Beleg 1

Ziel des ersten Belegs ist es, die Anwendung von Higher Order Functions in Zusammenhang mit unterschiedlichen Datenstrukturen zu üben. Ausgangspunkt der Übung sind Textdateien, für die verschiedenste Statistiken erstellt werden sollen. Weiter sollen die Dateien effizient durchsuchbar gemacht werden. Das Grundgerüst für alle zu schreibenden Funktionen sowie verschiedene Tests finden Sie in dem Beleg-Projekt, dass unter plus.htw-berlin.de heruntergeladen werden kann.

In dem Projekt befinden sich in Ordnern Ressourcen-Ordnern verschiedene Beispieldateien, die für die Analyse verwendet werden können/sollen: MobyDick.txt (Buch Moby Dick von Herman Melville), MobyDickC1.txt (Kapitel 1 von Moby Dick), MobyDickShort.txt (Teil aus dem ersten Kapitel), GoneWithTheWind.txt (Buch "Vom Winde verweht" von Margaret Mitchell) sowie AFINN-111.txt (Beispieldatei für die Sentimentanalyse).

Aufgabe 1: Vervollständigen Sie die Funktionen getWords, getAllWords und countTheWords, die das Zählen von Wörtern innerhalb eines Texts ermöglichen sollen. Der Text befindet sich dabei im folgenden Format: Jede Zeile besteht aus einem Tupel (Int, String), das eine Zeilennummer enthält sowie die Textzeile als String. Beim Zählen der Wörter soll keine Unterscheidung zwischen Groß- und Kleinschreibung gemacht werden, d.h. Baum wäre bspw. dasselbe Wort wie bauM. Gehen Sie dabei folgendermaßen vor:
Entwickeln Sie im ersten Schritt die Funktion getWords, die aus einem String alle Wörter extrahiert, Löschen Sie zu diesem Zweck alle Elemente die keine Buchstaben sind und splitten Sie dann den String auf Basis der Lehrzeichen (Hinweis: Schauen Sie sich die Funktionen split und replace dafür an.)
Danach implementieren Sie die Funktion getAllWords, die aus dem Gesamttext (Liste von Tupeln (Zeilennr, Text) alle Wörter extrahiert, Dabei sollen noch keine Zählungen vorgenommen werden.
Sind die ersten beiden Funktionen implementiert, so kann eine Liste aller im Text vorkommenden Wörter generiert werden, was wiederum Input der Funktion countAllWords ist. Sie zählt alle Vorkommen eines Wortes und gibt eine Liste bestehend aus dem Wort und die Anzahl der Vorkommen zurück. Verwenden Sie dafür die Datenstruktur Map.

Aufgabe 2: Implementieren Sie das Wörterzählen auf Basis der in Aufgabe 2 vorgegebenen MapReduce-Funktion. Definieren Sie dazu eine Map- sowie eine Reduce-Funktion und verwenden Sie diese für den Aufruf. Die Funktion countTheWordsMR soll demnach nur noch aus einem Aufruf von MapReduce mit den entsprechenden Typparametern bestehen.

Aufgabe 3: In dieser Aufgabe sollen große Texte durchsuchbar gemacht werden, d.h. es soll nach Zeilen gesucht werden können, in denen beliebig gewählte Schlüsselwörter vorkommen. Dazu soll im ersten Schritt ein sogenannter Inverser Index erstellt werden. Im Inversen Index soll gespeichert werden, in welchen Zeilen innerhalb eines Dokuments ein Wort vorkommt. Implementieren Sie hier zunächst die Funktion getAllWordsWithIndex. Diese Funktion bekommt den Text im Format List((Zeilennr, Zeilentext)), extrahiert alle Wörter und speichert diese als Tupel in der Form (Zeilennr, Wort). Jetzt müssen in der Funktion createInverseIndex sämtliche Vorkommen eines Wortes (Zeilennummern) innerhalb einer Map zusammengefasst werden. Ergebnis der Funktion ist somit eine Map, die ein Wort auf eine Liste von Zeilennummern (Int) abbildet.

Auf dieser Basis können jetzt die Funktionen andConjunction und orConjuction implementiert werden. Ziel der Funktionen ist es, für eine Liste von Wörtern die Zeilen herauszusuchen, die diese enthalten. Dabei sollen bei orConjuction die Wörter “verodert” werden und bei andConjuction mit dem logischen and verknüpft werden.

Aufgabe 4: Implementieren Sie das Wörterzählen mit der im Objekt mapreduce.BasicOperations vorgegebenen mapreduce-Funktion. Sie implementiert die “Google-Variante” der mapreduce-Funktion.

Aufgabe 5: In dieser Aufgabe soll eine Sentiment-Analyse durchgeführt werden. In einer Sentiment-Analyse wird gefühlsausdrückenden Wörtern ein Wert von +5 (positiv) bis -5 (negativ) zugeordnet. Diese Zuordnung befindet sich in der Datei AFINN-111.txt. In der vorgegebenen Funktion getSentiments wird diese Datei ausgelesen und im Format einer Map [String, Int] bereitgestellt. Auf Basis dieser Sentiment-Zuordnung soll eine Analyse durchgeführt werden. Entwickle hierzu als erstes die Funktion getDocumentGroupedByCounts. Die Funktion bekommt als Parameter einen Filenamen sowie die Anzahl von Wörtern, die innerhalb eines Abschnitts zusammengefasst werden sollen. Ergebnis der Funktion ist dann ein Tupel bestehend aus der Abschnittsnummern und einer Liste von Wörtern, die dort vorkommen.
Letzte zu implementierende Funktion ist analyzeSentiments. Diese bekommt eine Liste von Abschnitten (Abschnittnr, Liste von Wörtern) und errechnet den jeweiligen Sentimentwert. Ergebnis ist ein Tripel bestehend aus der Abschnittsnummer, dem Sentimentwert und der relativen Anzahl von Wörtern, die für die Sentimentanalyse verwendet werden konnten.

In der Klasse App befindet sich eine Mainklasse, die für das Buch „Vom Winde verweht“ die Ergebnisse der Sentimentanalyse graphisch darstellt. Testen Sie diese und überprüfen Sie die Plausibilität der Ergebnisse.

Die Belegarbeit kann in Gruppen von 1-2 Personen bearbeitet werden und ist bis zum 25.11.2014 abzugeben. Die Funktionen sollen mindestens die mitgelieferten Tests bestehen. Es ist ratsam, weitere Tests hinzuzufügen, da bei der Kontrolle der Aufgaben evtl. weitere Tests durchgeführt werden.
Die Abnahme der Belegaufgabe erfolgt in der Übung oder meiner Sprechstunde.