universidade de aveiro

Bloom Filter e Minhash

Desenvolvimento, teste e demonstração algorítmica para determinar pertença a um conjunto e determinação de itens similares

Métodos Probabilísticos para Engenharia Informática

2016/2017

P3

João Santos	*73761*
Manuel Gil	*72327*

Professor Amaro Sousa

Desenvolvimento Algorítmico

Para a realização do trabalho prático foi proposto pelos docentes a utilização da linguagem de programação JAVA, como consequência optamos pelo NetBeans para aplicar os algoritmos exigidos.

Módulos Implementados

Foram implementados os dois módulos inicialmente exigidos pelos docentes:

- **Bloom Filter**

Utilizado para a introduzir elementos num conjunto e testar se um outro elemento é membro desse mesmo conjunto utilizando todas as propriedades inerentes ao Bloom Filter.

Classe - BloomF.java .

Métodos - boolean[] initb, void insertString e boolean isMember.

- **MinHash**

Utilizado para estimar a similaridade entre dois conjuntos com a utilização da implementação de hash functions.

Classe - MinH.java .

Métodos - void getSetOfWords, void getFileInfo, int hashFunction, void hashSet e void hashSets .

Testes efetuados aos Módulos e respectiva demonstração

- **Bloom Filter**

Relativamente aos testes efetuados no Bloom Filter foram utilizados dois discursos bastante polémicos da anterior Primeira Dama dos Estados Unidos da América, Michelle Obama e da atual primeira dama, Melania Trump.

Discursos estes que se tornaram bastante polémicos devido à sua similaridade.

Classe - Teste.java

✓ Os dois documentos .txt que contém os discursos são introduzidos como argumentos.

- ✓ É pedido um valor para o tamanho do Bloom Filter e o número de funções de dispersão a utilizar.
- ✓ Todas as alocações do Bloom Filter são colocadas a falso (boolean[] initb).
- ✓ O ficheiro .txt é lido e armazenamos todas as palavras num ArrayList.
- ✓ Os caracteres especiais (.,!?-;) quando lidos são eliminados para facilitar a comparação no Bloom Filter.
- ✓ Depois de formatado o texto podemos então introduzir todas as palavras no Bloom Filter.
- ✓ Repetimos o processo para o segundo ficheiro .txt.
- ✓ Estamos prontos para fazer a comparação (boolean isMember) dos dois ficheiros e consequentemente fazer a contagem de positivos.
- ✓ Por fim apresentamos a probabilidade de falsos positivos, melhor número de hash functions, número de bits a 0 e a 1.

- **MinHash**

Para efetuar os testes ao MinHash utilizamos diferentes discursos de ex-presidentes dos Estados Unidos da América.

Classe - TesteMinH.java

- ✓ São definidas inicialmente 10 Hash Functions.
- ✓ Inicializamos a matriz e criamos um ArrayList de HashMaps.
- ✓ Efetuamos o shingle para dividir as palavras em blocos de 3.
- ✓ Agora é altura de ir buscar as palavras ao ficheiro para alocar no HashMap.
- ✓ Percorremos todas as posições do HashMap e fazemos o Hash(10x) dos blocos (método HashSet).
- ✓ Os valores colocados na matriz correspondem ao valor mais baixo recolhido por cada HashFunction.
- ✓ A matriz é impressa para consulta.
- ✓ Por fim fazemos a comparação de valores iguais na matriz de modo a calcular a similaridade e distância Jaccard.

Como correr os Programas

- **Bloom Filter (Terminal)**

  Na pasta 'src/mpeitp' do projecto:

  - *javac Teste.java BloomF.java*
  - *java -cp /the-classpath-root/( )/MpeiTp/src/mpeitp.Teste Michelle.txt Melania.txt*

- **MinHash (Terminal)**

  Na pasta 'src/mpeitp' do projecto:

  - *javac TesteMinH.java MinH.java*
  - *java -cp /the-classpath-root/( )/MpeiTp/src/mpeitp.TesteMinH*

Conclusão e referências

A realização deste Projeto contribuiu positivamente para a aprendizagem individual e como grupo destacando como aspecto principal a familiaridade com a linguagem JAVA.

Como aspecto negativo, a realização dos trabalhos de aula em MATLAB levou a uma reaprendizagem de alguns componentes a integrar no projeto em JAVA.

Os Guiões Praticos e Teoricos acabaram por ser a maior fonte para a elaboração do Projeto.

No geral achamos que obtivemos um bom aproveitamento das diferentes componentes que envolviam o Projeto.

Referências:

- *http://pages.cs.wisc.edu/~cao/papers/summary-cache/node8.html - bloom*
- *http://blog.bimarian.com/bloom-filters-implementation-in-java - bloom*
- *http://robertheaton.com/2014/05/02/jaccard-similarity-and-minhash-for-winners/ - minhash*
- *http://matthewcasperson.blogspot.pt/2013/11/minhash-for-dummies.html - shingle*
- *https://www.cs.cmu.edu/~adamchik/15-121/lectures/Hashing/hashing.html - hashing*
- *https://www.jasondavies.com/bloomfilter/ - bloom*
- *Guiões Práticos e Teóricos MPEI*