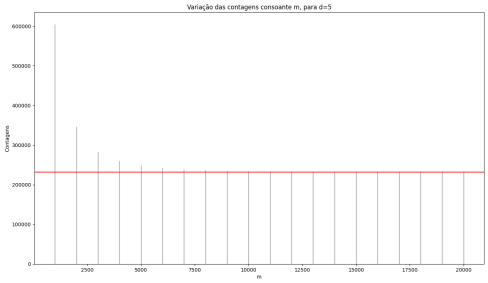

Figura 6: contagem inglesa: variação do m, para d=5

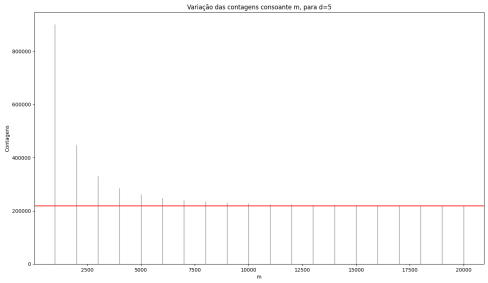

Figura 7: contagem francesa: variação do m, para d=5

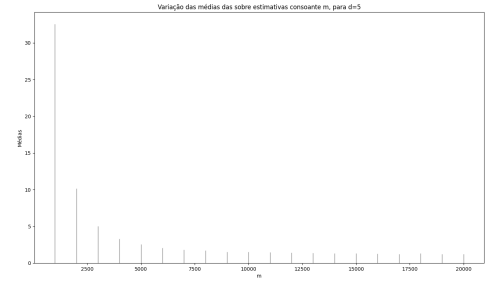

Figura 8: média inglesa: variação do m, para d=5

No entanto, como podemos ver nos seguintes grafos, **d** precisa de uma variação muito menor para demonstrar um impacto mais notável, mas que também tende a diminuir.

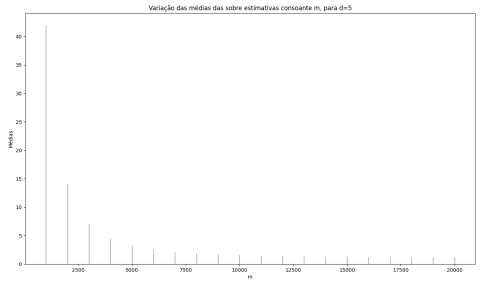

Figura 9: média francesa: variação do m, para d=5

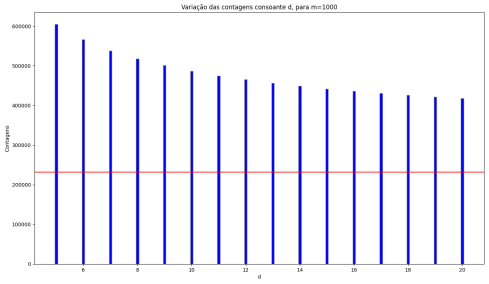

Figura 10: contagem inglesa: variação do d, para m=1000

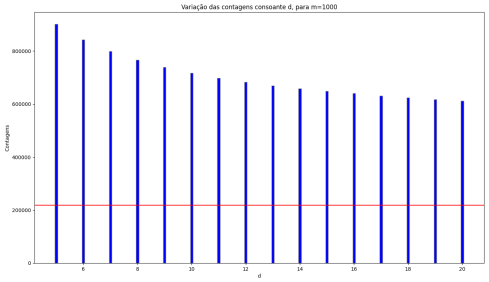

Figura 11: contagem francesa: variação do d, para m=1000

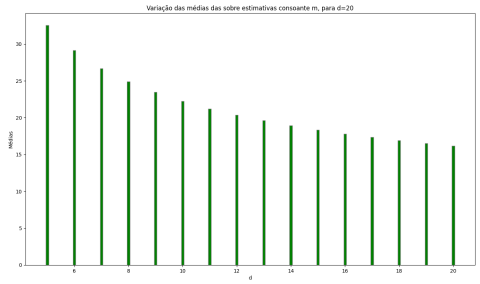

Figura 12: média inglesa: variação do d, para m=1000

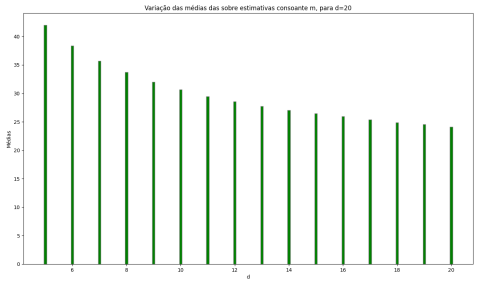

Figura 13: média francesa: variação do d, para m=1000

7 Conclusão

Após a realização do trabalho, foi possível obter um conhecimento aprofundado da estrutura de dados Count-Min Sketch, assim como também o impacto que as suas caraterísticas têm nas estimativas.

De maneira a comprovar melhor o comportamento da estrutura, os testes foram feitos em várias versões da mesma obra literária.

Os testes comprovam que a variação das caraterísticas **m** e **d** afetam respetivamente a quantidade de colisões e a chance de obtenção de más estimativas ao realizar queries. Embora ambas afetam a obtenção dos resultados, o aumento de **m** demonstra um impacto maior em relação a **d**.

Referências

[1] Graham Cormode e S. Muthukrishnan. "An Improved Data Stream Summary: The Count-Min Sketch and Its Applications". Em: *LATIN 2004: Theoretical Informatics*. Ed. por Martín Farach-Colton. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 29–38.

[2] Alessandro Margara e Tilmann Rabl. "Definition of Data Streams". Em: *Encyclopedia of Big Data Technologies*. Ed. por Sherif Sakr e Albert Y. Zomaya. Cham: Springer International Publishing, 2019, pp. 648–652. ISBN: 978-3-319-77525-8. DOI: 10.1007/978-3-319-77525-8_188. URL: https://doi.org/10.1007/978-3-319-77525-8_188.

[3] Lukasz Golab. "Stream Models". Em: *Encyclopedia of Database Systems*. Ed. por LING LIU e M. TAMER ÖZSU. Boston, MA: Springer US, 2009, pp. 2834–2836. ISBN: 978-0-387-39940-9. DOI: 10.1007/978-0-387-39940-9_370. URL: https://doi.org/10.1007/978-0-387-39940-9_370.

[4] JetBrains. URL: https://www.jetbrains.com/pycharm/.

[5] Alexandre Dumas. *The Three Musketeers*. URL: https://www.gutenberg.org/ebooks/1257.

[6] Alexandre Dumas. *Le Troi Mousquetaires*. URL: https://www.gutenberg.org/ebooks/13951.