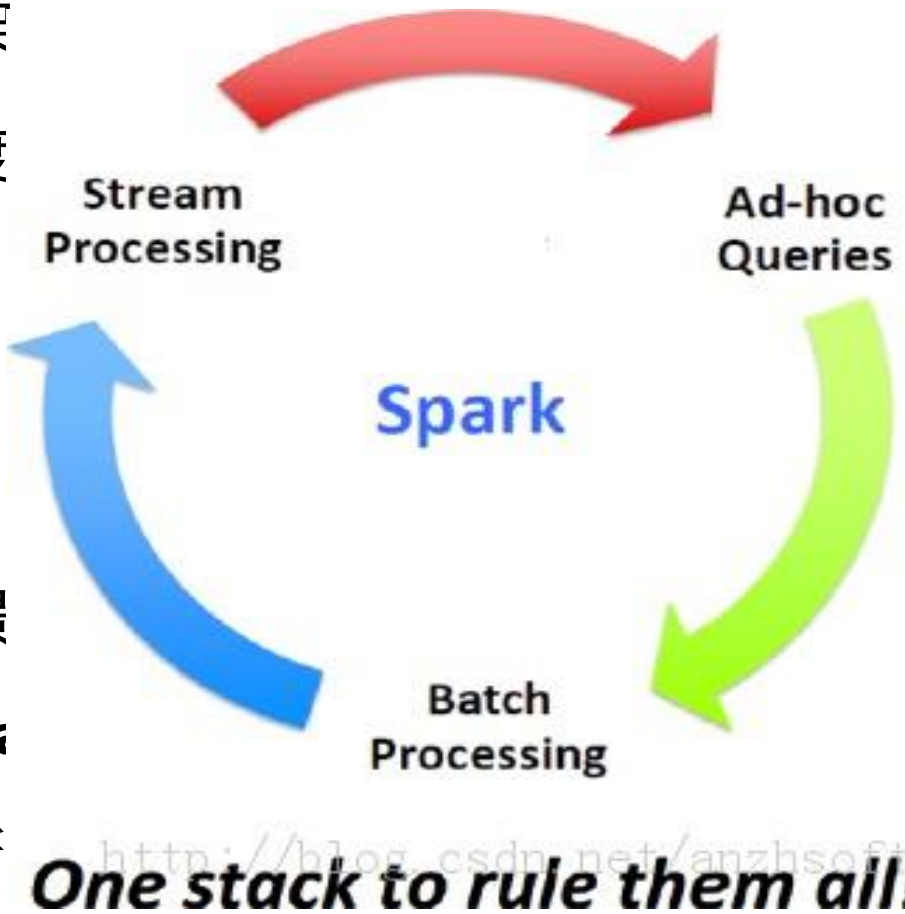
Spark技术内幕

张安站

2015

什么是Spark ？

- 分布式计算框架
- 基于内存的调度
- 兼容Hadoop生
  - 数据存储格式
  - **Works with**
- 丰富的应用场景
  - **batch /strea**
  - **SQL / 机器学**



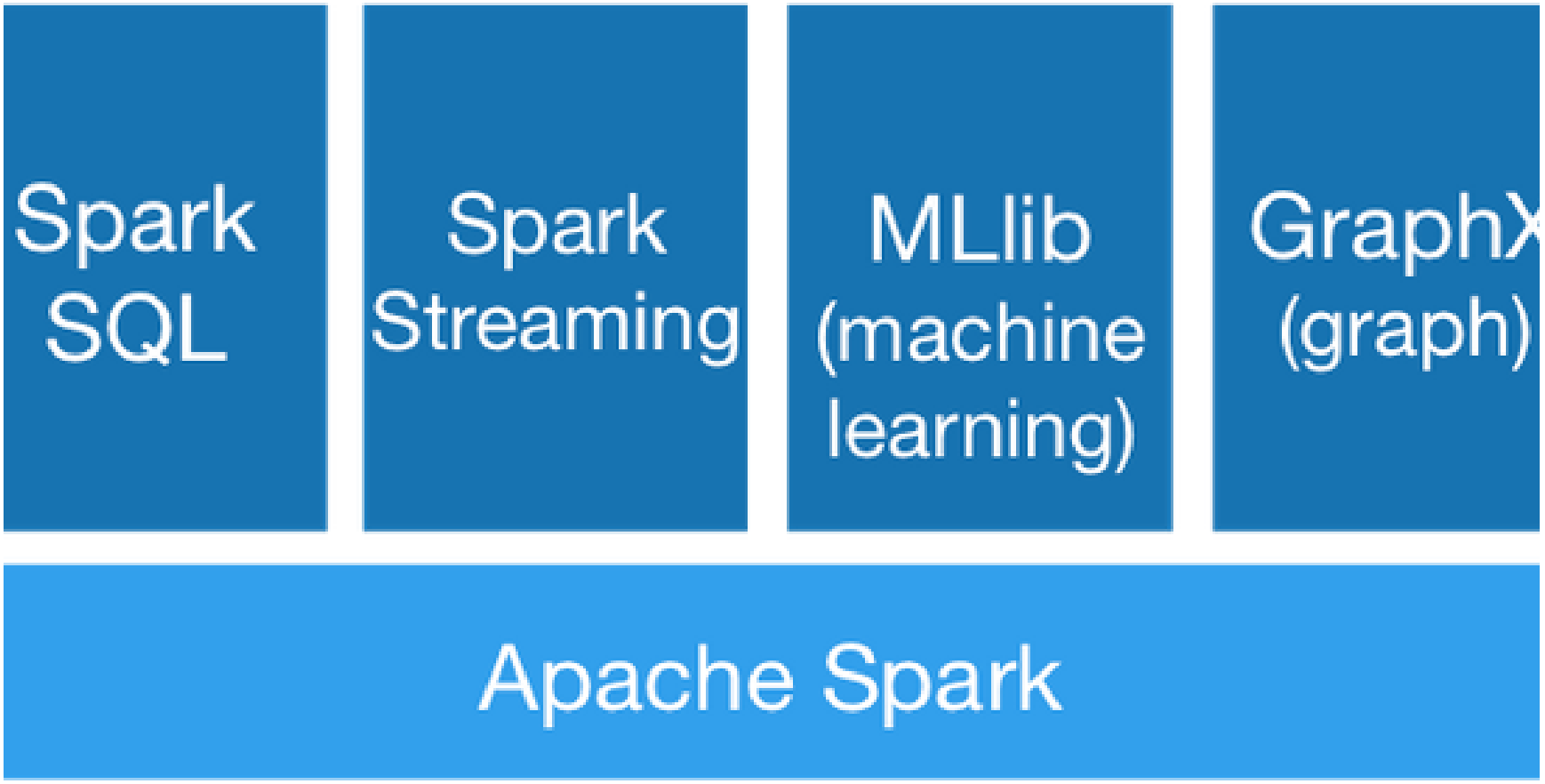
.. etc.

快速发展的生态环境

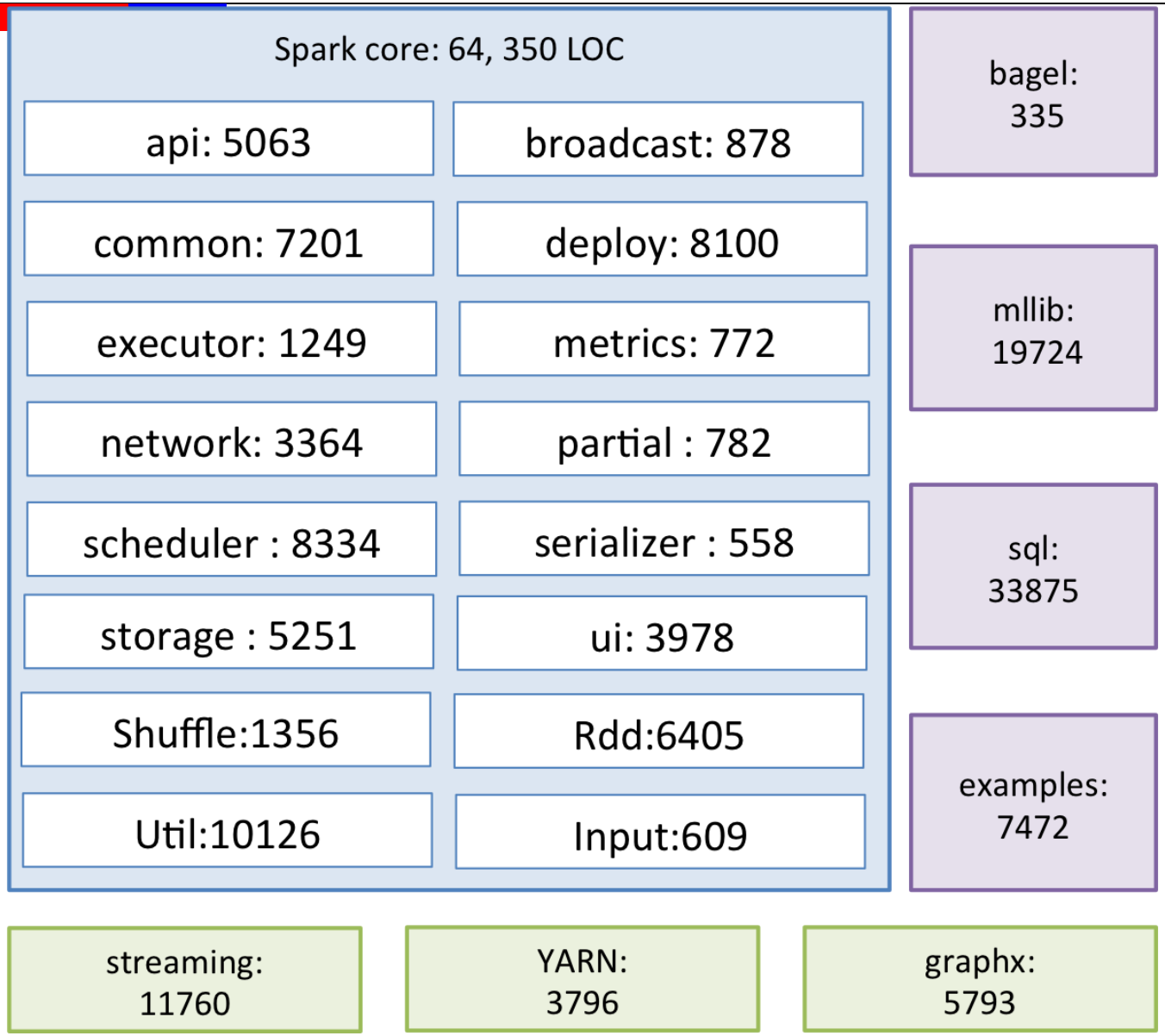
- 快速从Apache Incubator项目毕业成为Apache顶级项目
- 大数据解决方案提供商的支持
  - **Cloudera / MapR / Hortonworks / Pivotal / SAP …**
  - 华为 / 星环科技
- 应用
  - 百度/阿里/腾讯/爱奇艺/优酷/京东…
- 2014年大数据领域最活跃的开源项目

Baidu 百度

核心组件

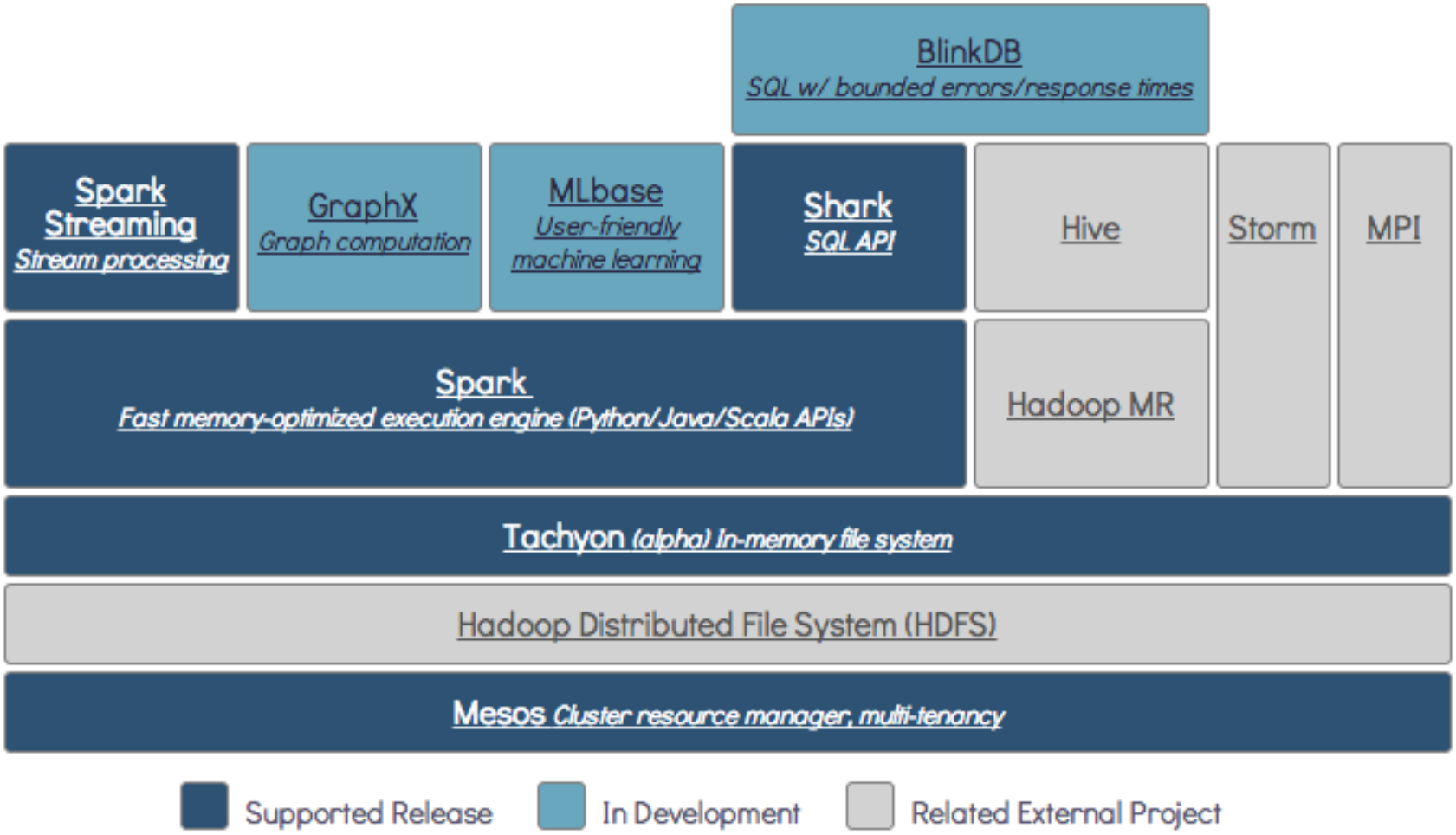


代码规模

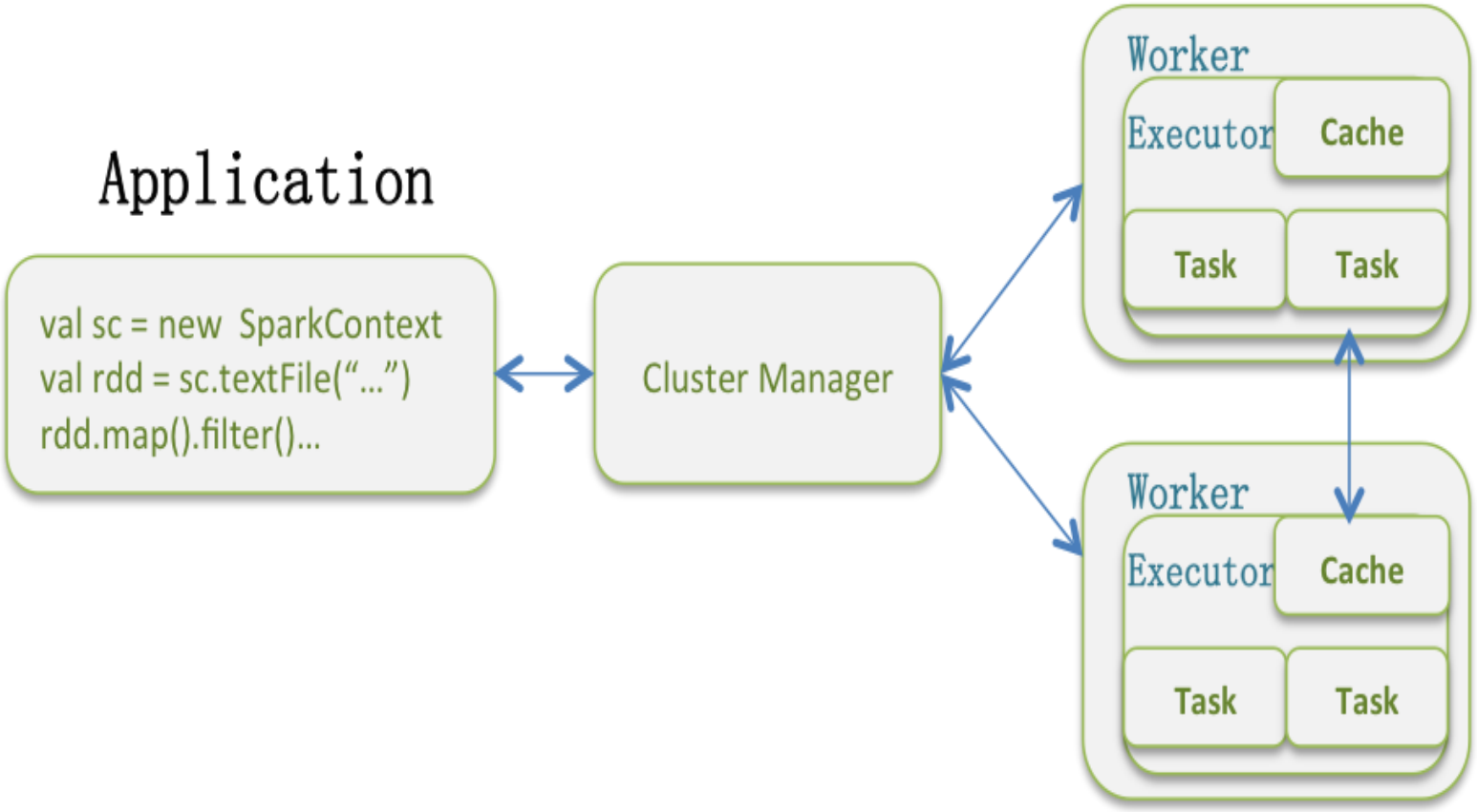


Baidu 百度

BDAS


BlinkDB
SQL w/ bounded errors/response times
Spark Streaming
Stream processing
GraphX
Graph computation
MLbase
User-friendly machine learning
Shark
SQL API
Hive
Storm
MPI
Spark
Fast memory-optimized execution engine (Python/Java/Scala APIs)
Hadoop MR
Tachyon (alpha) In-memory file system
Hadoop Distributed File System (HDFS)
Mesos Cluster resource manager, multi-tenancy
Supported Release
In Development
Related External Project

整体架构


Application
val sc = new SparkContext
val rdd = sc.textFile("...")
rdd.map().filter()...
Cluster Manager
Worker
Executor
Cache
Task
Task
Worker
Executor
Cache
Task
Task

Spark的目标

- 高效和通用的编程模型
  - 相比**MapReduce**，更加适用于两种类型的应用
    - 迭代算法类（机器学习，图计算）
    - 交互式数据挖掘
  - 相似的编程接口
- 良好的用户体验
    - 编程效率：基于**Scala**的核心模块，并提供**Java/python**编程接口
    - 功能强大的**API,**丰富的操作算子
    - 交互式的解释执行接口（调试，学习）

Baidu 百度

相似的编程接口

```
val conf = new SparkConf()
val sc = new SparkContext(conf)
val lines  = sc.textFile(args(1))
val words = lines.flatMap(_.split(" "))
val result = words.map(x => (x, 1)).reduceByKey(_ + _).collect()
```

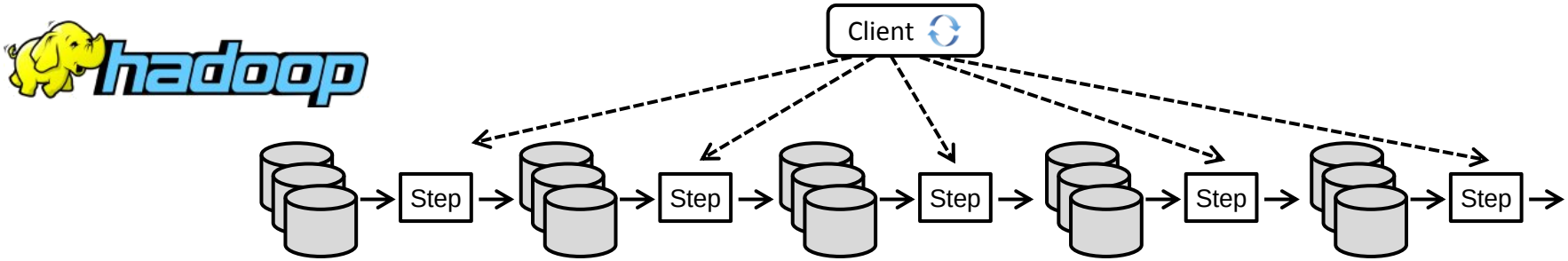
普通版本

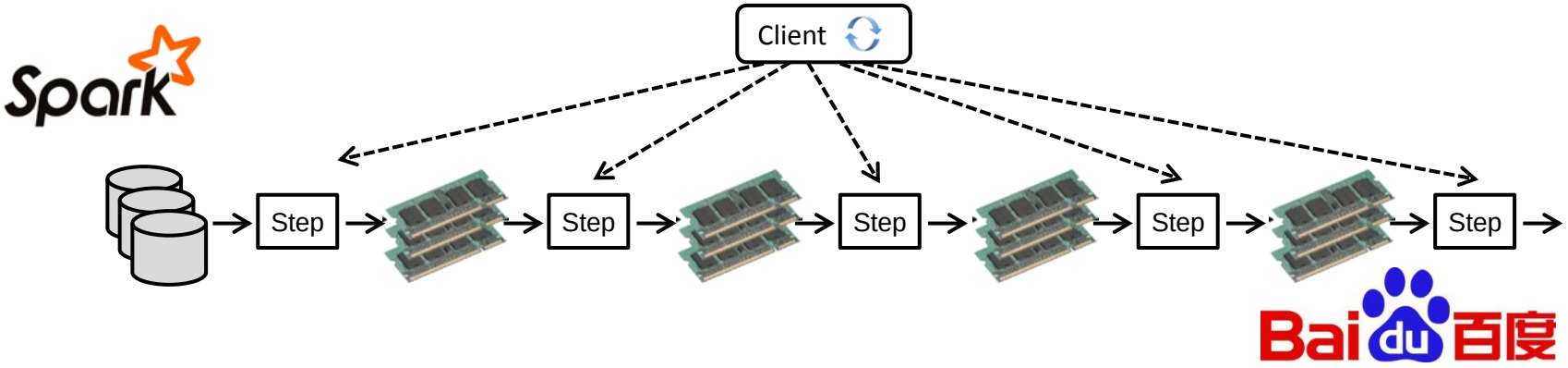
```
val conf = new SparkConf()
val ssc = new StreamingContext(conf, Seconds(1))
val lines = ssc.textFileStream(args(1))
val words = lines.flatMap(_.split(" "))
val result = words.map(x => (x, 1)).reduceByKey(_
+ _).collect()
ssc.start()
```

流式版本

Baidu 百度

仅仅是因为内存？


hadoop
Client
Step
Step
Step
Step
Step


Spark
Client
Step
Step
Step
Step
Step
Baidu 百度

从Word Count开始

- 代码实现
- RDD
- 资源分配
- 任务划分
- 任务调度
- 任务执行
  - **Shuffle**的具体实现

Word Count的代码实现

Python | Scala | Java

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

Python | Scala | Java

```python
file = spark.textFile("hdfs://...")
counts = file.flatMap(lambda line: line.split(" ")) \
             .map(lambda word: (word, 1)) \
             .reduceByKey(lambda a, b: a + b)
counts.saveAsTextFile("hdfs://...")
```

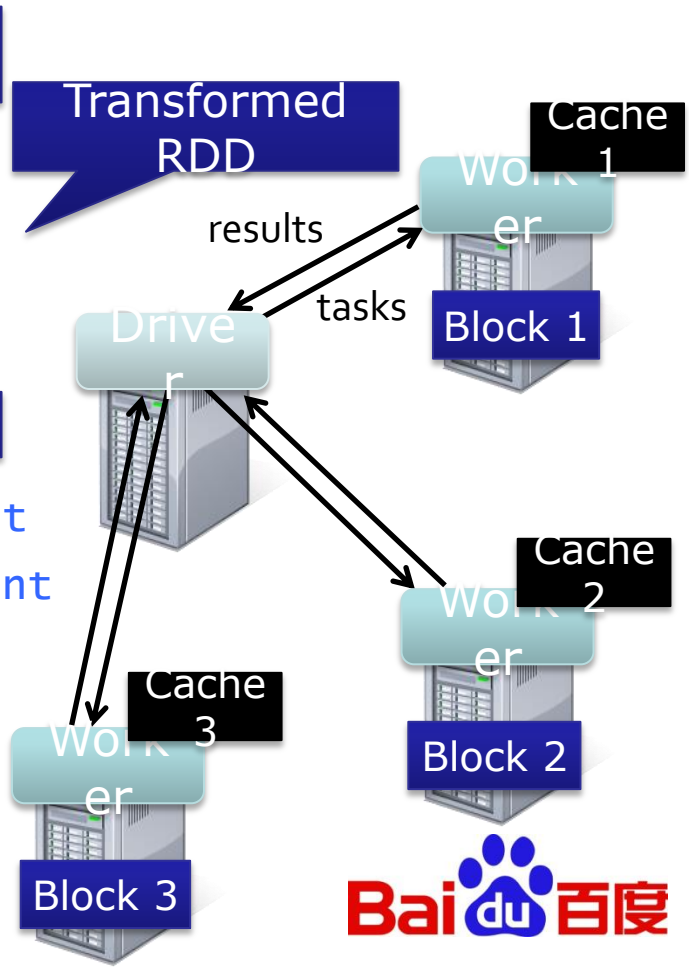
RDD

- Resilient distributed dataset，弹性分布式数据集
- 不可变的，按分区组织的数据对象
- 支持多种转换 + 动作
- 可以通过多种数据源创建RDD
- 缓存 + 检查点
- 容错，数据本地性，可扩展性

RDD的缓存

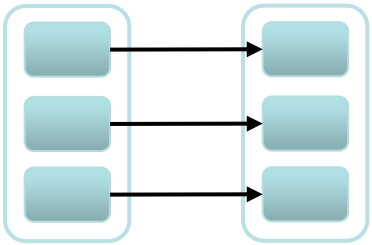
```
lines = spark.textFile(“hdfs://...”)
errors = lines.filter(_.startsWith(“ERROR”))
messages = errors.map(_.split(‘\t’)(2))
cachedMsgs = messages.cache()

cachedMsgs.filter(_.contains(“wordseg”)).count
cachedMsgs.filter(_.contains(“wordrank”)).count
  . . .
```

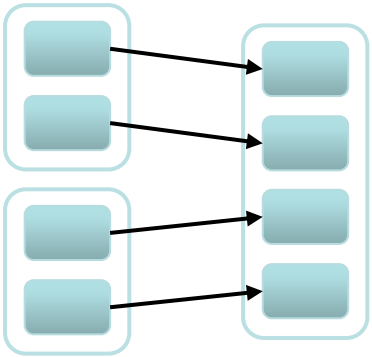


RDD的不同依赖

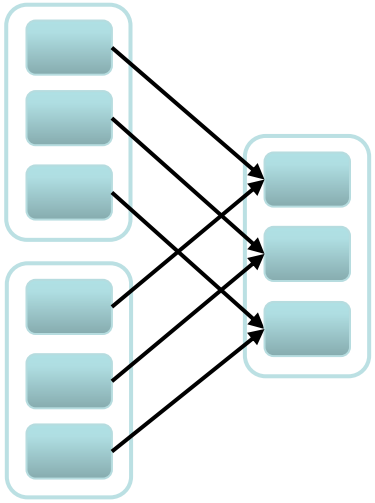
窄依赖:



map, filter

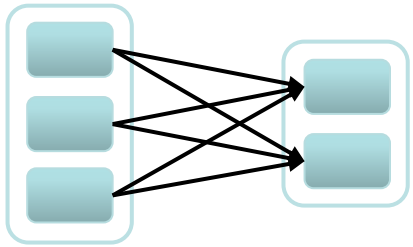


union

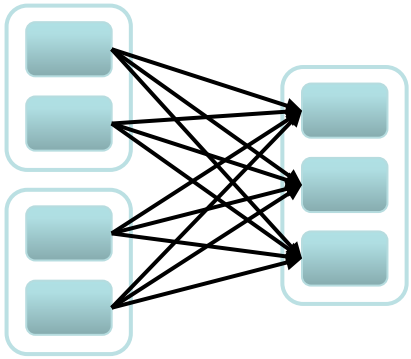


join with inputs co-partitioned

宽依赖:

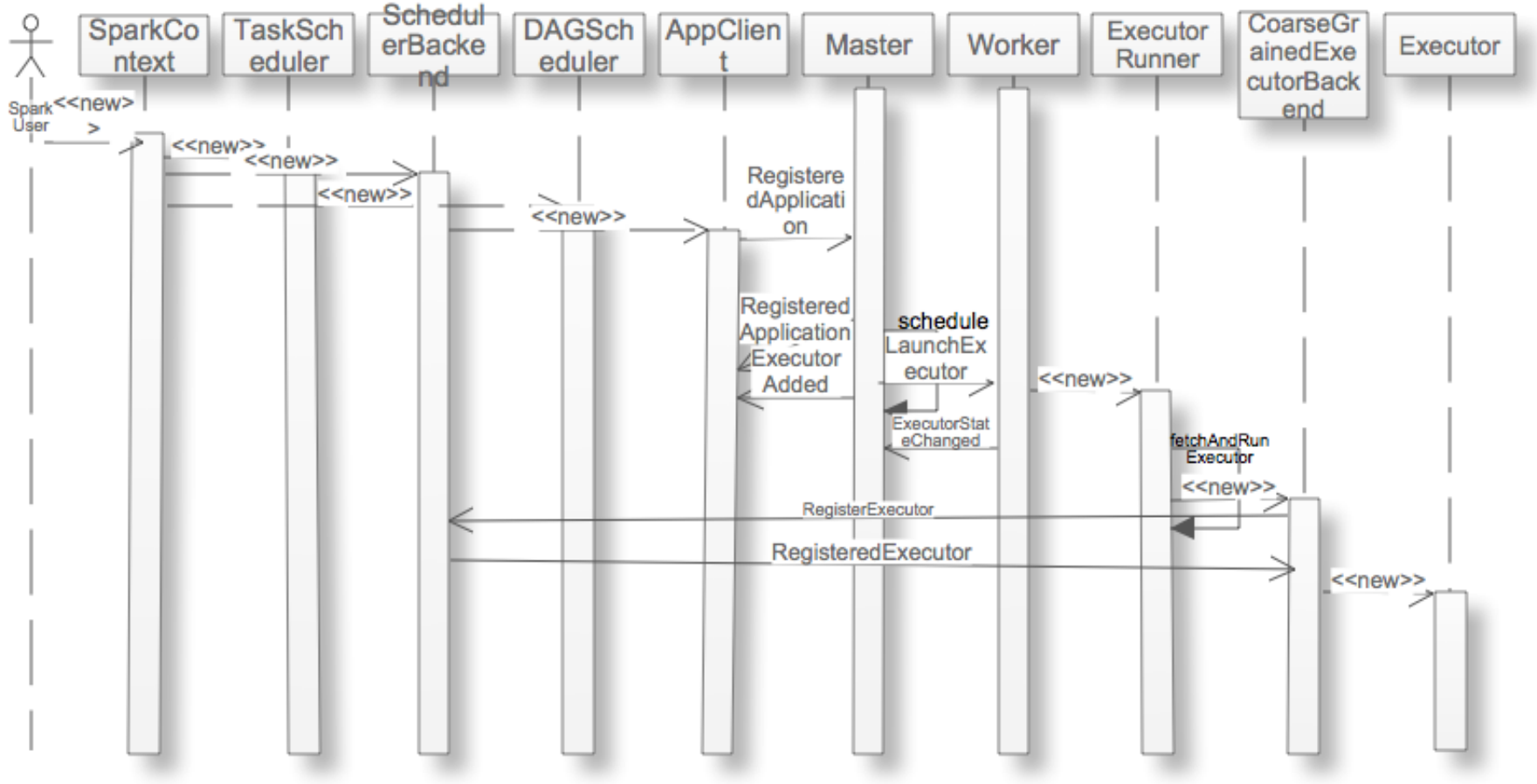


groupByKey

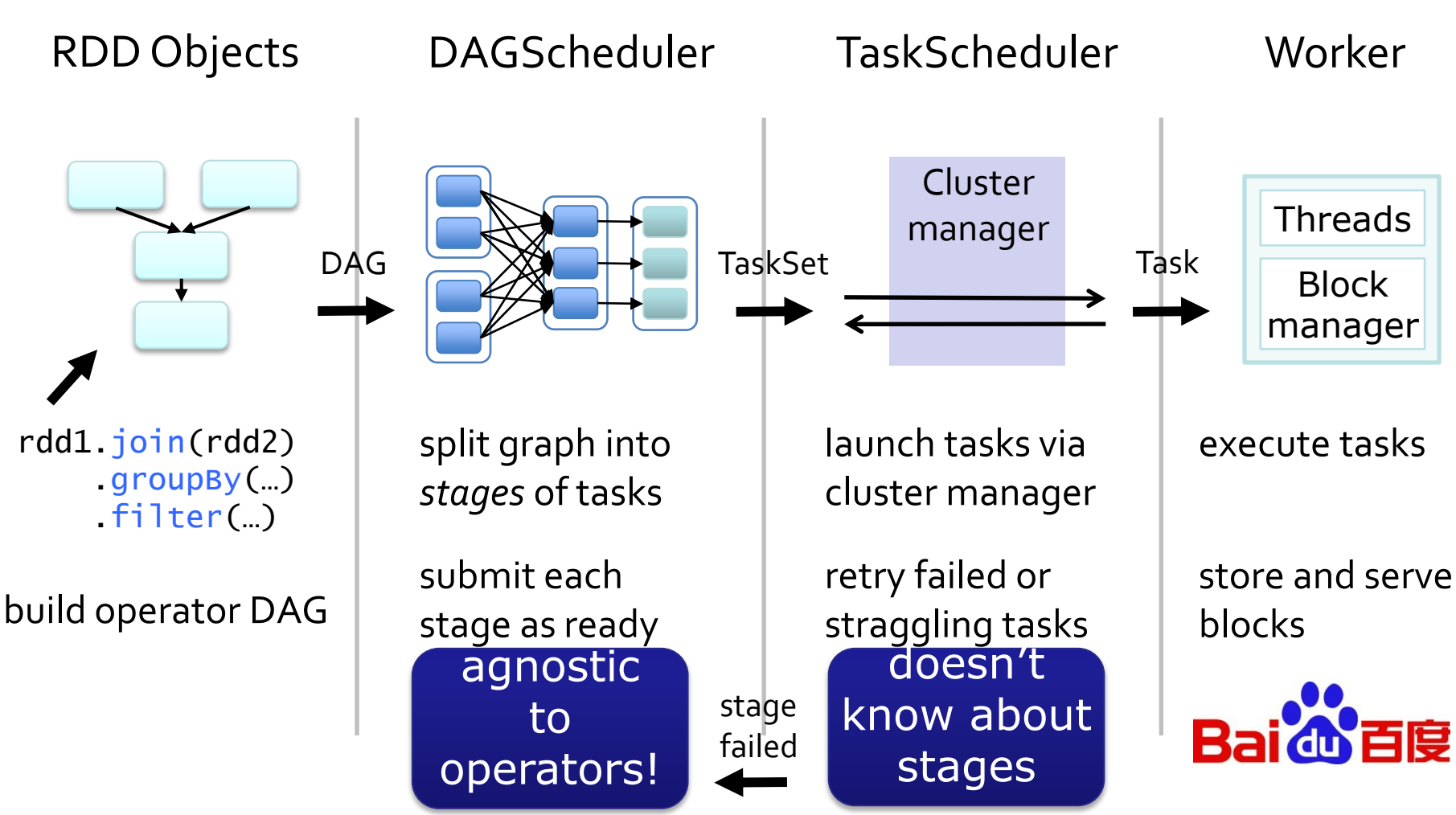


join with inputs not co-partitioned

资源的分配

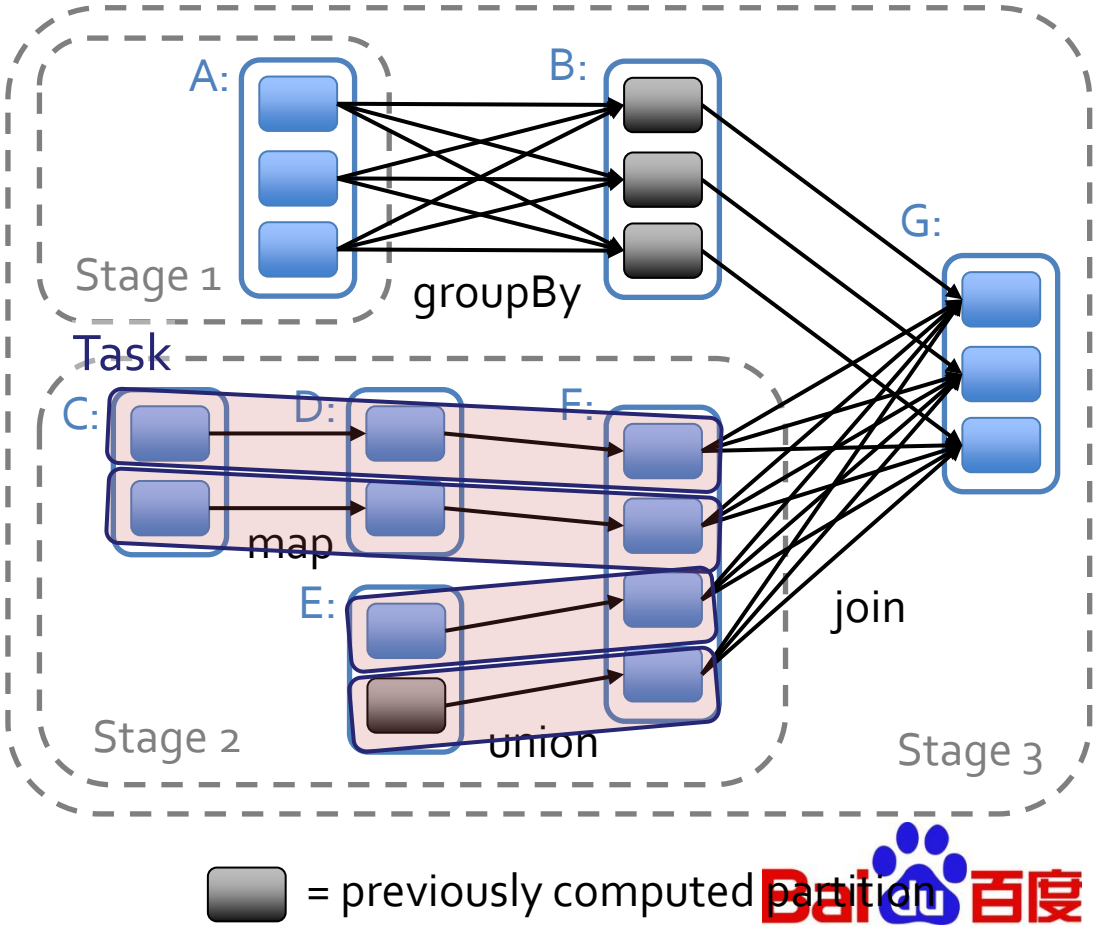

SparkContext
TaskScheduler
SchedulerBackend
DAGScheduler
AppClient
Master
Worker
ExecutorRunner
CoarseGrainedExecutorBackend
Executor
Spark User
<<new>>
<<new>>
<<new>>
<<new>>
<<new>>
RegisteredApplication
RegisteredApplication
ExecutorAdded
schedule
LaunchExecutor
ExecutorStateChanged
<<new>>
fetchAndRunExecutor
<<new>>
RegisterExecutor
RegisteredExecutor
<<new>>

任务调度


RDD Objects
DAGScheduler
TaskScheduler
Worker
DAG
TaskSet
Task
Cluster manager
Threads
Block manager
rdd1.join(rdd2)
.groupBy(…)
.filter(…)
build operator DAG
split graph into *stages* of tasks
submit each stage as ready
launch tasks via cluster manager
retry failed or straggling tasks
execute tasks
store and serve blocks
agnostic to operators!
stage failed
doesn't know about stages
Baidu 百度

DAG的划分

- 由DAGScheduler完成DAG不同Stage的划分
- 最大化pipeline
- 尽可能的减少Shuffle



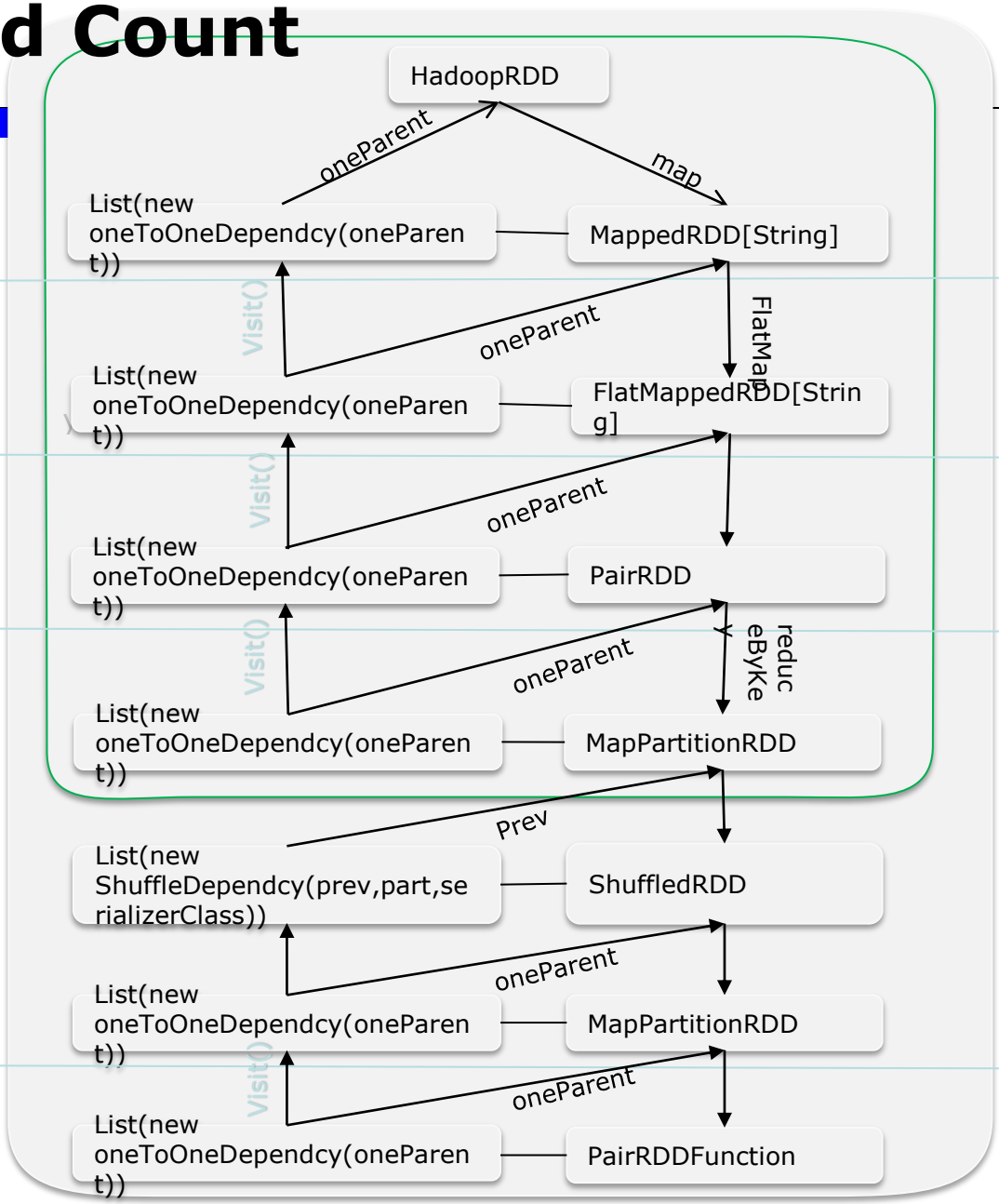
Task的生成

- 根据Final Stage的Partition生成ResultTask，每个Partition对应于一个Task
- 其余的Stage的每个Partition生成一个ShuffleMapTask

Task的执行

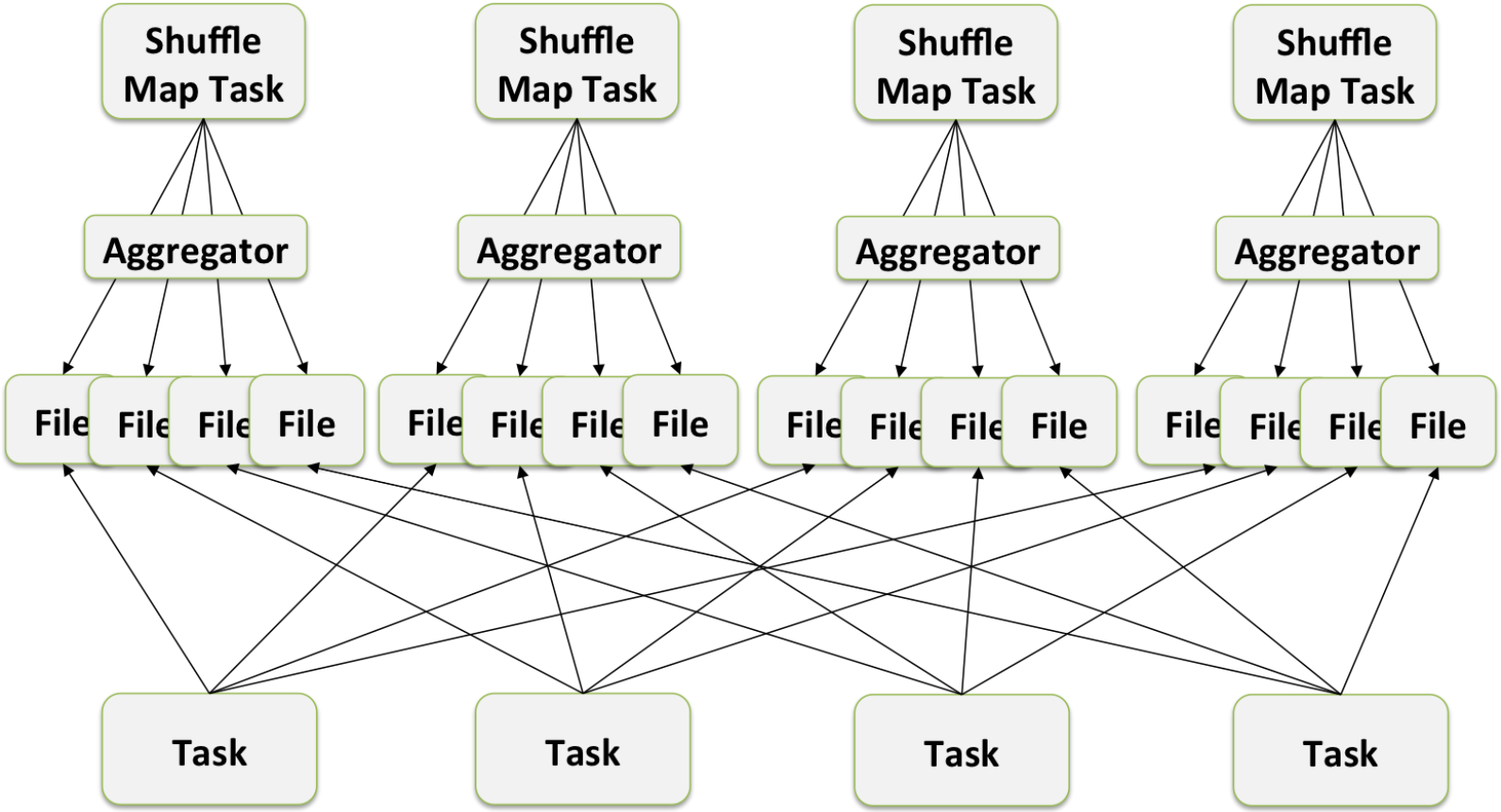
- TaskScheduler会将任务发送到Executor
- Executor开始每个Task的执行

Word Count

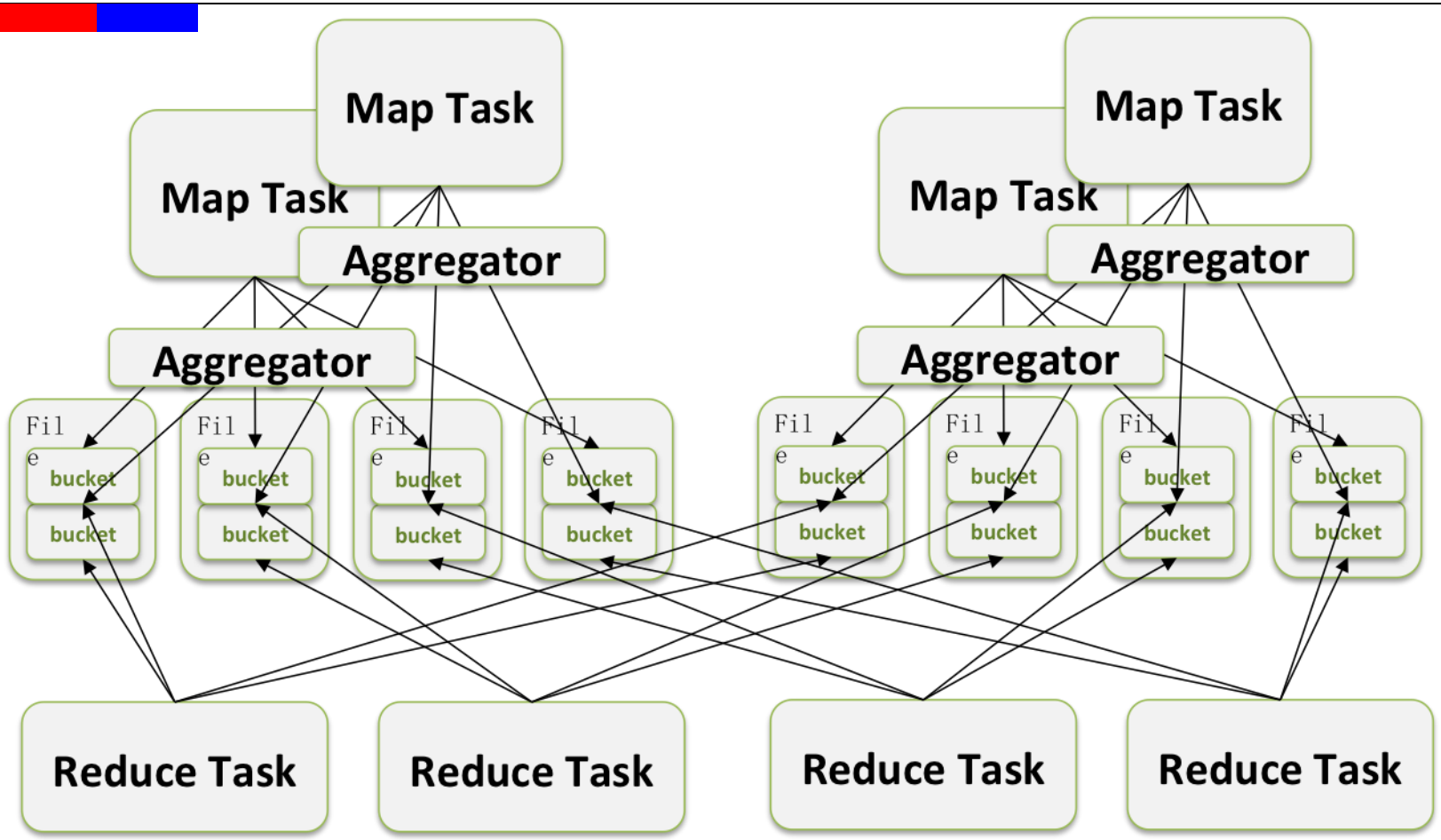



```
lines=ssc.textFile("filepath")
words=lines.flatMap(x=> x.split(" "))
wordCounts=words.map(x=> (x,1))
result=wordCounts.reduceByKey(_ + _)
result.saveAsTextFile("output_filepath")
```

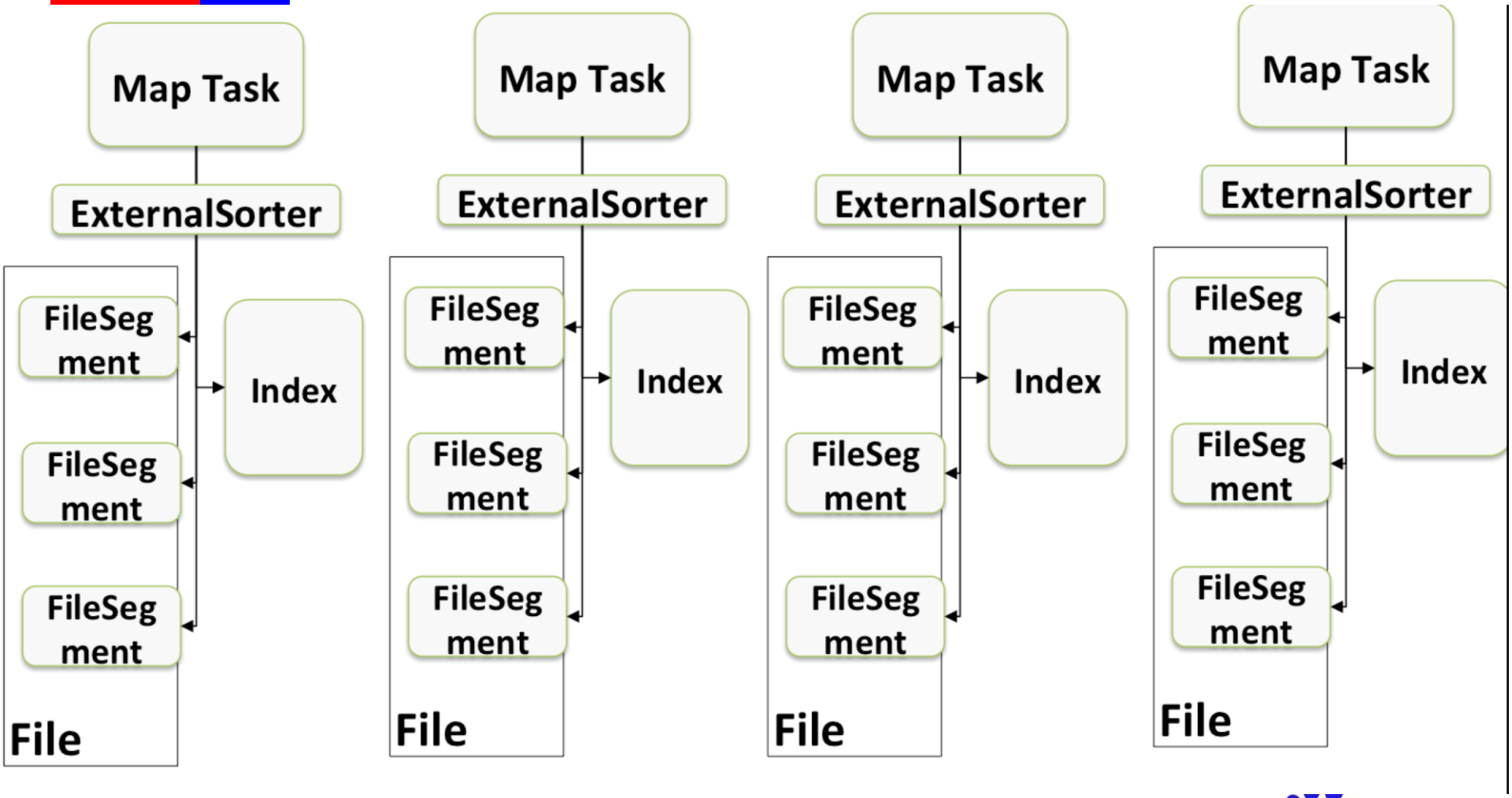
Base Shuffle Write



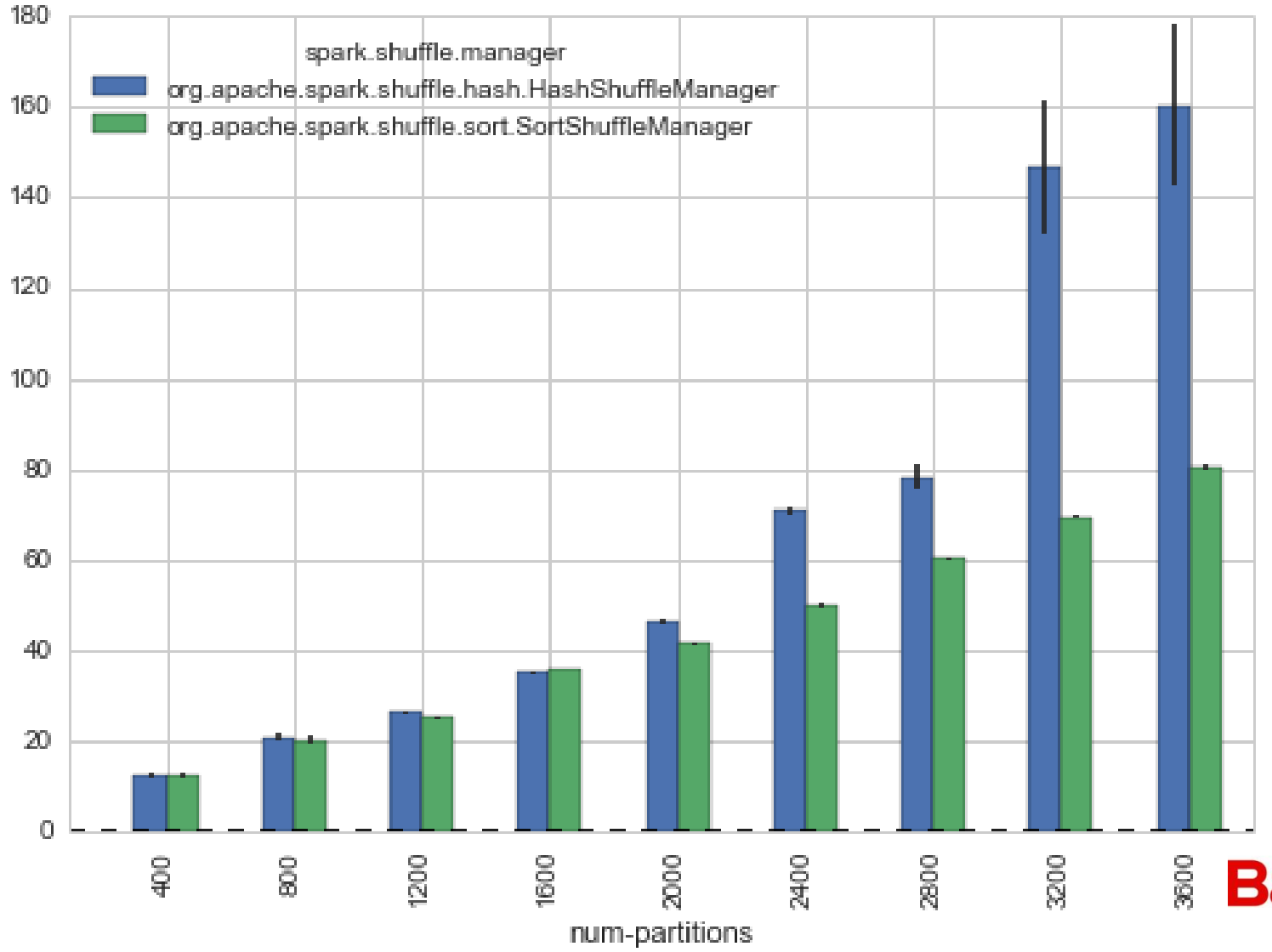
Consolidate Shuffle Write



Sort Based Shuffle - Shuffle Writer

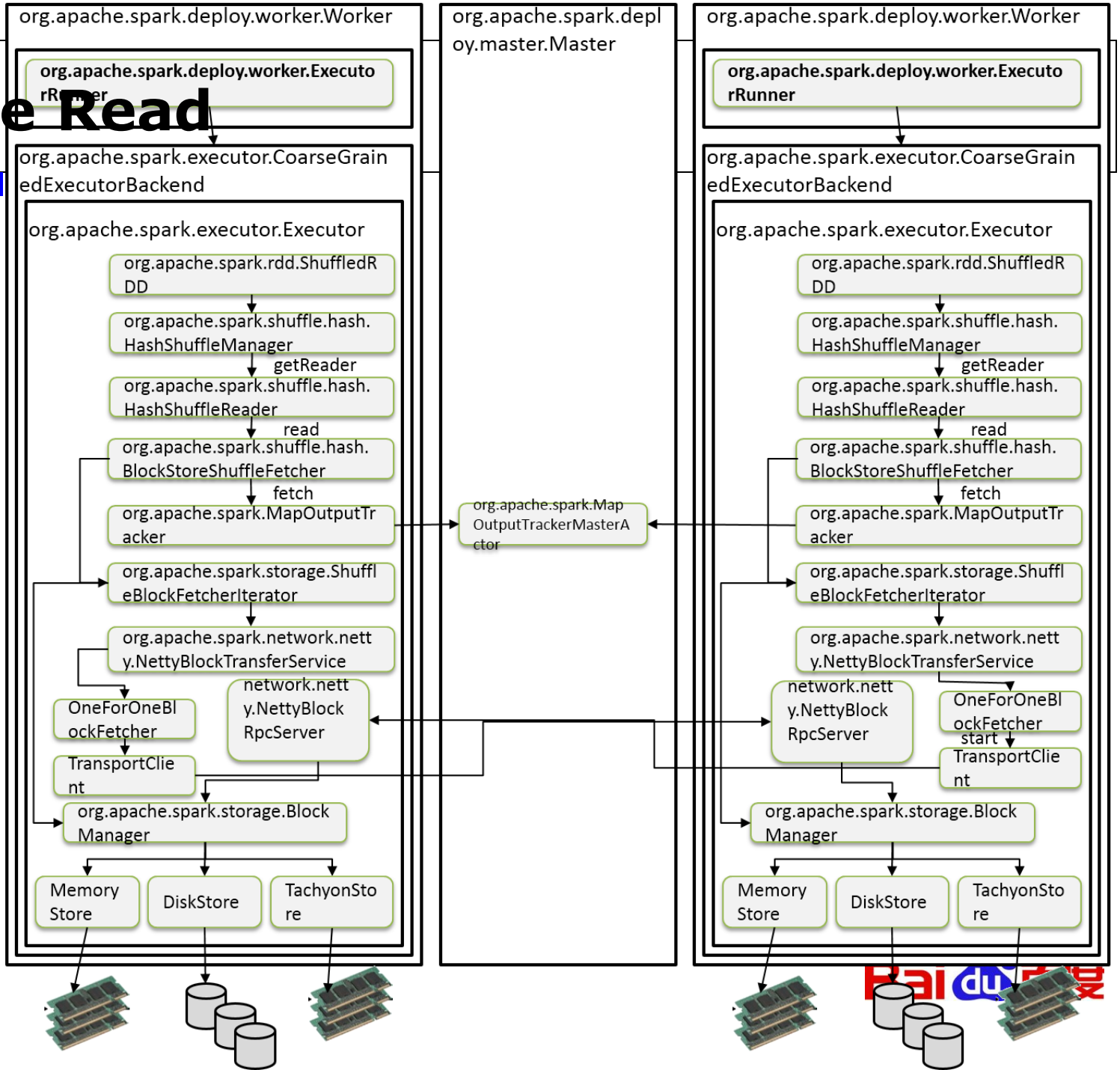


Hash Based VS Sort Based



Baidu 百度

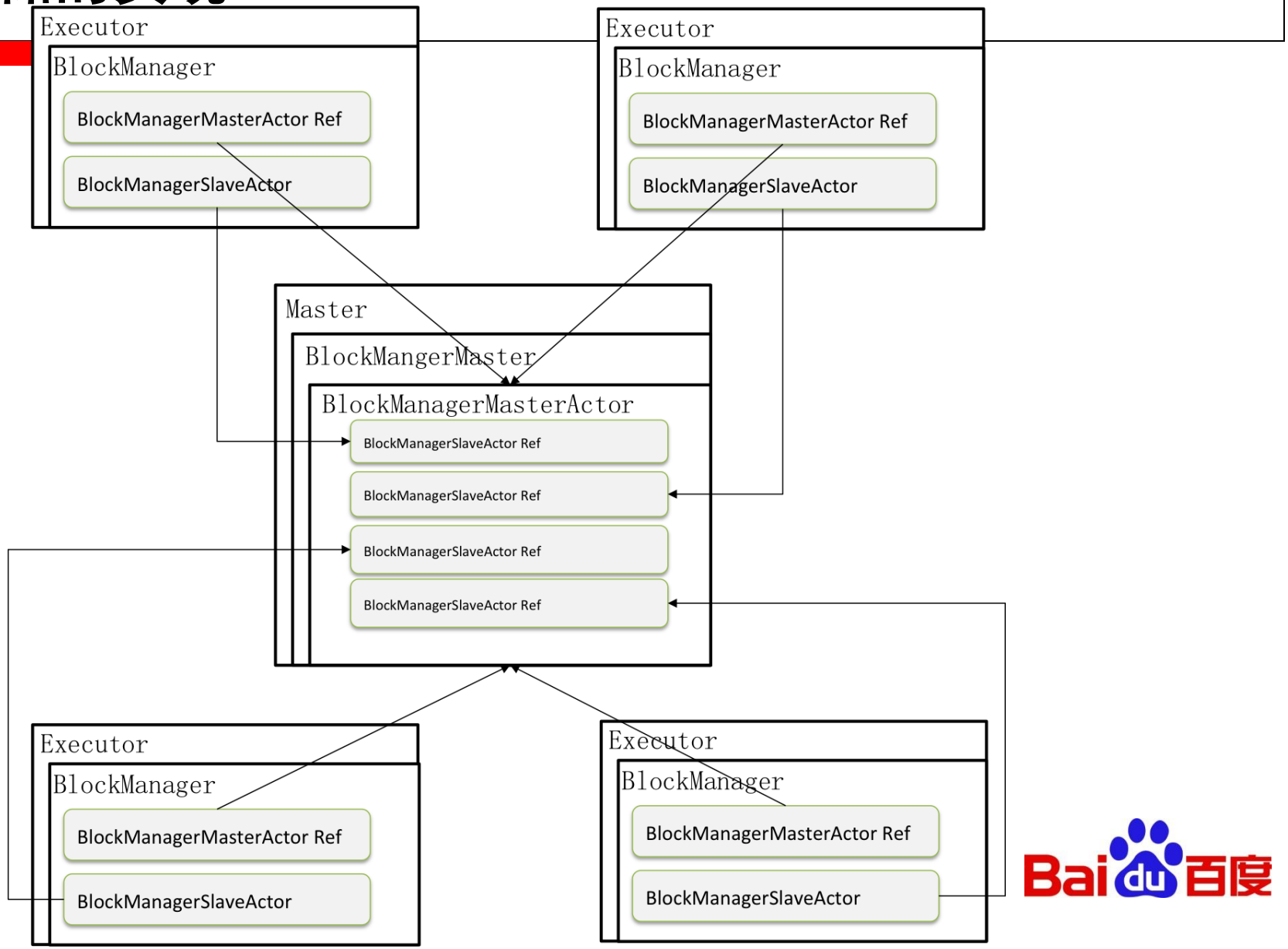
Shuffle Read



Shuffle Pluggable 框架

Baidu 百度

存储的实现


Executor
BlockManager
BlockManagerMasterActor Ref
BlockManagerSlaveActor
Executor
BlockManager
BlockManagerMasterActor Ref
BlockManagerSlaveActor
Master
BlockMangerMaster
BlockManagerMasterActor
BlockManagerSlaveActor Ref
BlockManagerSlaveActor Ref
BlockManagerSlaveActor Ref
BlockManagerSlaveActor Ref
Executor
BlockManager
BlockManagerMasterActor Ref
BlockManagerSlaveActor
Executor
BlockManager
BlockManagerMasterActor Ref
BlockManagerSlaveActor

Important But not Covered

- Scala
- Spark SQL
- Tachyon & RAMCloud
- Spark Best Practice（performance tuning）
- Spark in Industry
- Spark Ecosystem

Baidu 百度

Q&A