# 6. Übungsblatt

Ausgabe: 24. Juni 2020 · Besprechung: ab 08. Juli 2020

## Aufgabe 1: Gruppierte Aggregation mit MapReduce

MapReduce ist ein Programmiermodell für Clusteranwendungen. Der Programmierer spezifiziert eine Anwendung durch eine Map-Funktion und eine Reduce-Funktion:

**Map** $f_1 : \alpha \to [\langle \beta, \gamma \rangle]$
Map extrahiert aus einer Eingabe $\alpha$ eine Liste von Schlüssel-Wert Paaren $\langle \beta, \gamma \rangle$.

**Reduce** $f_2 : \langle \beta, [\gamma] \rangle \to \delta$
Reduce verarbeitet für einen Schlüssel $\beta$ die Liste aller zugehörigen Werte $\gamma$.

Systeme wie z.B. Hadoop führen den Map und Reduce Schritt dann auf einem verteilt gespeicherten Datensatz aus. Zwischen Map und Reduce sorgt *Shuffle* dafür, dass alle Elemente mit dem gleichem Schlüssel an den selben Reduce Aufruf geleitet werden.

Sie sollen nun die folgende Anfrage mittels MapReduce berechnen lassen:

```
select   f.DateKey, sum(f.SalesAmount)
from     FactSales f
group by f.DateKey
```

Die Berechnung sollen Sie in folgenden Teilaufgaben beschreiben:

1. Definieren Sie die Funktionen $f_1$ (Map) und $f_2$ (Reduce) durch Pseudocode.
2. Zeigen Sie anhand einfacher Beispieldaten, wie MapReduce auf $n \geq 4$ Maschinen mit Ihren Funktionen die gegebene Anfrage auswertet.

## Aufgabe 2: MapReduce Verarbeitungsmodelle

Für Clusteranwendungen haben sich unterschiedliche Verarbeitungsmodelle etabliert. Zum Beispiel wird das klassische MapReduce Modell [1] durch Apache Hadoop umgesetzt. Apache Spark [2] erweitert MapReduce und verspricht einige Verbesserungen.

Erklären auf Basis der Quellen welche Vorteile Spark gegenüber Hadoop bietet. Gehen Sie dabei auf die folgenden Aspekte ein:

### a) Datenfluss

Welche Erweiterungen des Datenflusses bietet Spark gegenüber dem Ablauf Map → Reduce?

### b) Datenhaltung

Wie ermöglicht Spark schnellere Datenzugriffe bei mehrstufigen Datenflüssen?

## Aufgabe 2: Bloom Filter

Bei der Berechnung von *Joins* ist das *Testen auf Enthaltensein* eine wichtige Operation. Kann ausgeschlossen werden, dass der Joinschlüssel eines Tupels in der Join-Tabelle enthalten ist, kann das Tupel verworfen werden.

Der *Bloom Filter* ist eine kompakte probabilistische Datenstruktur, die das Testen auf Enthaltensein ermöglicht. Der Filter besteht aus einem $m$-dimensionalen Bitvektor and $k$ Hashfunktionen, die Werte auf eine Zahl von 0 bis $m - 1$ abbilden. Jeder Wert der Menge $\mathcal{M}$ wird mit jeder der $k$ Funktionen gehasht und für jeden Hash-Wert wird ein Bit an der entsprechenden Position des Bitvektors gesetzt.

Für die Testmenge $\mathcal{T}$ werden Anfragen an den Filter gestellt. Jeder Wert $t \in \mathcal{T}$ wird mit jeder der $k$ Funktionen gehasht. Steht an mindestens einer der Positionen im Bitvektor eine 0, kann ausgeschlossen werden dass $t$ in $\mathcal{M}$ enthalten ist.

#### a) Beispiel

Veranschaulichen Sie zunächst die Funktionsweise des Bloom Filters mit zwei einfachen Hash-Funktionen und zwei kleinen **Beispielmengen** $\mathcal{M}$ und $\mathcal{T}$ von ganzen Zahlen. Als Hash-Funktionen eignen sich Beispielweise eine Kombination aus Modulo und Quersumme. Achten Sie darauf, dass die Hash-Werte im Bereich 0 und $m - 1$ liegen müssen.

#### b) Anfrage

Skizzieren Sie die Verwendung eines Bloom Filters zur Auswertung der folgenden Starjoin-Anfrage:

```
select sum(Sales.Revenue)
  from Sales, Territory, Date
 where Sales.TerritoryKey = Territory.Key
   and Sales.DateKey = Date.Key
   and Territory.Country = 'United States'
   and Date.CalendarYear between 2005 and 2008
```

## Literatur


[1] Jeffrey Dean et al.: MapReduce: simplified data processing on large clusters. Communications of the ACM 2008.

[2] Matei Zaharia et al.: Spark: Cluster computing with working sets. HotCloud 2010.