**Informações sobre os programas (do projeto de MPEI)**

João Eduardo Alcatrão, 76763, CM

**Módulos**

Os módulos presentes neste trabalho são:

* CountingFilter.m (Counting Bloom Filter, é idêntico ao Insert\_Bloom\_filter mas incrementa o valor do index correspondente à hash do item processado. Funciona melhor em classes ‘double’ devido ao range de valores que esta classe admite (2^32 vs 256 para a classe ‘uint8’, que é a classe ‘geral’ do Bloom Filter).
* Stochastic.m (Contador estocástico, calcula o nº de ocurrências de um elemento numa estrutura de dados até certo ponto. Isto faz com que seja possível ter uma estimativa mais ou menos precisa sobre esse nº de ocurrências na estrutura de dados sem ter que se verificar todos os elementos dessa mesma estrutura. Basicamente, troca-se precisão/exatidão por tempo. Funciona melhor quando existe aleatoriedade uniforme nos elementos das estruturas analisadas).
* T3MinHash.m (Função que calcula a similaridade entre elementos recorrendo aos conceitos de distância de Jaccard e MinHashing. É umas das 2 funções de MinHashing presentes neste projeto).
* MinHashingstring2hash.m (Outra função de cálculo de similaridade, que também recorre aos conceitos de distância de Jaccard e MinHashing. Esta função é implementada de forma diferente da função acima, e usa a função string2hash.m para projetar hashes).

Além destas, existem outras funções no trabalho que apoiam os módulos principais e que são usadas nos testes. Estas são:

* Bloom\_Filter\_insert.m (Função que ‘insere’ a hash de um item no Bloom Filter. Mais concretamente, é calculado um valor através da divisão modular de uma hash pelo tamanho do Bloom Filter, e esse valor é usado como index no Bloom Filter, e nessa posição será marcado um ‘1’ para assinalar a pertença da hash no Bloom Filter).
* BloomCount.m (Anexo do counting Bloom Filter, estima o nº de ocurrências de algum item em função das respetivas hashes do item inseridas no Bloom Filter).
* Bloom\_Filter.m (Cria um vetor de classe ‘uint8’ (ocupa menos espaço) que servirá para armazenar informações sobre a pertença ou não de uma hash, e por consequência um item ao próprio Bloom Filter. Isto é útil para saber se um item (talvez) pertence ou definitivamente não pertence a um conjunto de dados de maneira muito rápida, onde só é necessária uma operação (não é necessário verificar os elementos do conjunto um a um).
* String2Hash.m (Função de hashing providenciada nas aulas de MPEI. Dado um item (que pode ser um número, uma string, etc.; pode ser de diversos tipos), esta função calcula um valor numérico inteiro entre 0 e 2^32-1 de forma aleatória, mas que é sempre o mesmo para o mesmo item. Itens diferentes podem ter a mesma hash, mas a inserção de um item na hash function retorna sempre a mesma hash).
* Shingles.m (Cria uma célula de strings a partir de um documento, em que a ordem das strings é conservada. O nº de elementos (strings) nessa célula é length(doc)-shingle\_size+1, sendo doc o documento a ser analisado e shingle\_size o nº de strings concatenadas a partir do documento por elemento).
* **Testes**

Todos os testes foram feitos para serem executados a partir do Matlab, usando o comando “Run Section” para visualizar os resultados de maneiras mais eficiente. Os testes aqui presentes são:

T1\_Contador.m (Teste que utiliza o contador estocástico para estimar quantas vezes uma sequência de cartas existe num grande conjunto de experiências. Estas experiências têm resultados uniformemente distribuídos, pelo que se pode considerar com um teste apropriado a este módulo)

T2\_CountingBloomFilter.m (Usa o Counting Bloom Filter para verificar se um número escolhido por nós é primo ou não. Usa um conjunto de dados exterior (Primos.txt) para facilitar a implementação. Tem uma segunda parte, onde é gerado um conjunto aleatório de primos e é estimada o nº de ocorrência de cada um usando o Counting Bloom Filter)

T3\_SimilarItems.m (Usa a função T3\_MinHash.m para cálculo da similaridade entre vários produtos de beleza (ficheiros de texto que contêm os ingredientes do respetivo produto)).

T3in1.m (Uso do contador estocástico e do counting bloom filter para determinar se uma determinada frase é dita (e quantas vezes o é) em cada um dos três primeiros filmes do Harry Potter.

De seguida, calcula-se a similaridade entre os filmes em que a frase é dita através de MinHashing).