

IR-Grundlagen - Worksheet 1

Dieses Worksheet behandelt den Inverted-Index sowie boolsches Retrieval und verschiedene merge-Algorithmen (AND, ANDNOT, OR, XOR). Ziel des Worksheets ist es am Ende einfache 2-Wort Queries durchführen zu können.


Zuerst beschäftigen wir uns mit der Indexierung. Dafür sind folgende Schritte notwendig:

1. Zuerst müssen die vorhandenen Dokumente indexiert werden. Sie müssen sich nicht darum kümmern die Dateien in Java einzulesen. Diese Arbeit wurde bereits übernommen. Die Main-Controller-Klasse besitzt eine ArrayList (`collection`) mit Document-Objekten. Jedes Dokument der `collection` wird durch ein solches Dokument-Objekt repräsentiert. Die InvertedIndex-Klasse bekommt `collection` vom Maincontroller übergeben.

Sie besitzt außerdem folgende Variable:

```
HashMap<String, ArrayList<Integer>> invertedIndex
```

Diese Variable repräsentiert den Inverted-Index wobei der Key der Hashmap einen Term darstellt und der Value eine Liste der Dokumente in denen dieser Term auftritt. Sie können diese Variable nutzen um den Inverted-Index zu speichern. Sie können sich auch eine eigene Datenstruktur überlegen.

- a. Nutzen sie die zur Verfügung stehenden `public` Methoden der Document-Klasse um den Index zu befüllen. Erledigen Sie diese Arbeit im Konstruktor der Inverted-Index-Klasse. Legen sie bei Bedarf weitere Methoden und Variablen an.
-  b. Nun da jedes Dokument indexiert wurde benötigen Sie eine Methode herauszufinden in welchen Dokumenten ein Term auftritt. Implementieren sie hierzu die Methode

```
ArrayList<Integer> searchForSingleWord(String word)
```

Diese Methode soll eine Liste von denjenigen Dokumenten-ID's (Integer-Werte) zurückgeben in denen sich der Term (`String word`) befindet.


Um alle Schritte zu testen, die wir bisher durchgeführt haben, erstellen Sie im MainController eine Methode die ein Wort vom User einliest und die Liste der Dokumente in denen dieses Wort vorkommt ausgibt. Denken Sie hier an Case-folding.

Wir haben jetzt alle Voraussetzungen für ein simples Retrievalsystem. Die nächsten beiden Aufgaben beschäftigen sich mit der Integration von booleschen Operatoren in die Suche

2. Als erstes müssen sie sich um das Einlesen der Query kümmern. Im MainController befinden sich die Methode

```
String[] queryTerms = getQueryTerms()
```

Lesen Sie in dieser Methode die Query vom User ein (Queries sollten die Form: "firstTerm AND secondTerm" haben), führen Sie alle notwendigen Formatierungen durch (z.B.: Case-Folding) und geben sie die beiden Query-Terms in einem Array zurück.

-  3. Das Array mit den Suchbegriffen wird an die Methode




```
performANDMerge(String[] queryTerms)
```

in der InvertedIndex-Klasse übergeben. Implementieren Sie diese so, dass Sie eine `public ArrayList<Integer>` mit den Dokumenten-IDs der Dokumente die beide Terme beinhaltet, zurückgibt.

Hinweis: Pseudocode für diese Methode finden sie in den Folien der Vorlesung.

Optionale Aufgaben:

4. Erweitern Sie die InvertedIndex-Klasse um Methoden für folgende Operatoren

-  a. ANDNOT
-  b. OR
-  c. XOR

5. Wie müsste ein Vorgehen aussehen das Suchanfragen mit mehreren (verschiedenen) Operatoren ermöglicht?

Alle mit einem  gekennzeichneten Aufgaben besitzen einenJUNIT-Test.

Um diese Aufgaben zu testen klicken sie in Eclipse mit der rechten Maustaste auf ihr Projekt, wählen sie "Run as" → "JUnit Test". Die Ergebnisse werden in einem Tab in Eclipse angezeigt. Grüne Ergebnisse sind korrekt, blaue Ergebnisse haben ein anderes Ergebnis als erwartet (sie können die erwarteten und tatsächlichen Ergebnisse ansehen) und bei roten Ergebnissen ist ein Error aufgetreten.