Übung Big-Data-Technologien
Prof. Dr. Christoph Schmitz

Übungsblatt 2

MapReduce

Hinweise:

- In Stud.IP steht eine Projektvorlage `bdt-uebung-2-mapredexample.zip` zur Verfügung. Diese enthält eine Mapper-, eine Reducer- und eine Treiberklasse.
- Bauen unter Eclipse funktioniert mit „Run as → Maven build“
- Bei Problemen mit den Windows-Rechnern kopieren Sie Ihr Projekt (den Quellcode) am besten auf einen der infbdtXY-Rechner und bauen ihn dort mit `mvn clean package`.
- Das unten genannte Beispiel `caesar.txt` finden Sie im HDFS unter `/data/caesar.txt`.

Zum Warmwerden (ohne Wertung)

Schauen Sie sich die Struktur des Job-Jar-Files an. Machen Sie sich klar, wo Ihr Code und wo Bibliotheken eingepackt sind. Werfen Sie einen Blick auf die Manifest-Datei. Wozu brauchen wir diese?

1 MapReduce: Filtern

Schreiben Sie einen MapReduce-Job, der Eingabezeilen auswählt, die ein bestimmtes Wort enthalten. Das Wort soll konfigurierbar sein.

Brauchen Sie hierzu einen Reducer? Finden Sie heraus, wie man die Anzahl der Reducer konfiguriert (im Code oder extern per Kommandozeile).

Aufrufbeispiel: `yarn jar filter-job.jar caesar caesar-filtered bellum`

Hierbei sind `caesar` und `caesar-filtered` Ein- und Ausgabeverzeichnis, und „bellum“ ist das gesuchte Wort.

2 MapReduce: Aggregation

Schreiben Sie einen MapReduce-Job, der Eingabezeilen zählt, die länger als 100 Zeichen sind. Ihr Job soll als Ergebnis eine Statistik über die Häufigkeiten der Zeilenlängen der verbliebenen Zeilen erstellen, also z. B.

101	1
102	2
104	8
105	12
…	…

Dies steht für: 1 Zeile mit 101 Zeichen, 2 Zeilen mit 102 Zeichen usw. Probieren Sie Ihr Programm wieder an `caesar.txt` aus.

Hinweis: Sie können zur Kontrolle für mäßig große Daten das gewünschte Ergebnis lokal berechnen mittels

```
awk 'length($0) > 100 {print length($0)}' < caesar.txt | sort -n | uniq -c
```