DWH

Лекция №5:

Hadoop. Spark

В прошлой лекции

. . .

Hive

Плюсы:

- SQL-подобный язык
- MapReduce под капотом. Уходит много лишней работы, связанной с написанием MR-задач: описание моделей данных, входных и выходных форматов, цепочек MR-задач.
- Интерактивность. Хорош для анализа данных в разных срезах.
- Быстрота разработки
- Отсутствие зависимостей, компиляции, сборки

Минусы:

- Не всё хорошо ложится в парадигму SQL

Форматы хранения файлов

ORC – для горячих данных

Parquet – для холодных данных

AVRO – для хранения данных в последовательном виде

Spark

Spark

Apache Spark – фреймворк с открытым исходным кодом для высокопроизводительных, универсальных, распределенных вычислений.

Spark. Особенности

- Spark – одна из самых быстрых систем среди аналогов.
- Высокоуровневое, относительно простое в использовании API
- Почти все вычисления производятся в оперативной памяти.
- Иногда требуется тонкая настройка.
- Доступны API для Scala, Java, Python, R
- Интерактивная консоль (spark-shell, pyspark)

Spark. Вспомним MR

Задача: Вывести топ-N самых встречающихся слов в документе

Map Reduce:

Spark. Главное отличие от MR

Задача: Вывести топ-N самых встречающихся слов в документе

Смысл тот же самый как и у MR, за исключением, что Spark старается все держать в оперативной памяти, не сохраняя промежуточные данные на hdfs или локальные диски.

Map Reduce:

Spark:

Spark. Варианты запуска

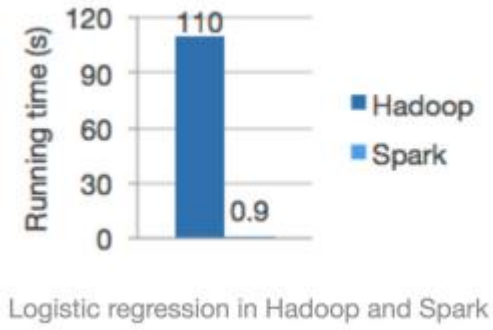
Spark может работать:

• Локально (local[K]). K – количество ядер.

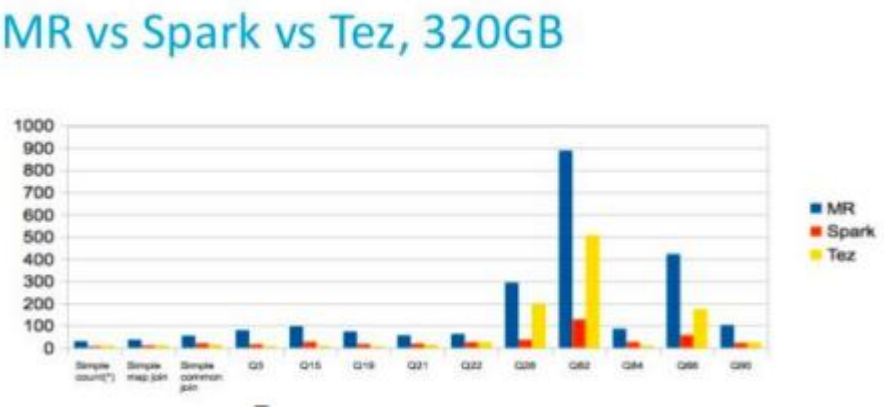
• Standalone – автономный диспетчер кластера (часть Spark). Требует установк park в всех узлах кластера.

• С менеджерами ресурсов Hadoop YARN и Apache Mesos

Spark в сложных задачах сильно быстрее классического Hadoop MapReduce



Logistic regression in Hadoop and Spark



Spark. Архитектура

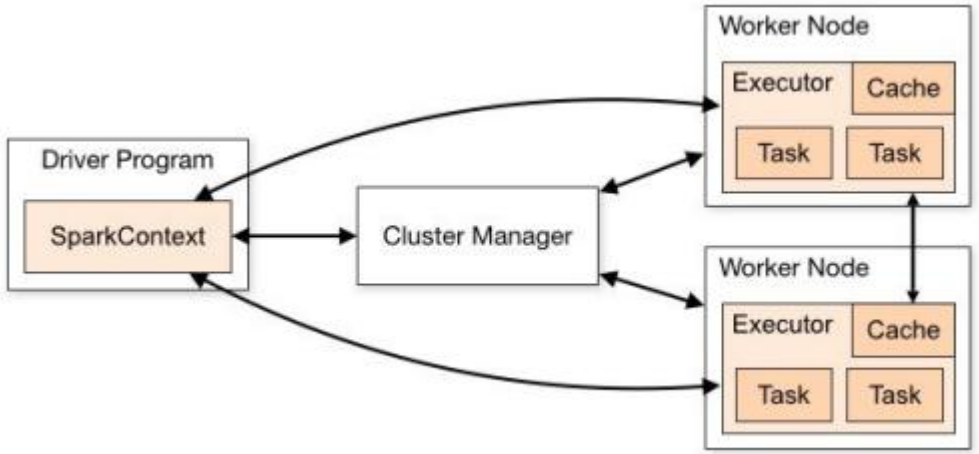
Driver Program – процесс, который выполняет main() функцию приложения и создает SparkContext

SparkContext – это клиент среды исполнения приложения (сердце приложения)

Cluster Manager – менеджер кластера (YARN, Mesos, Standalone)

Worker Node – рабочая нода

Executor - процесс, запущенный на рабочих нодах, который выполняет таски и хранит данные



Spark. Модули

Основные модули:

• Spark Core – основа фреймворка. Обеспечивает распределенное управление заданиями, планирование и базовые функции ввода-вывода.

• Spark SQL – библиотека для структурированной обработки данных. Более эффективные варианты хранения данных, продвинутый оптимизатор и операции над данными.

• Spark Streaming – библиотека для потоковой обработки данных

• Spark ML и MLLib – два пакета машинного обучения

• GraphX – фреймворк для работы с графами

• Также есть множество сторонних библиотек. В частности для GraphX имеется альтернатива - GraphFrames.

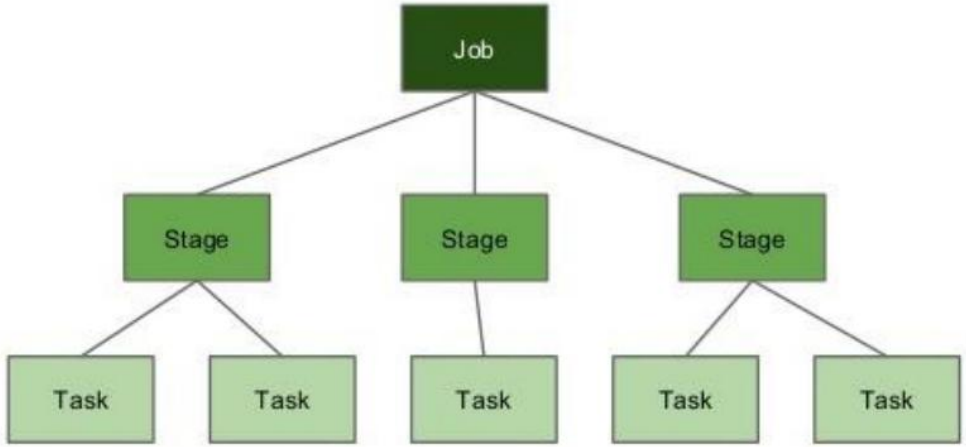
Spark. Иерархия приложения

Application – приложение.

Job – задание, которое генерируется на действие (сохранить на диск, count, collect). У каждого приложения может быть много заданий.

Stage – стадия. Каждое задание делиться на стадии между которыми происходит shuffle.

Task – задача. Каждая задача соответствует одному локальному расчету на executor’е (обработать один блок данных).



Spark. RDD

Ядром Spark является абстракция данных под названием Resilient Distributed Datasets (RDD, отказоустойчивые распределенные наборы данных):

- Resilient – устойчивый к отказу с возможностью восстановления из предыдущих стадий;
- Distributed – распределенный по нодам;
- Datasets – партицированные наборы данных из базовых примитивов.

Особенности:

- RDD вычисляются отложенным образом
- RDD по возможности хранятся в оперативной памяти
- RDD неизменяемы, любое преобразование порождает новый RDD

Spark. RDD. Операции

Базовые операции над RDD:

- Узкие преобразования (не требующие shuffle):

`filter, map, mapPartitions, flatMap…`

- Широкие преобразования(требуют shuffle):

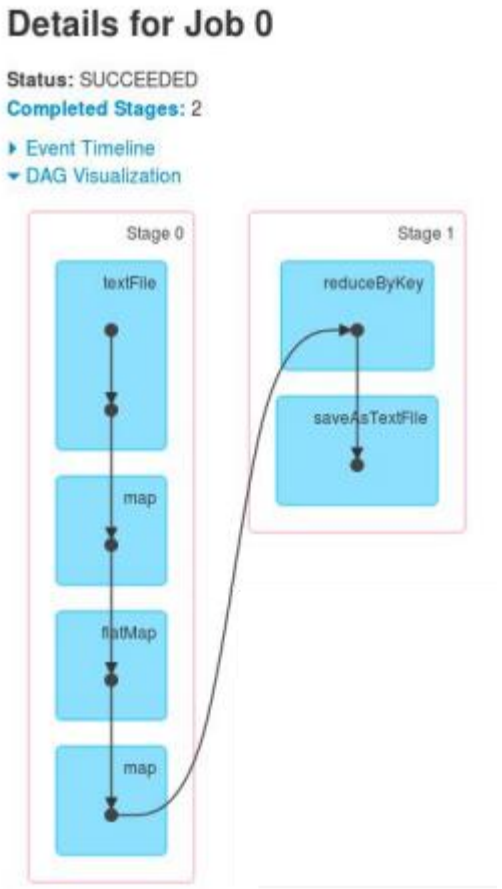
`groupByKey, reduceByKey, sortByKey, sort, …`

- Действия, вызывающие вычисление RDD:

`count, collect, take, sample, max, min, saveAsTextFile, …`

Spark. DAG

DAG (Directed Acyclic Graph) – ориентированный ациклический граф выполнения



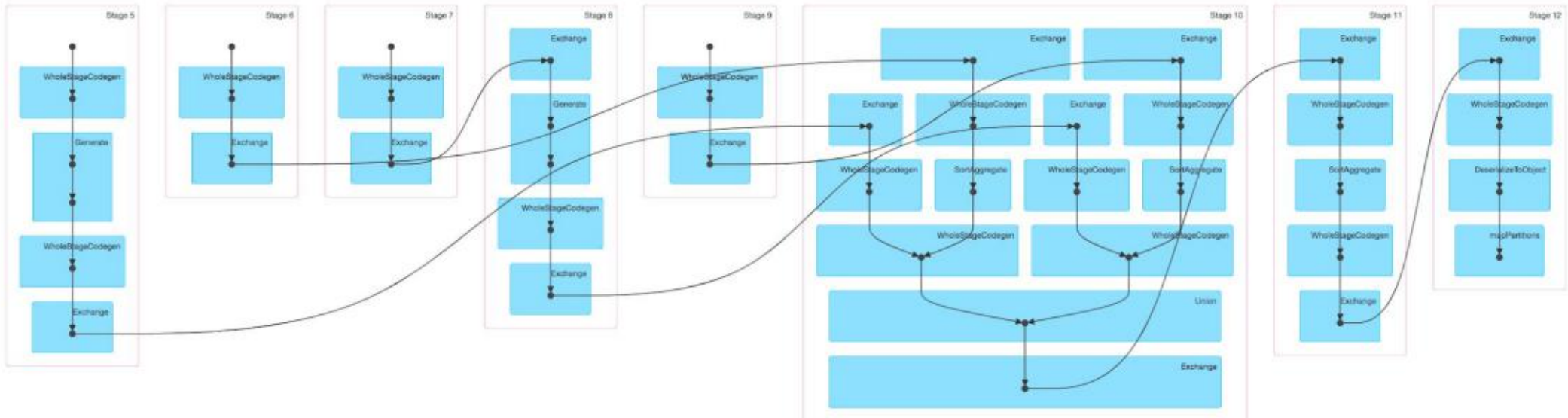

```
val textFile = sc.textFile("hdfs://...")
val counts = (
    textFile
        .flatMap(line => line.split(" "))
        .map(word => (word, 1))
        .reduceByKey(_ + _)
)

counts.saveAsTextFile("hdfs://...")
```

Spark. DAG

Details for Job 5

Status: RUNNING
Pending Stages: 6
Completed Stages: 2

▸ Event Timeline
▾ DAG Visualization



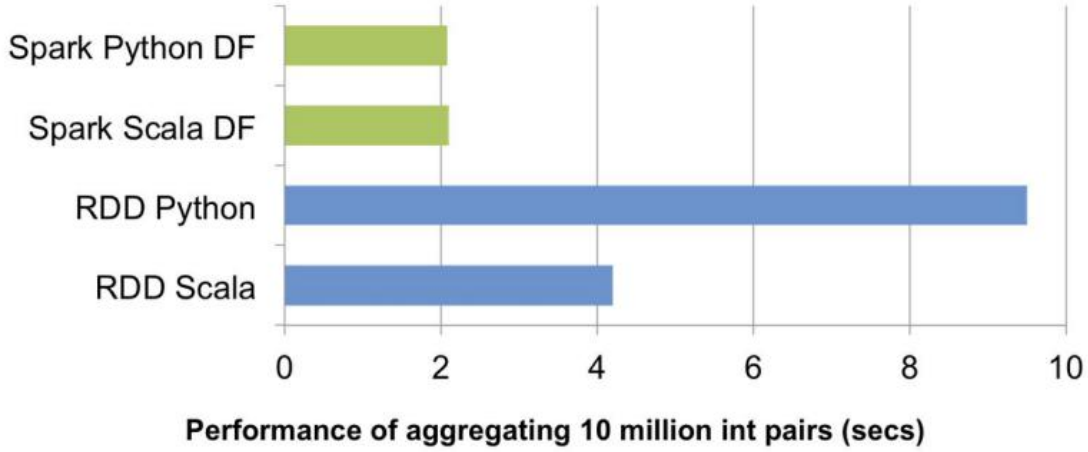
Pending Stages (6)

Stage Id	Description		Submitted	Duration	Tasks: Succeeded/Total	Input	Output	Shuffle Read	Shuffle Write
12	zipWithIndex at SegmentsETL.scala:199	+details	Unknown	Unknown	0/2499				
11	rdd at SegmentsETL.scala:196	+details	Unknown	Unknown	0/2500				
10	rdd at SegmentsETL.scala:196	+details	Unknown	Unknown	0/5000				
9	rdd at SegmentsETL.scala:196	+details	Unknown	Unknown	0/1				
8	rdd at SegmentsETL.scala:196	+details	Unknown	Unknown	0/2500				
7	rdd at SegmentsETL.scala:196	+details	Unknown	3.6 h	0/201				

Spark. Spark SQL

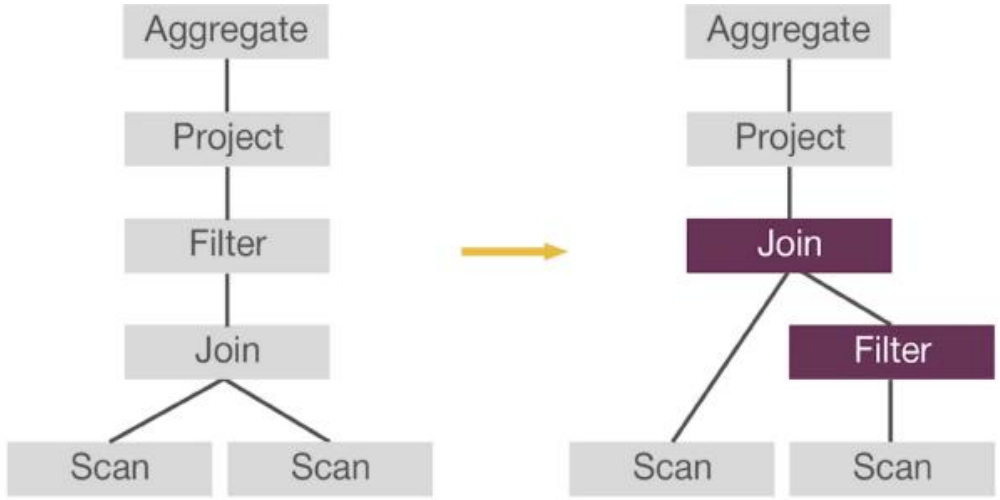
Особенности:

- Позволяет писать SQL запросы аналогично hive
- Удобное SQL-подобное API (DataFrame/DataSet API)
- Интеграция с Hive (можно работать с таблицами из hive metastore)
- Catalyst – оптимизатор планов выполнения (сильно ускоряет приложения)
- Эффективное хранение и примитивные операции (Tungsten)



Spark. Catalyst

```
val serverLog = spark.read.orc("/hdfs/data/logs").as("log")
val userSiteAction = spark.read.orc("/hdfs/data/actions")).as("action")
val badUsers = serverLog
 .join(userSiteAction, col("log.user_id") === col("action.user_id"), "inner")
 .filter(col("logs.code") === "DANGER")
 .groupBy("logs.user_id")
 .agg(count("code").as("cnt"))

badUsers.write.saveAsTable("bad_users")
```



Практика Spark

Spark. Jupyter

Чтобы подключится к jupyter:

• В отдельном окне создаем ssh туннель

```
ssh -D 3128 ваш_логин@89.208.231.195
```

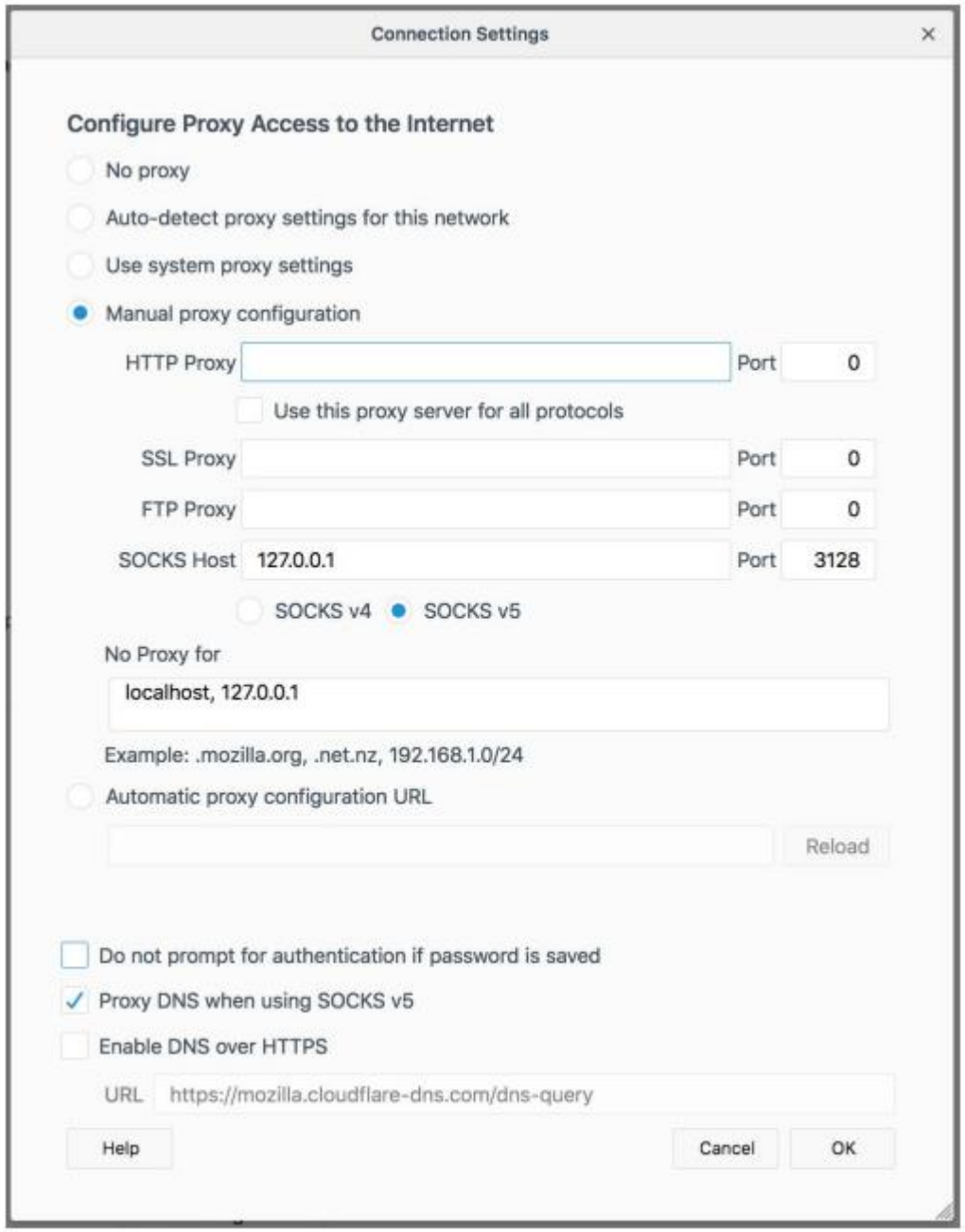
• Устанавливаем firefox и конифгурируем прокси

• Переходим в firefox в jupyter:

http://vk-edu-s202346b22c11-head-0.mcs.local:8000

• Для запуска pyspark notebook

```
New -> PySpark
```



Spark. Ссылки

YARN:

http://vk-edu-s202346b22c11-head-0.mcs.local:8088

Полезная страница с функциями и типами pyspark

https://spark.apache.org/docs/2.3.2/api/python/pyspark.sql.html

Спасибо за внимание!