innFactory

Anwendung der funktionalen Programmierung mit Scala

TH Rosenheim - SoSe 2025

innFactory

Data Engineering / Spark

Spark
Was ist Spark?

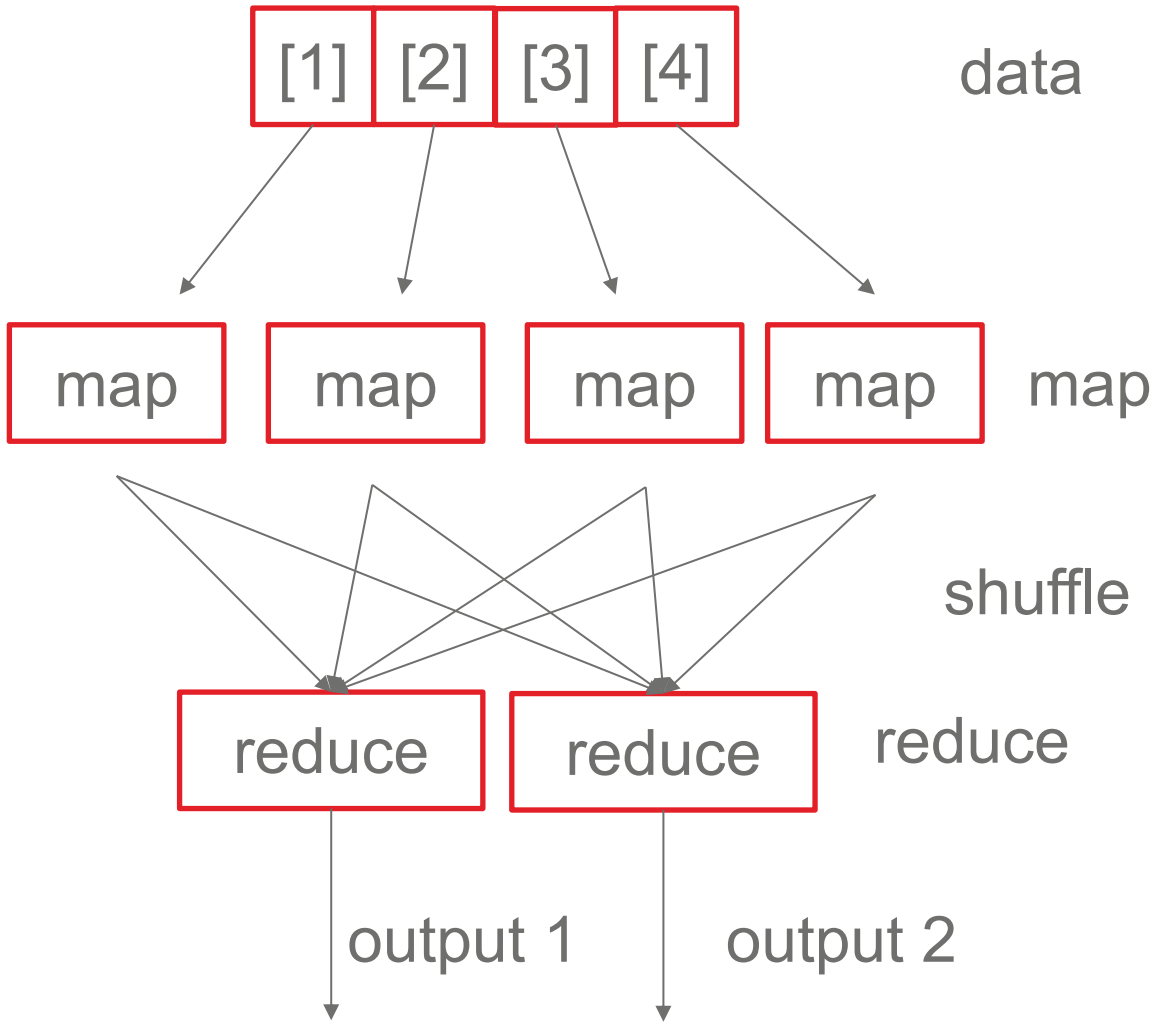
- Spark ist eine Unified Computing Engine für die verteilte Datenverarbeitung v.a. im Big-Data Context
- Spark unterstützt eine Vielzahl von Datenverarbeitungstasks
  - Daten aus verschiedenen Quellen laden
  - Machine Learning
  - Streaming und Batch Jobs
- Unified bedeutet, dass die verschiedenen APIs / Programmiersprachen konsistent und die Daten optimiert verarbeitet werden. Computing Engine bedeutet, dass I/O vom Speicher getrennt ist.
- Im Standard: Spark SQL, MLLib, Streaming und GraphX, aber erweiterbar durch viele open-source third-party Bibliotheken

Spark
Big Data als Motivation

- Computing vs. Data
  - CPUs und Storage werden nicht in gleichem Ausmaß billiger / besser
  - Mehr Daten zu speichern ist für Unternehmen zunehmend einfacher und wichtiger. Die CPU-Leistung steigt aber nicht in gleicher Art und weise
- Durch die zunehmende Menge an Daten müssen diese verteilt und parallel verarbeitet werden

Parallel Computing / Hadoop
MapReduce

- 2004 von Google für das GDR (Google Data Format) für die Verarbeitung von großen Datenmengen auf mehreren Rechnern entwickelt
- **Map-Phase:**
  - Zerlegt Daten in kleinere Stücke
- **Shuffle-Phase:**
  - Sortiert und gruppiert die Schlüssel-Wert-Paare
- **Reduce-Phase:**
  - Aggregiert die gruppierten Daten



Spark
Herkunft und Motivation

- 2009 als Projekt an der UC Berkeley gestartet
- Bis zu diesem Zeitpunkt war MapReduce das gesetzte Verfahren bei der Verarbeitung von großen Datenmengen
- MapReduce ist eher in effizient für große Applikationen und Machine Learning und arbeitet nicht In-Memory

- Spark Entwicklungsstufe 1
  - Eine einfache funktionale Programmier-API in Scala
  - Optimiert für Multi-Step Applikationen
  - In-Memory Computation und Data Sharing über mehrere Knoten hinweg

Spark
Herkunft und Motivation

- Spark Entwicklungsstufe 2
  - Weiterentwicklung mit ad-hoc Ausführungen bzw. für interaktive Data-Science Umgebungen (Jupyter NB)
  - Spark Shell und Spark SQL
- Spark Entwicklungsstufe 3
  - Gleiche Engine, aber neue Bibliotheken für ML, Streaming, GraphX bzw. als offenes Ökosystem
- Spark ist heute die weltweit populärste Data Processing Engine im Big-Data Umfeld und wird von unzähligen Unternehmen verwendet. Alle US-Hyperscaler und auch andere Anbieter wie z.B. Databricks bieten Spark Cluster als PaaS Plattform an
- Als Apache Projekt sehr gut dokumentiert und gepflegt und die Basis für viele andere Computing Engines
- Spark ist kein Teil von Hadoop und hat auch mit den Datenquellen (S3, Azure, Hdfs, files, ..) nicht direkt etwas zu tun

Spark
RDDs (Resilient Distributed Datasets)

- Grundlegende Datenstruktur von Spark
- Unveränderliche, verteilte Sammlung von Objekten
- Unterstützt Fault Tolerance durch Lineage und Wiederherstellung

Spark
DataFrames

- Verteilte Sammlung von Daten, organisiert in benannten Spalten
- Ähnlich wie Pandas DataFrames, jedoch optimiert für verteilte Verarbeitung
- Unterstützt SQL-ähnliche Abfragen und Abfragen über Spark SQL

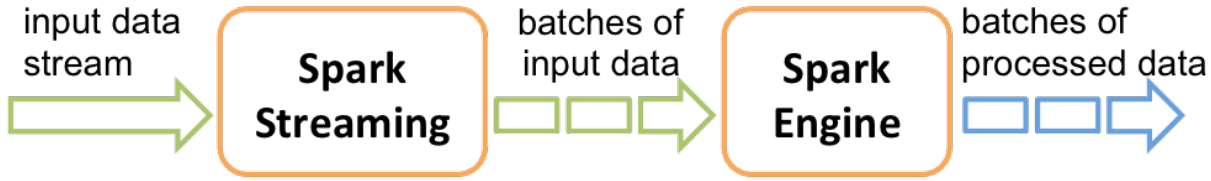
Spark
Datasets

- Kombination der Vorteile von RDDs und DataFrames
- Stark typisiert, bietet Typensicherheit für Entwickler
- Ermöglicht sowohl Abfragen als auch Transformationen auf strukturierte Daten

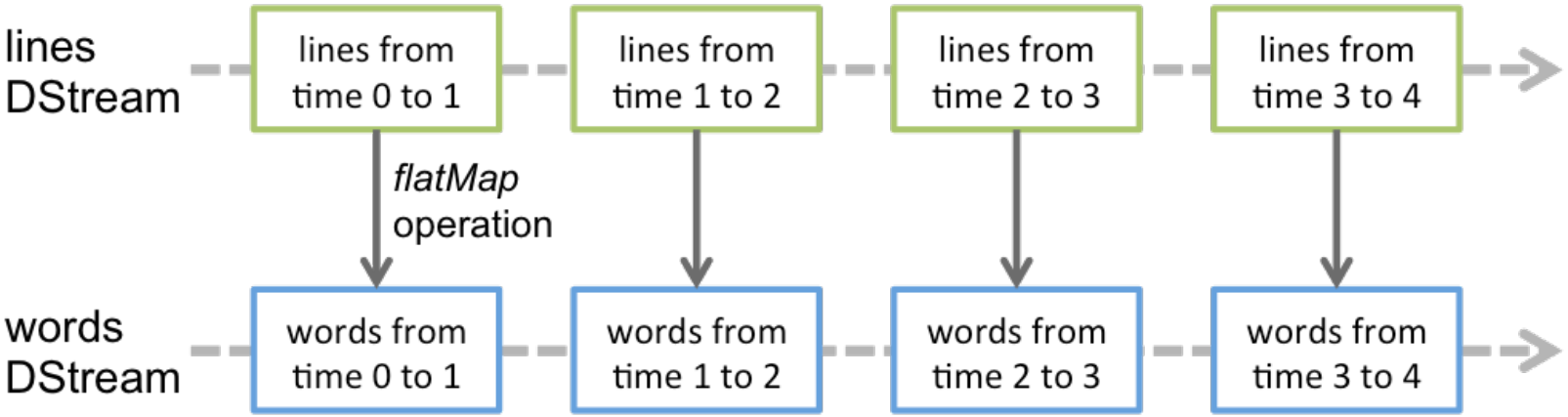
Spark
Spark SQL

- Modul zur Verarbeitung von strukturierten Daten
- Bietet eine SQL-Schnittstelle für Datenabfragen
- Unterstützt Konnektoren zu verschiedenen Datenquellen (z.B. Hive, Parquet)

Spark Streaming



- Echtzeit-Datenverarbeitung
- Verarbeitung kontinuierlicher Datenströme (z.B. von Kafka, Flume)
- Unterstützt Micro-Batch-Verarbeitung und kontinuierliche Verarbeitung



Spark
Machine Learning (MLlib):

- Bibliothek für maschinelles Lernen
- Bietet Algorithmen für Klassifikation, Regression, Clustering und mehr
- Unterstützung für Pipelines, zur Vereinfachung von Datenvorverarbeitung und Modellentwicklung

Beispiel Recommendation Engine:
https://spark.apache.org/docs/latest/ml-collaborative-filtering.html

Spark
GraphX

- Framework zur Verarbeitung und Analyse von Graphdaten
- Bietet APIs für graphbasierte Berechnungen
- Kombiniert RDD-Operationen mit graphenorientierten Operationen

Spark
Basic Architecture

- application

- high-level APIs

- low-level APIs

innFactory

Übung

Übung
Spark

- Starter-Code eingecheckt unter https://github.com/innFactory-Classrooms/afps/tree/main/vl08/spark
  1. Spark 3.5.5 erfordert Scala 2.13.0 und Java 11
  2. Spotify Dataset von Kaggle downloaden: *https://www.kaggle.com/datasets/asaniczka/top-spotify-songs-in-73-countries-daily-updated/data und unter resources/data/ ablegen.*
- Übung:
  - Übungsbeispiele „SpotifyTop50Germany“ ausführen.
  - Versucht mit der Spark UI das ausgeführte Graph bzw. die Stages zu verstehen (http://localhost:4040)
  - Erstellt auf Basis der beiden Beispiele eine Analyse für „SpotifyTop10DEPerMonth“, mit der Ihr die Top 10 in Deutschland für jeden Monat ausrechnet.
  - Bonus: Spotify hat ein eigenes Scala Framework für solche Cases, das auf Apache Beam aufsetzt. https://github.com/spotify/scio Zur weiteren Übung könnt ihr das Beispiel nochmal in Scio implementieren.