大数据处理生态系统及其最新进展

孙锐
Intel软件与服务事业部大数据团队

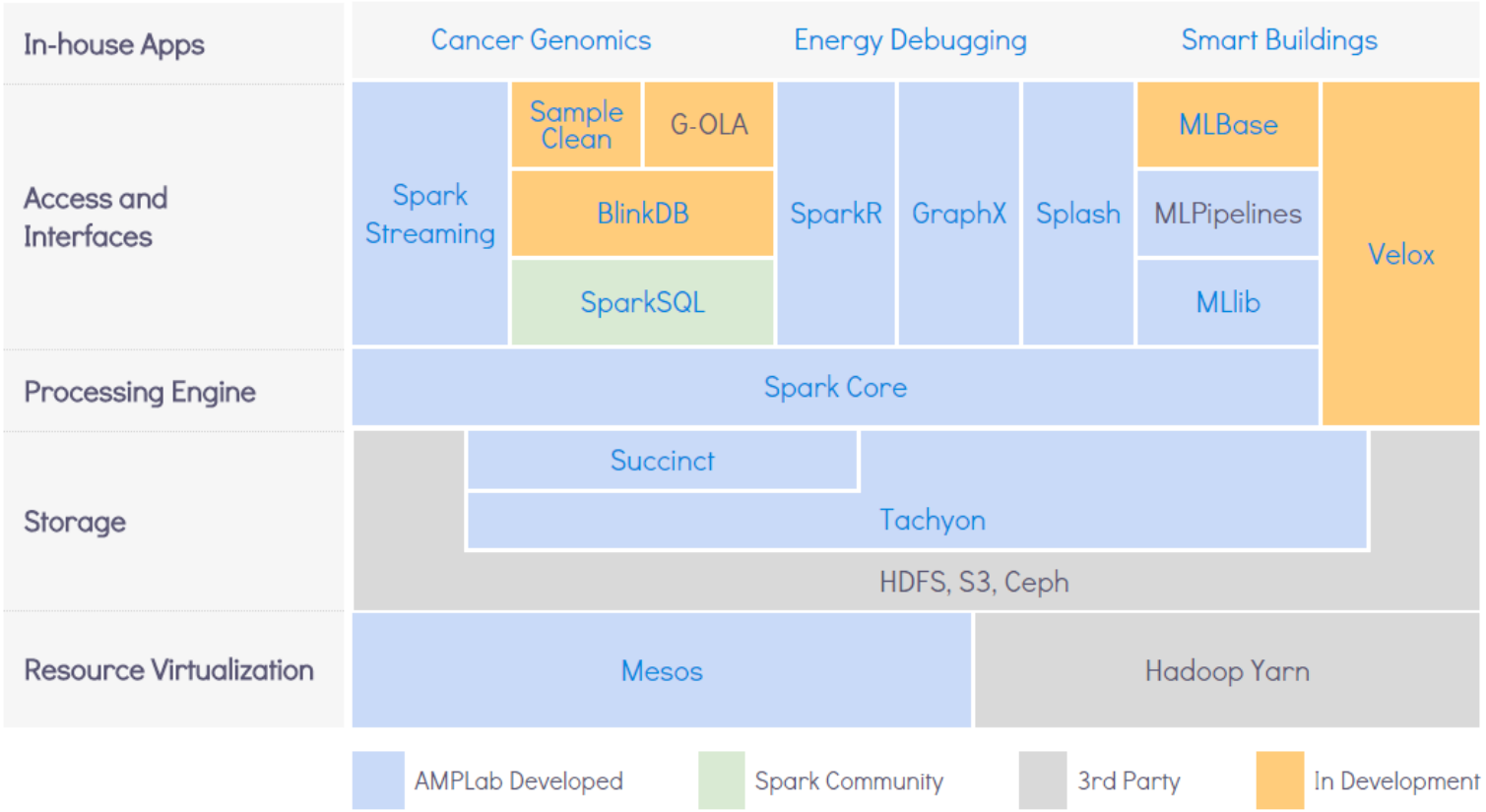
我们是谁

- Intel软件和服务事业部大数据团队
- 国内最早参与Spark的开发和推广
- Spark及相关项目中拥有超过10位活跃的Contributor
  - Spark Core, Spark Streaming, Spark SQL,SparkR, Tachyon

内容概要

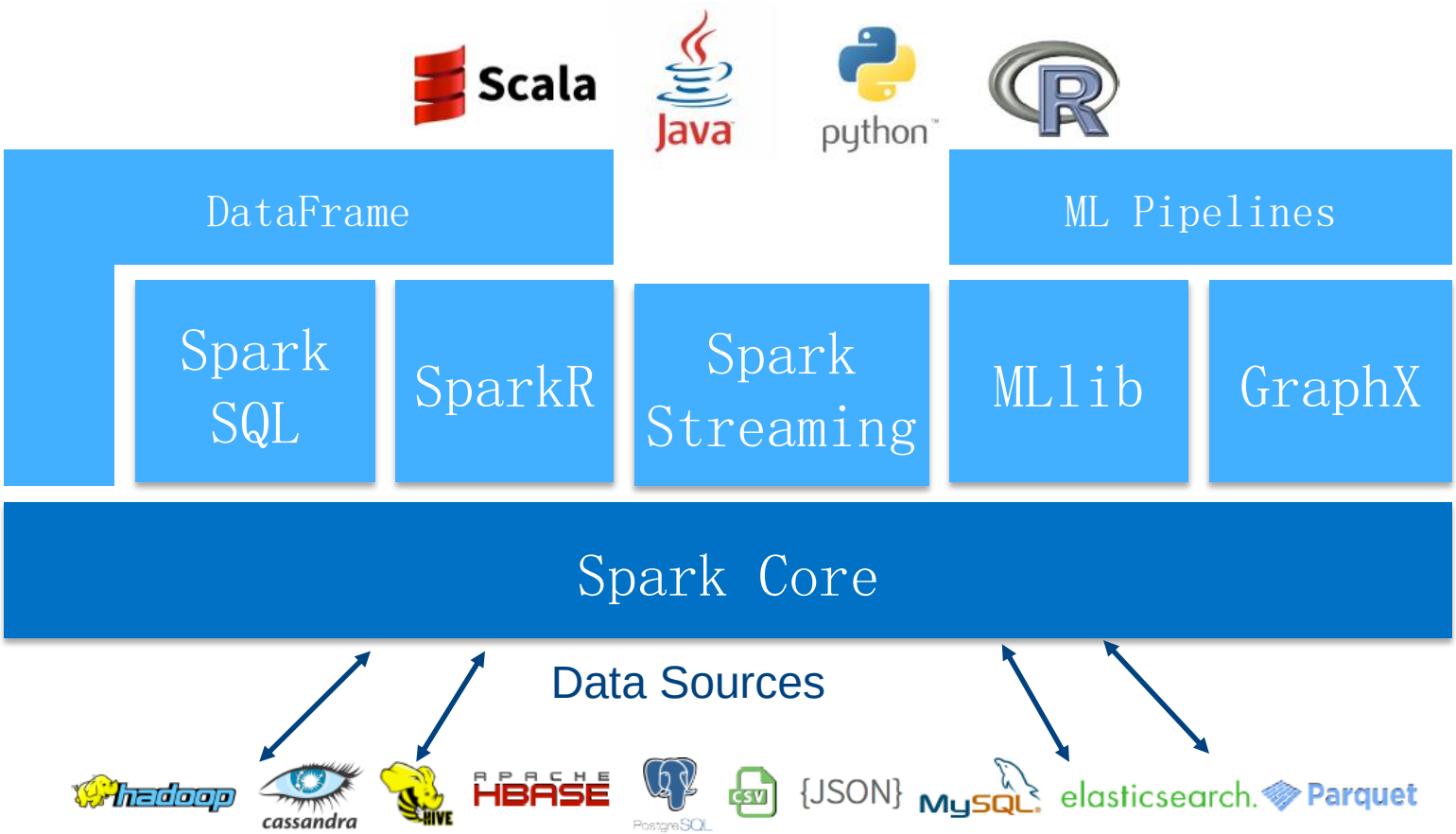
- Spark生态系统概览
- Spark 1.4的新功能、改进
- Intel对Spark生态系统的贡献

BDAS Stack



https://amplab.cs.berkeley.edu/software/

Apache Spark: A Unified Platform



Spark Packages

Databricks维护的网站，列出第三方的Spark相关的库和工具。

构造Spark Package的工具

sbt-spark-package

spark-package-cmd-tool

spark-submit 提供 --packages 选项方便用户使用Spark Package.

例如：

```
$SPARK_HOME/bin/spark-shell --packages com.databricks:spark-avro_2.10:1.0.0
```

http://spark-packages.org/

Spark在快速演进

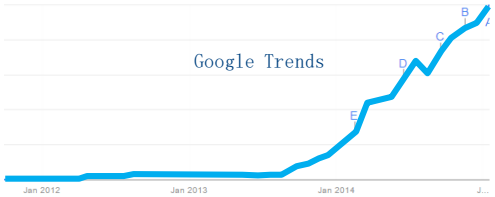
Spark: Most Active Open Source Community

The Apache Software Foundation

35%
Are using/
plan to use Spark †

82%
Users replaced
MapReduce with Spark †

Awareness of Spark

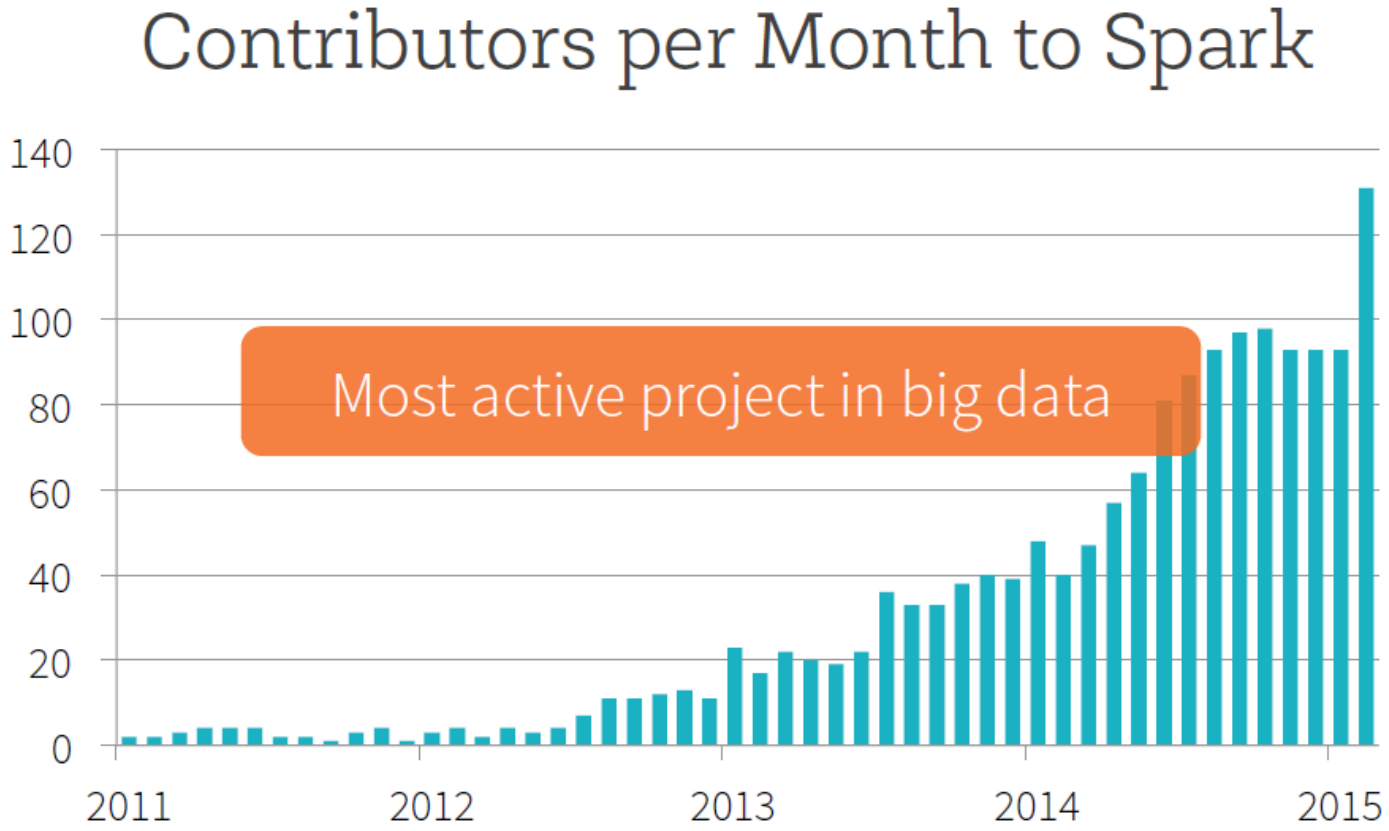


Major Hadoop Distributions & Services are Integrating Spark

amazon web services™ · cloudera · databricks
DATASTAX · EMC² · Hortonworks
IBM · MAPR · ORACLE
SAP · TERADATA

†TypeSafe Apache Spark Survey 2015. *Other names and brands may be claimed as the property of others

Spark社区



https://spark-summit.org/wp-content/uploads/2015/03/SSE15-1-Matei-Zaharia.pdf

Spark社区最新进展

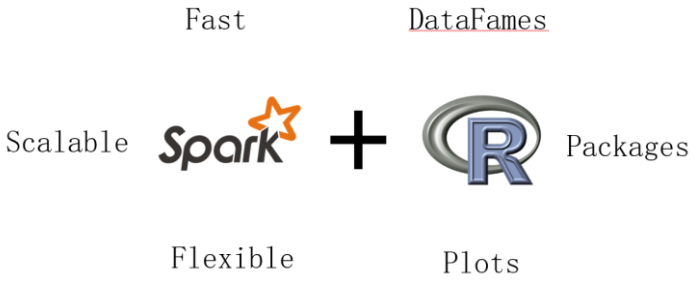
- 1.4.0于6月11日发布
  - 1.4.1于7月15日发布
  - 1.5大概在8月份发布
- Spark Summit 2015 （6月15日）
  - Databricks cloud 演示
  - Spark 1.4的新功能和下一步发展趋势
    - SparkR, DataFrame, Spark Streaming, Spark on YARN, Tungsten
  - Spark的用例分享
    - 来自中国的用例，阿里，腾讯，百度，JD…

Spark 1.4主要亮点

- SparkR
- DataFrame 新API
- ML Pipeline API稳定化
- Spark运行信息可视化，方便调试和监控

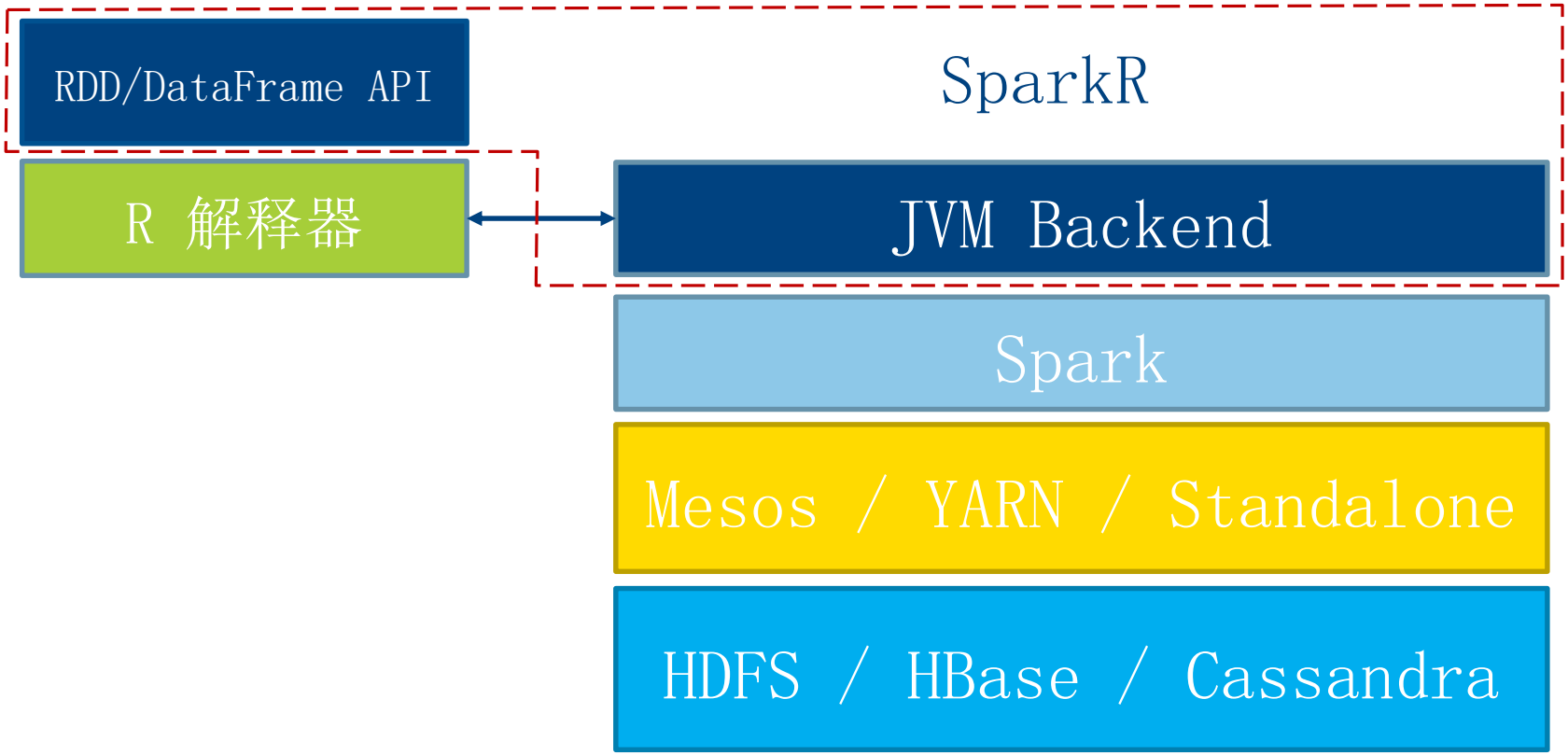
SparkR

- 在Scala/Java/Python API之外增加R语言API
  - RDD
  - DataFrame
- 在R中利用Spark进行分布式计算
  - 克服R的单机瓶颈问题
- RDD API没有公开，推荐使用DataFrame API
  - 支持各种external data source
  - 查询优化，代码生成
  - 性能几乎相当于Scala DataFrame



Formats and Sources supported by DataFrames

SparkR软件栈


RDD/DataFrame API
SparkR
R 解释器
JVM Backend
Spark
Mesos / YARN / Standalone
HDFS / HBase / Cassandra

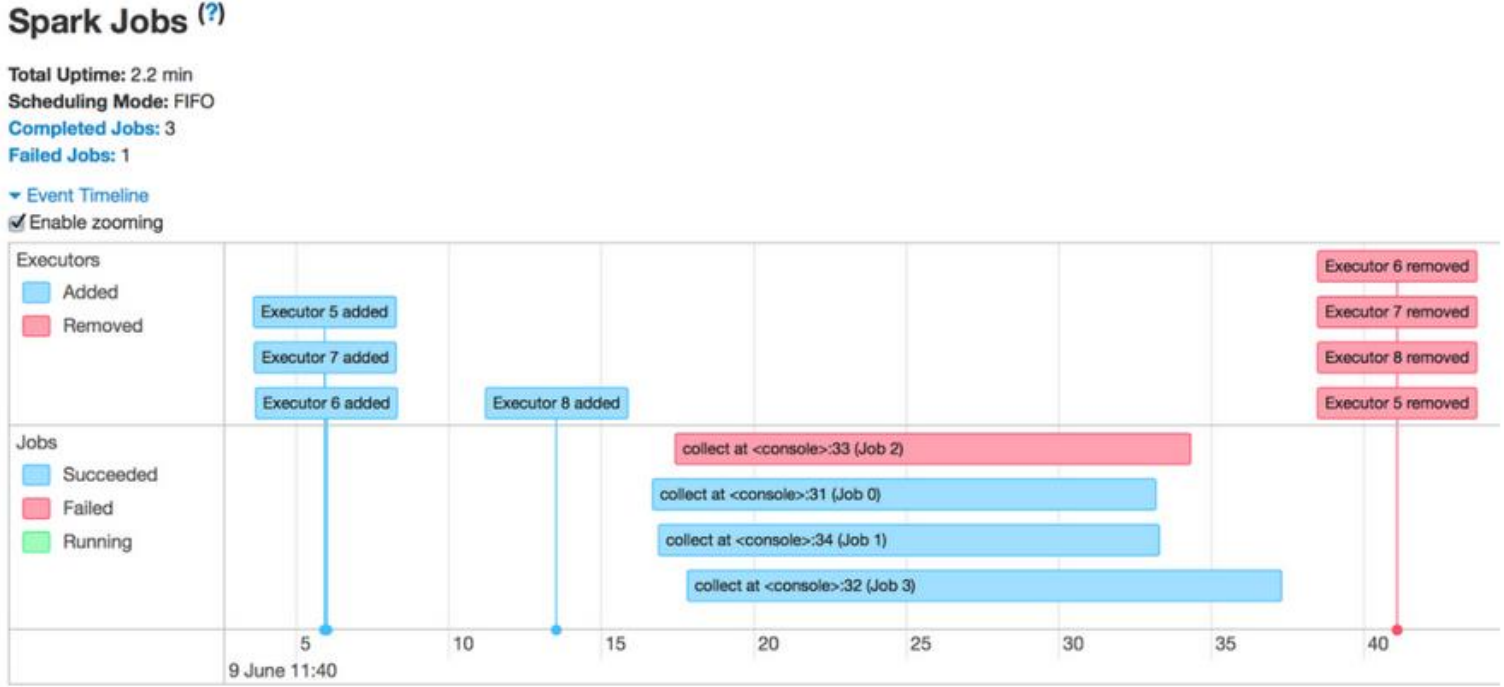
SparkR DataFrame 示例

```
library(SparkR)

#初始化SparkContext和SQLContext

sc <- sparkR.init("local", "AverageAge")
sqlCtx <- sparkRSQL.init(sc)

#从当前目录的一个JSON文件创建DataFrame

df <- jsonFile(sqlCtx, "person.json")
#调用DataFrame的操作来计算平均年龄

df2 <- agg(df, age="avg")
averageAge <- collect(df2)[1, 1]
```

Spark Core

Spark监控信息可视化，方便分析和调试:

按时间轴显示Spark事件

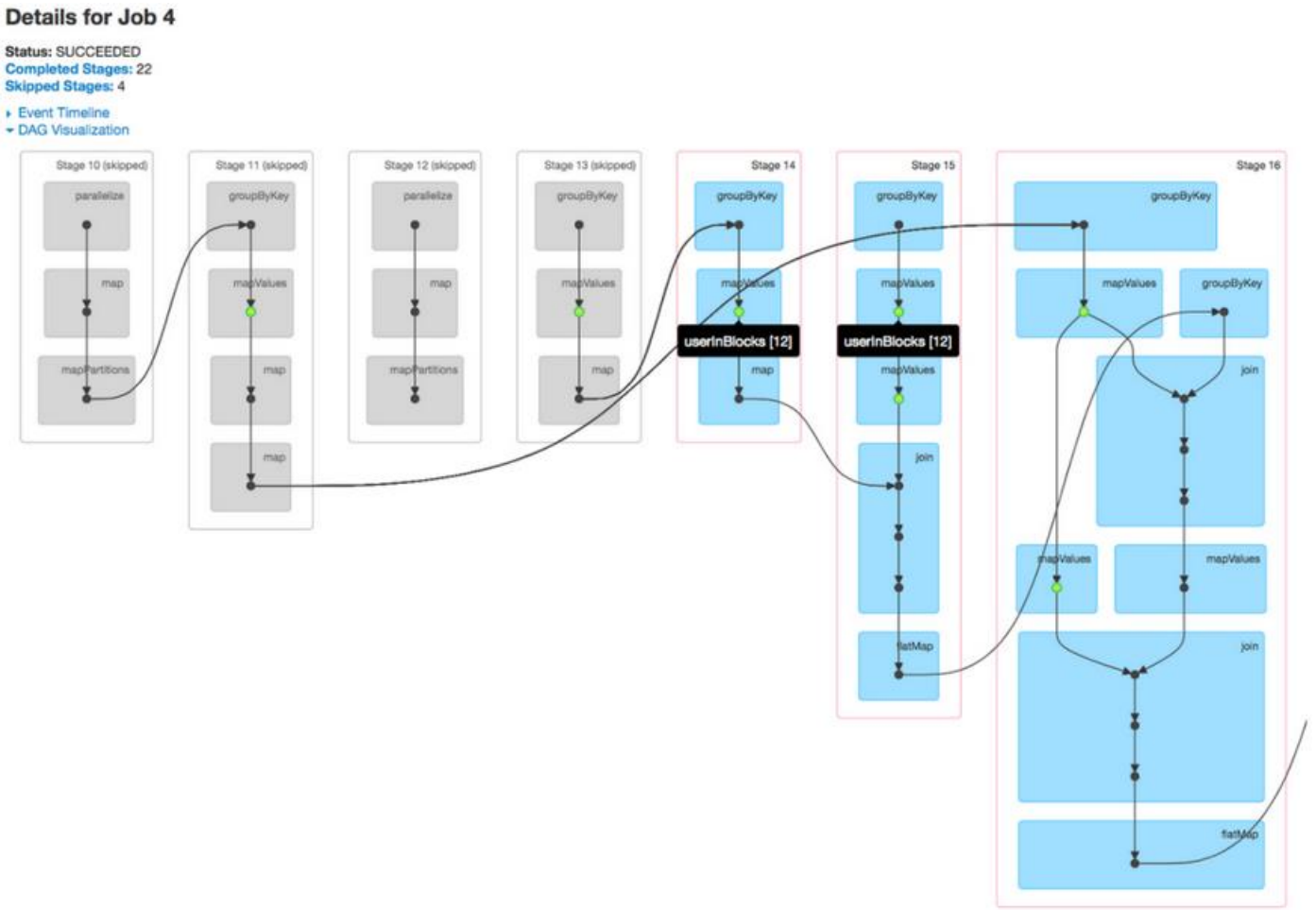


https://databricks.com/blog/2015/06/22/understanding-your-spark-application-through-visualization.html

Spark Core

Spark监控信息可视化，方便分析和调试：

DAG可视化



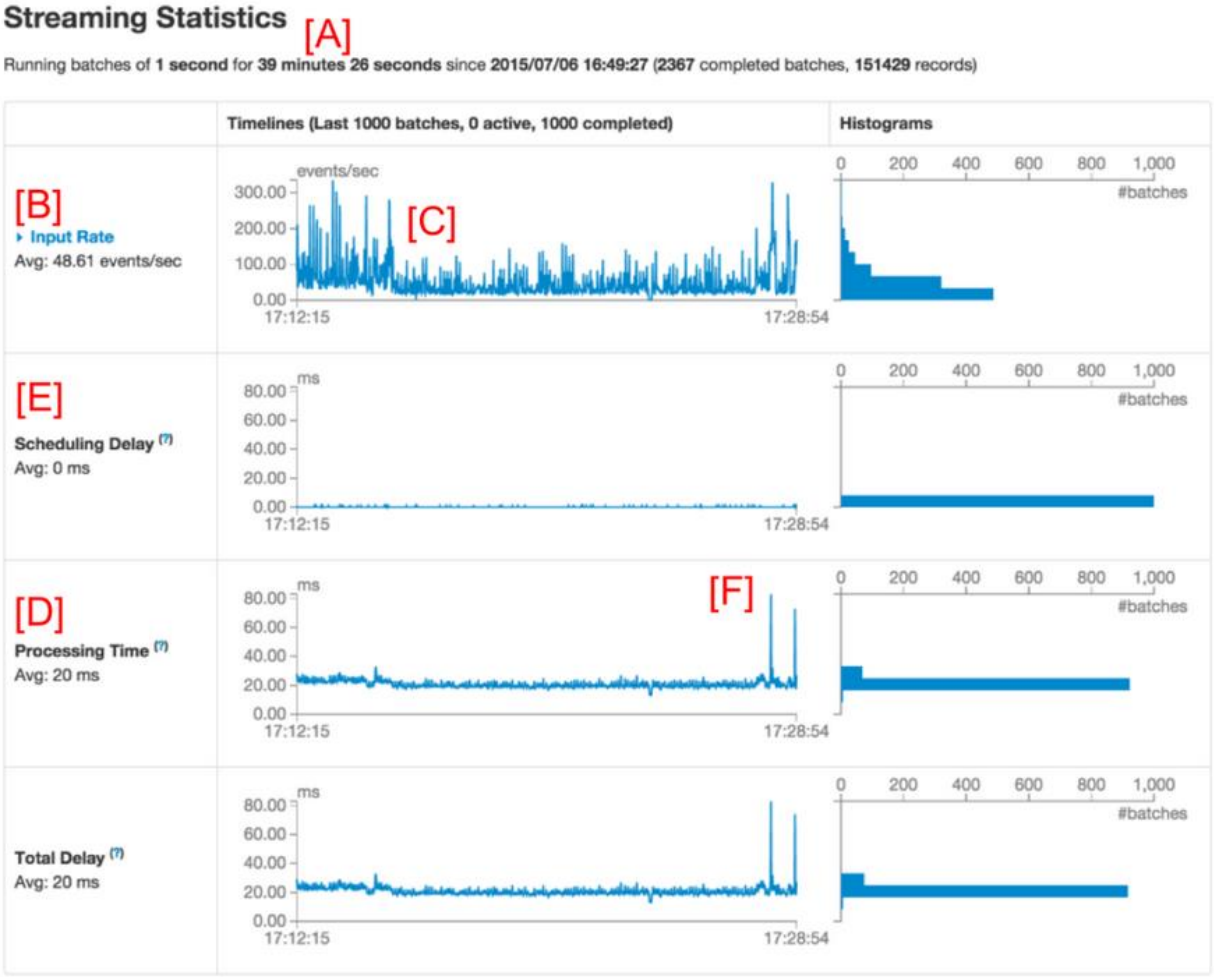
https://databricks.com/blog/2015/06/22/understanding-your-spark-application-through-visualization.html

Spark Core

Spark监控信息可视化，方便分析和调试：

Spark Streaming

统计信息



https://databricks.com/blog/2015/07/08/new-visualizations-for-understanding-spark-streaming-applications.html

Spark Core: Project Tungsten

Databricks发起的旨在改进Spark执行引擎性能的计划，会跨若干个release

- 关键点是提高Spark对CPU、内存的使用效率
- 手段
  - 显式内存管理
  - 算法、数据结构对CPU cache更友好
  - 代码生成

Spark 1.4.0包含了一些早期成果

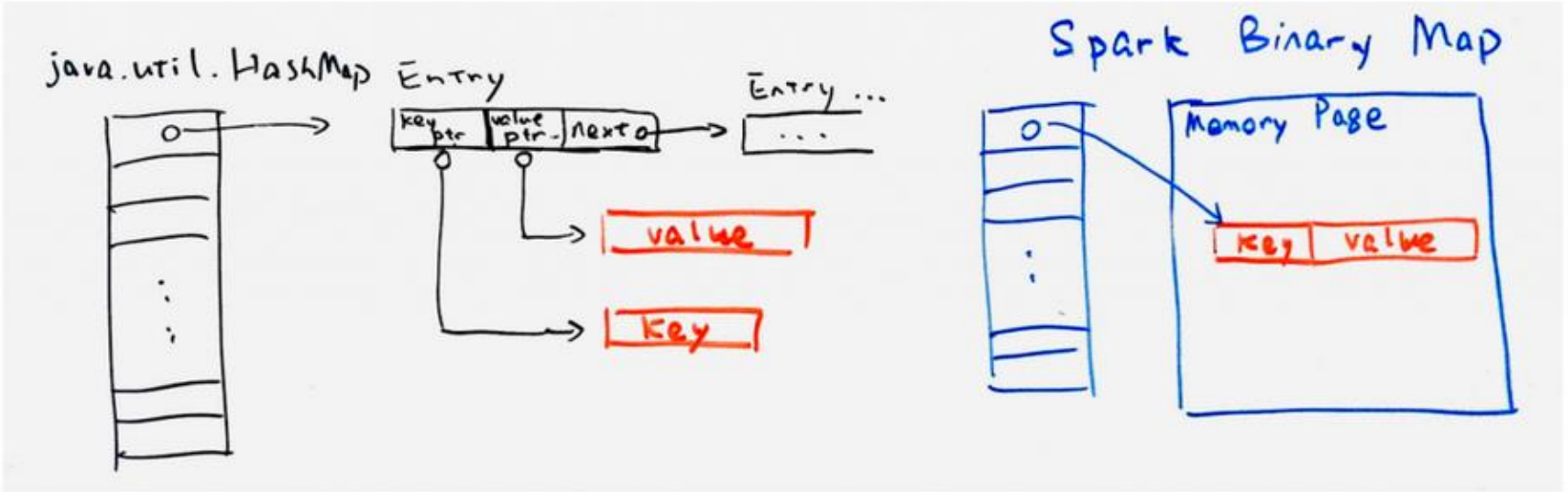
https://databricks.com/blog/2015/04/28/project-tungsten-bringing-spark-closer-to-bare-metal.html

Project Tungsten: 新HashMap

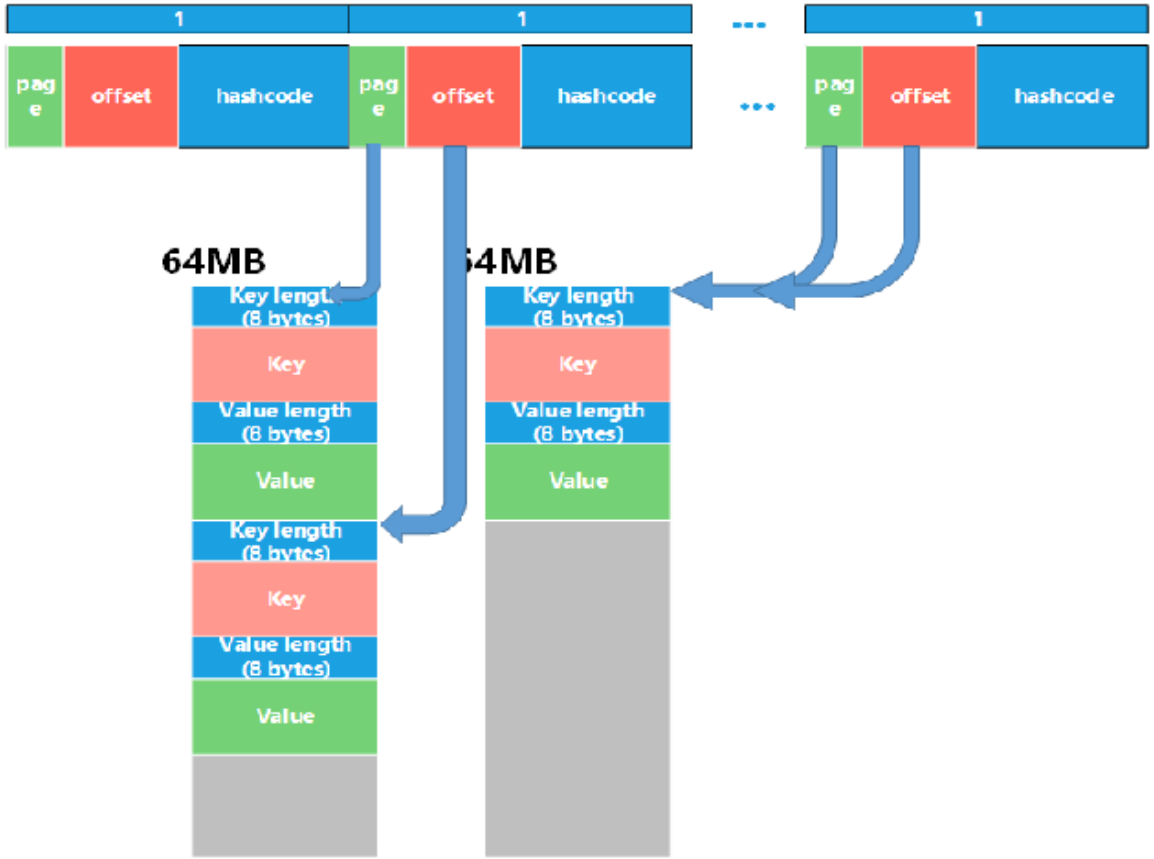
基于显式内存管理实现了自己的Hash Map

与Java标准HashMap相比，数据存储效率更高，减少或不受GC影响

在Spark SQL的aggregation利用此Hash Map



Project Tungsten: 新HashMap

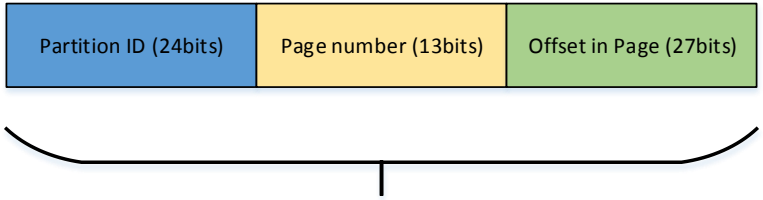


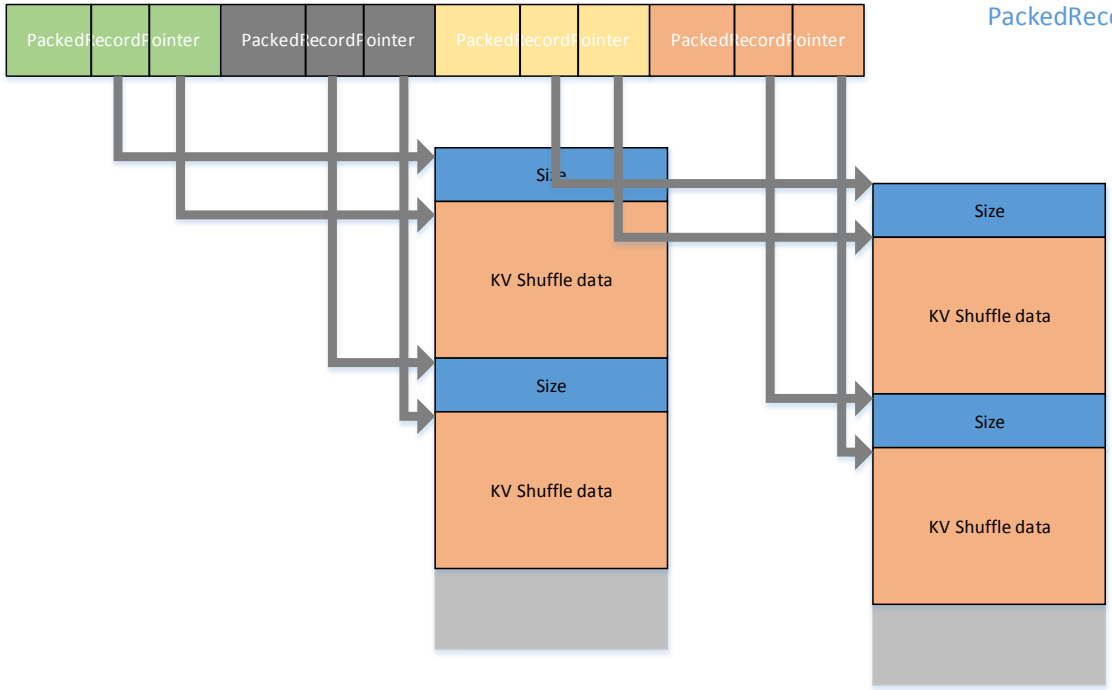
Project Tungsten: Shuffle

Sort-based Shuffle问题：

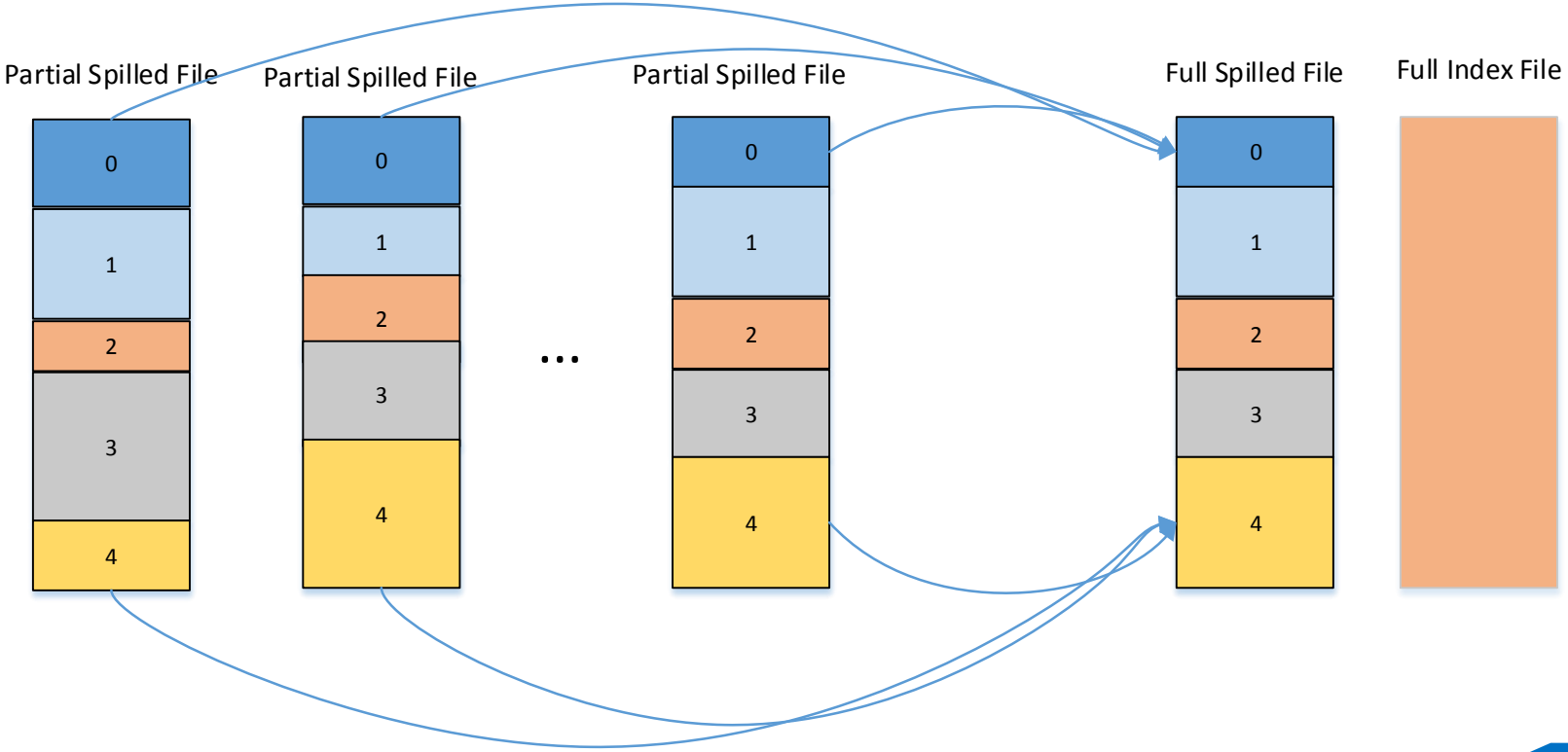
数据结构对CPU Cache不友好

排序、合并需要反复序列化/反序列化Java对象





Project Tungsten: Shuffle



Spark Core

pySpark支持Python 3

pySpark：sortByKey和groupByKey支持external sort

支持Mesos cluster mode

支持Mesos Docker

Spark SQL

- 加入统计函数
  - 计算两列的皮尔森相关系数
  - 计算两列的样本协方差
  - 在列中寻找频繁项
- 支持数学函数
  - 三角函数，对数，幂，指数，平方根等
- 支持窗口函数
  - rowNumber，rank，lag，lead等
- 支持rollup和cube
- 支持ORC文件格式

- 支持sort merge join
- 改进Data Source API
  - 支持partitioned table（自动发现partition，读/写partitioned table）
- DataFrame的Input/Output API改动：引入DataFrameReader、DataFrameWriter

```
val df = sqlContext.read.format(“json”).load(“/path/to/json file”)

df.write.mode("append").format("json").partitionBy("date").saveAsTable("myJsonTable")
```

ML Pipelines、MLlib、Graphx

ML pipelines API接口稳定化

实现了不少新的feature transformer

增加ML pipelines的Python接口

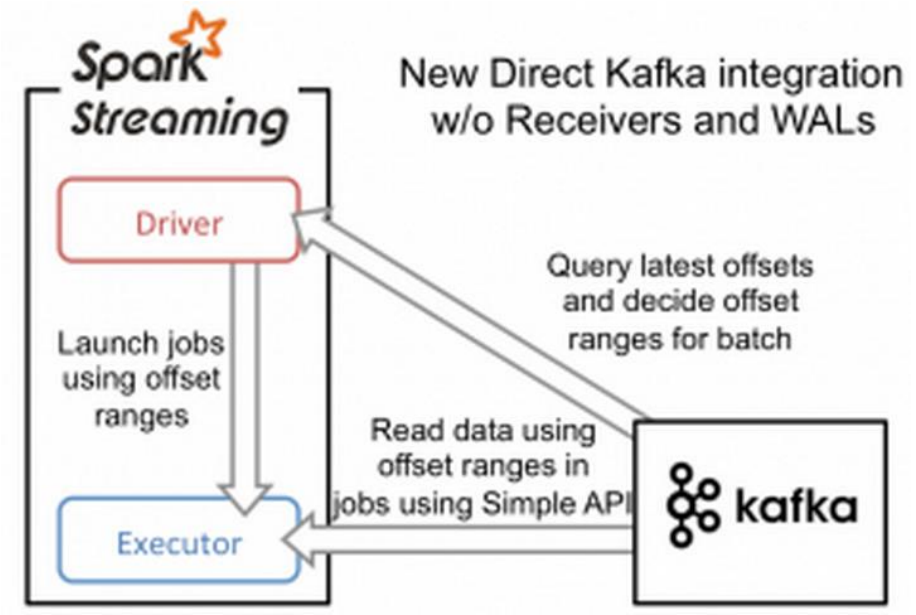
MLlib：

- 增强现有的算法，如逻辑回归，LDA(Latent Dirichlet Allocation)
- 增加伯努利模型的朴素贝叶斯算法
- 支持评估PMML模型

在Graphx中增加个性化PageRank算法

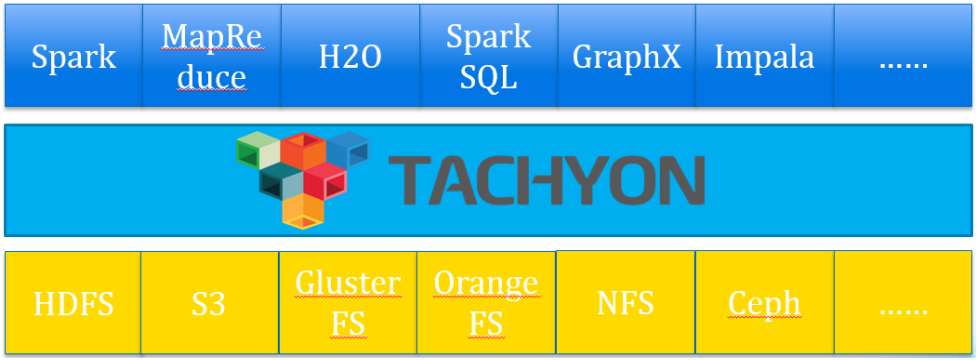
Spark Streaming

- 监控信息可视化
- 提供Kafka Direct Mode的Python接口
- 可插拔的WAL接口（支持不同的存储系统）



Tachyon

基于内存的分布式存储系统



设计思想:

1. 基于内存的OffHeap的分布式存储
2. 通过在存储层保存数据的Lineage实现容错

http://tachyon-project.org

Spark中应用Tachyon

- 用Tachyon作输入、输出、checkpoint
- Cache RDD到Tachyon（Tachyon是Spark中OFF_HEAP的默认存储）

数据共享

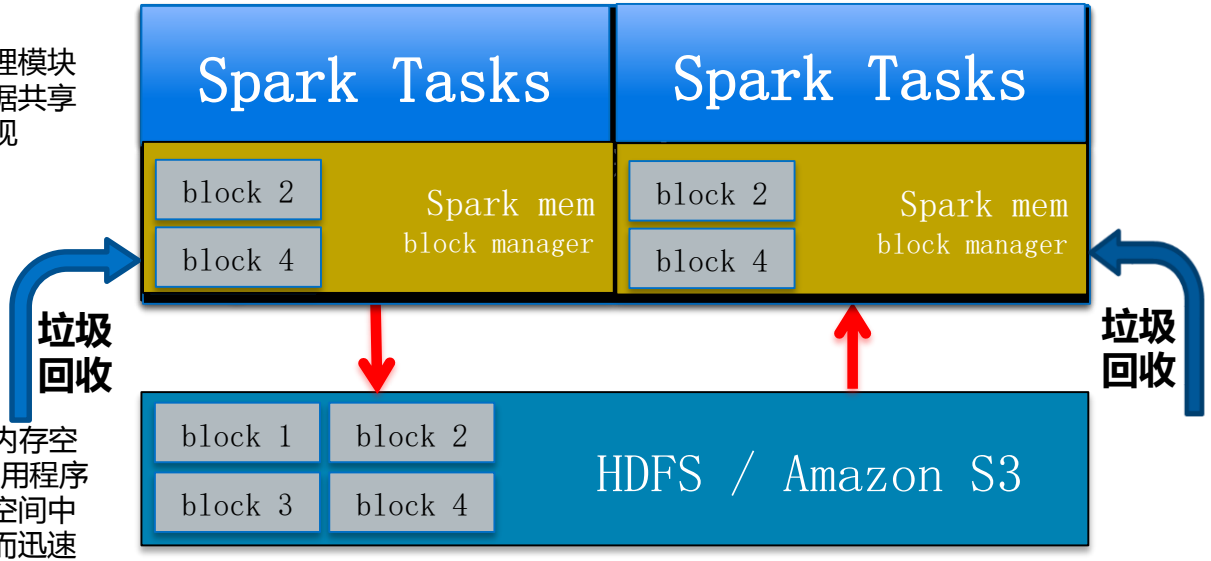
计算引擎和内存管理模块在同一进程中，数据共享需通过共享存储实现(HDFS)

缓存数据丢失

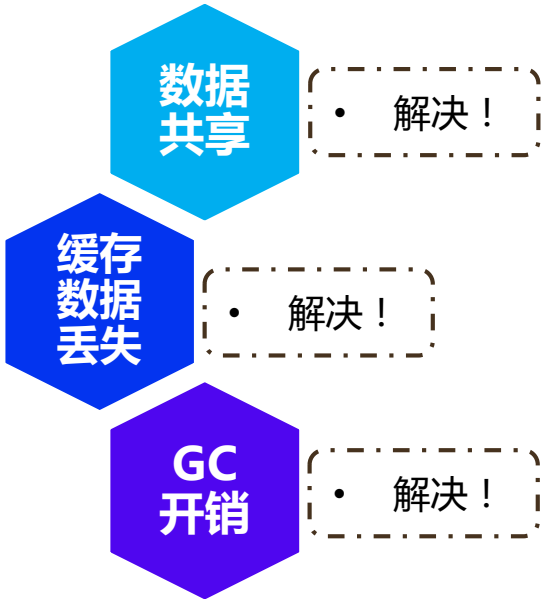
缓存数据在执行引擎JVM堆内存中，执行引擎崩溃，缓存数据丢失。

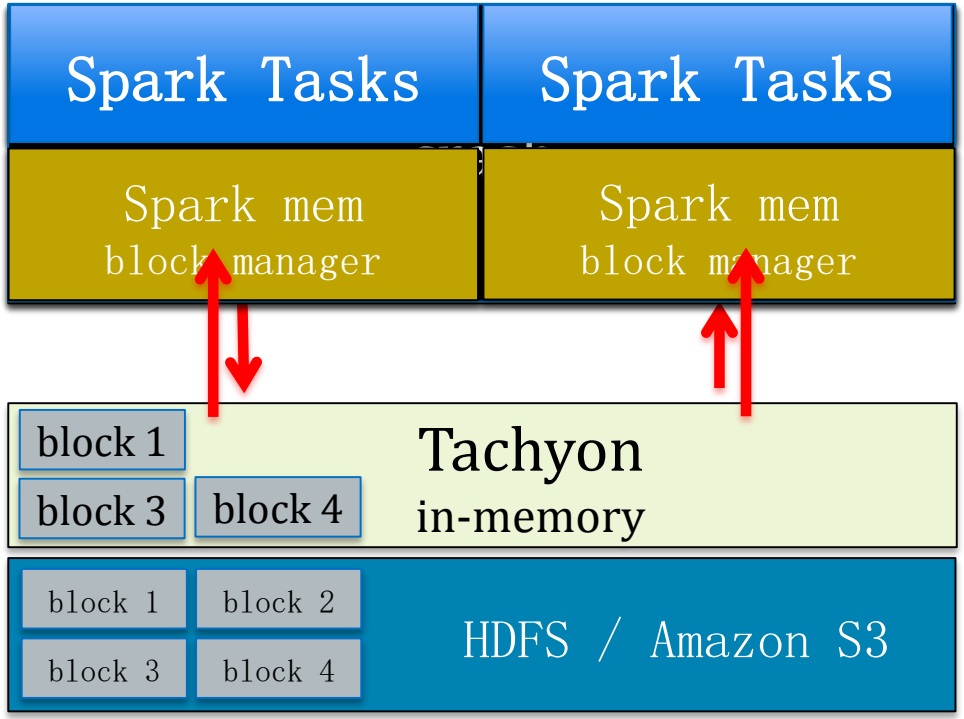
GC开销

数据缓存在JVM堆内存空间中，GC开销随应用程序运行时间和堆内存空间中缓存数据量的增长而迅速增长

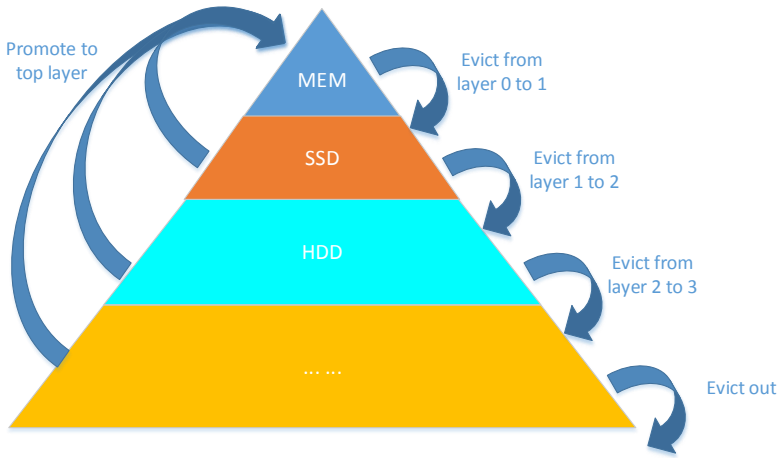


Spark中应用Tachyon





多层级的本地存储



使用SSD/HDD扩展Tachyon存储空间

- 分层结构，不同层有不同的优先级
- 热数据放在顶层存储，不热的数据放在下层存储
- 同一层中可以拥有多个目录

当数据不再“热”时，可以从顶层被移动至下层

- 不同的eviction策略

当数据再次变“热”时，可以从下层重新被移动至上层

http://tachyon-project.org/Hierarchy-Storage-on-Tachyon.html

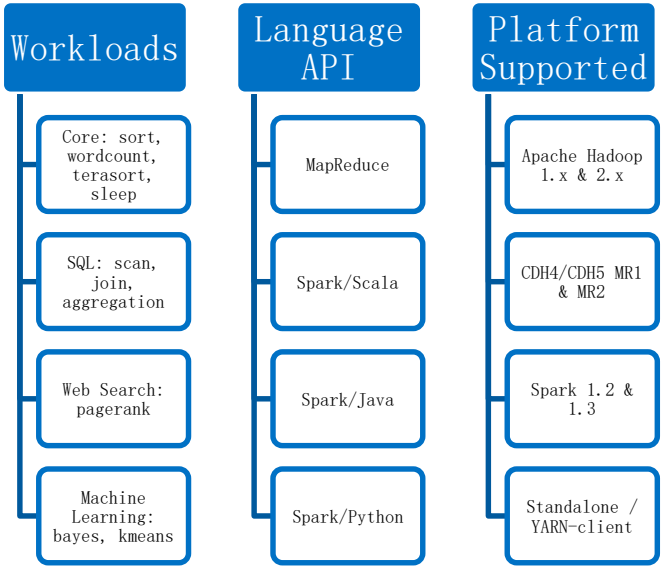
Intel对Spark生态系统的贡献

- HiBench
  - 4.0：支持Spark
  - 5.0(计划)：StreamBench
- Performance Portal for Apache Spark
  - 帮助Spark社区及时发现和修正性能退步
  - 在10个节点的集群上用HiBench, Spark-perf作性能测试
  - 每周发布测试结果
    - http://01org.github.io/sparkscore/result.html
- StreamingSQL

HiBench – 大数据性能测试工具

发布的版本

1.0 (2010)	2.2 (2012)	3.0 (14H2)	4.0 (15H1)
✓ Paper published in ICDE' 10 workshops ✓ 3 Categories, 8 workloads ✓ Internal use and shared 3rd Part under NDA	✓ 2012 open source under Apache 2.0 License ✓ Add Analytical Query Category, 2 hive workloads (join, aggregation)	✓ 2014 Yarn support ✓ Unify MR1 & MR2 to one branch ✓ Add Sleep job ✓ Add concurrent mode	✓ Spark support (scala, java, python) ✓ Unified configuration and reports ✓ Easy visualization report



Baidu 百度 intel Inventec hp HUAWEI SAMSUNG CRAY ...

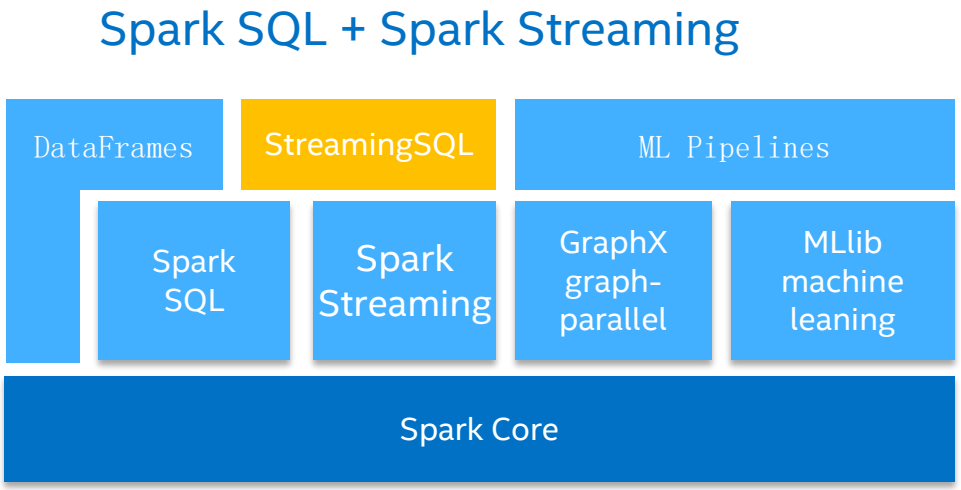
Open source under Apache 2.0 license

https://github.com/intel-hadoop/HiBench

intel Software

Streaming SQL for Apache Spark

- 在Spark Streaming之上提供SQL查询接口
- 快速掌握流计算，降低学习曲线



Open source under Apache 2.0 license

https://github.com/intel-bigdata/spark-streamingsql

StreamingSQL让流计算编程更简单

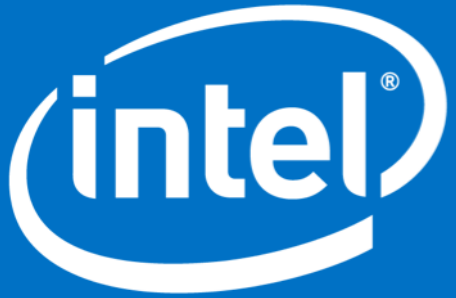
Spark-streaming

```
val weblog= KafkaUtils.createStream(…)
 .map(_._2)
 .flapMap(_.split(“|”)).map(_._2, _)

val category=
  sc.textFile(…).flatMap(_.split(“|”))

val result
  = weblog.transform(r => r.join(category))
   .map(_._2._2, 1L))
   .reduceByKey(_ + _)

result.print()
```

StreamingSQL

```
CREATE TABLE weblog ( sourcep_ip STRING,
cookie_id INT, item_id INT …) USING
stream.source.KafkaSource OPTIONS( zkQuorum
“localhost:2181”, topic “weblog:1”…)

CREATE TABLE category (item_id INT, category
STRING);

SELECT category, COUNT(*) FROM weblog JOIN
category ON category.item_id = weblog.item_id
GROUP BY category.category;
```

Q&A


intel