Life is short, you need Spark!

从零开始

不需要任何基础，带领您无痛入门 Spark

云计算分布式大数据 Spark 实战高手之路

王家林著

Spark 亚太研究院系列丛书

伴随着大数据相关技术和产业的逐步成熟，继 Hadoop 之后，Spark 技术以其无可比拟的优势，发展迅速，将成为替代 Hadoop 的下一代云计算、大数据核心技术。

本书特点

- 云计算分布式大数据 Spark 实战高手之路三部曲之第一部
- 网络发布版为图文并茂方式，边学习，边演练
- 不需要任何前置知识，从零开始，循序渐进

本书作者

Spark 亚太研究院院长和首席专家，中国目前唯一的移动互联网和云计算大数据集大成者。在 Spark、Hadoop、Android 等方面有丰富的源码、实务和性能优化经验。彻底研究了 Spark 从 0.5.0 到 0.9.1 共 13 个版本的 Spark 源码，并已完成 2014 年 5 月 31 日发布的 Spark1.0 源码研究。

Hadoop 源码级专家，曾负责某知名公司的类 Hadoop 框架开发工作，专注于 Hadoop 一站式解决方案的提供，同时也是云计算分布式大数据处理的最早实践者之一。

Android 架构师、高级工程师、咨询顾问、培训专家。

通晓 Spark、Hadoop、Android、HTML5，迷恋英语播音和健美。

“真相会使你获得自由。”

—— 耶稣《圣经》约翰 8:32KJV

“所有人类的不幸都来源于不能直面事实。”

—— 释迦摩尼

“道法自然”

—— 老子《道德经》第 25 章

《云计算分布式大数据 Spark 实战高手之路》

系列丛书三部曲

《云计算分布式大数据 **Spark** 实战高手之路---从零开始》：

不需要任何基础，带领您无痛入门 Spark 并能够轻松处理 Spark 工程师的日常编程工作，内容包括 Spark 集群的构建、Spark 架构设计、RDD、Shark/SparkSQL、机器学习、图计算、实时流处理、Spark on Yarn、JobServer、Spark 测试、Spark 优化等。

《云计算分布式大数据 **Spark** 实战高手之路---高手崛起》：

大话 Spark 源码，全世界最有情趣的源码解析，过程中伴随诸多实验，解析 Spark 1.0 的任何一句源码！更重要的是，思考源码背后的问题场景和解决问题的设计哲学和实现招式。

《云计算分布式大数据 **Spark** 实战高手之路---高手之巅》：

通过当今主流的 Spark 商业使用方法和最成功的 Hadoop 大型案例让您直达高手之巅，从此一览众山小。

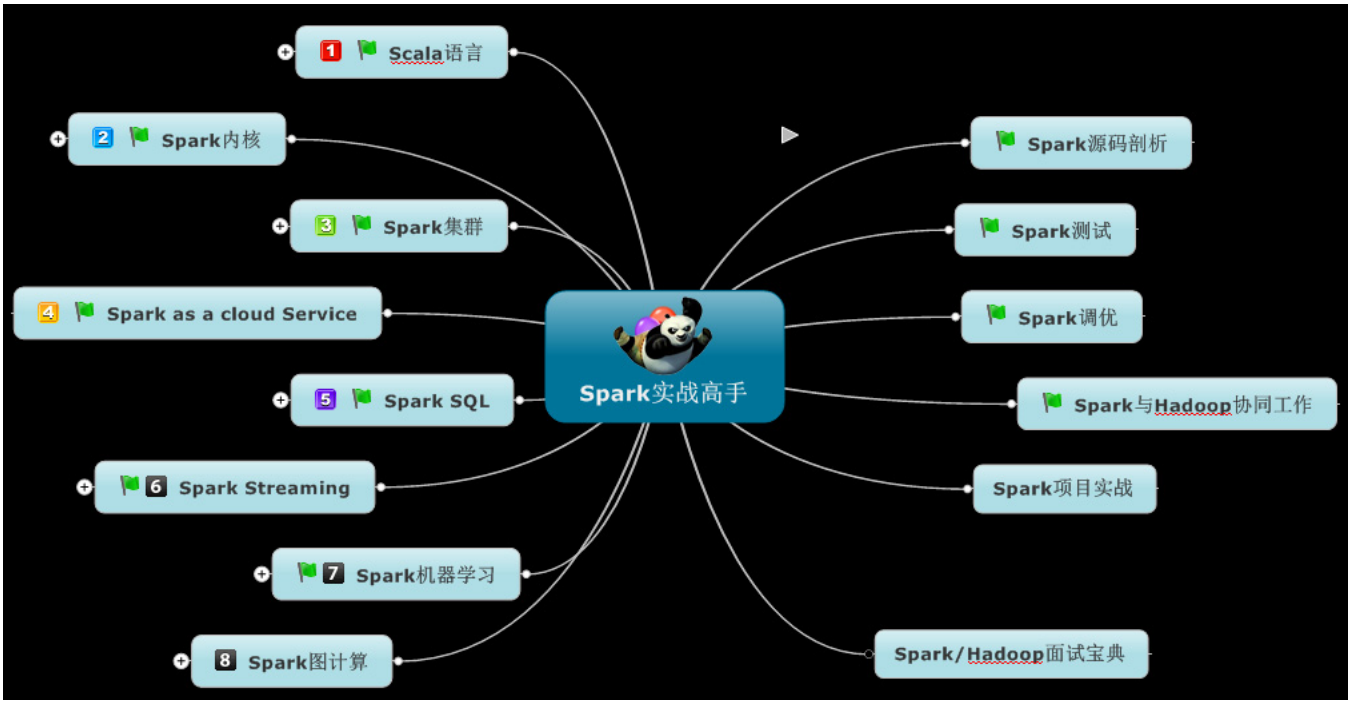
www.sparkinchina.com
TEL: 4006-998-758

《前言》

Spark采用一个统一的技术堆栈解决了云计算大数据的如流处理、图技术、机器学习、NoSQL查询等方面的所有核心问题，具有完善的生态系统，这直接奠定了其一统云计算大数据领域的霸主地位；

要想成为Spark高手，需要经历六大阶段

Spark 实战高手之核心技能点



第一阶段：熟练的掌握Scala语言

1. Spark 框架是采用 Scala 语言编写的，精致而优雅。要想成为 Spark 高手，你就必须阅读 Spark 的源代码，就必须掌握 Scala,；
2. 虽然说现在的 Spark 可以采用多语言 Java、Python 等进行应用程序开发，但是最快速的和支持最好的开发 API 依然并将永远是 Scala 方式的 API，所以你必须掌握 Scala 来编写复杂的和高性能的 Spark 分布式程序；
3. 尤其要熟练掌握 Scala 的 trait、apply、函数式编程、泛型、逆变与协变等；

推荐课程：" 精通Spark的开发语言：Scala最佳实践"

第二阶段：精通Spark平台本身提供给开发者API

1. 掌握Spark中面向RDD的开发模式 掌握各种transformation和action函数的使用；
2. 掌握 Spark 中的宽依赖和窄依赖以及 lineage 机制；
3. 掌握 RDD 的计算流程，例如 Stage 的划分、Spark 应用程序提交给集群的基本过程和 Worker 节点基础的工作原理等

推荐课程： "18 小时内掌握Spark：把云计算大数据速度提高 100 倍以上!"

第三阶段：深入Spark内核

此阶段主要是通过 Spark 框架的源码研读来深入 Spark 内核部分：

1. 通过源码掌握 Spark 的任务提交过程；
2. 通过源码掌握 Spark 集群的任务调度；
3. 尤其要精通 DAGScheduler、TaskScheduler 和 Worker 节点内部的工作的每一步的细节；

推荐课程： “Spark 1.0.0 企业级开发动手：实战世界上第一个Spark 1.0.0 课程，涵盖Spark 1.0.0 所有的企业级开发技术”

第四阶级:掌握基于Spark上的核心框架的使用

Spark 作为云计算大数据时代的集大成者，在实时流处理、图技术、机器学习、NoSQL 查询等方面具有显著的优势，我们使用 Spark 的时候大部分时间都是在使用其上的框架例如 Shark、Spark Streaming 等：

1. Spark Streaming 是非常出色的实时流处理框架，要掌握其 DStream、transformation 和 checkpoint 等；
2. Spark 的离线统计分析功能，Spark 1.0.0 版本在 Shark 的基础上推出了 Spark SQL，离线统计分析的功能的效率有显著的提升，需要重点掌握；
3. 对于 Spark 的机器学习和 GraphX 等要掌握其原理和用法；

推荐课程： “Spark企业级开发最佳实践”

第五阶级:做商业级别的Spark项目

通过一个完整的具有代表性的 Spark 项目来贯穿 Spark 的方方面面，包括项目的架构设计、用到的技术的剖析、开发实现、运维等，完整掌握其中的每一个阶段和细节，这样就可以让您以后可以从容面对绝大多数 Spark 项目。

推荐课程： “Spark架构案例鉴赏：Conviva、Yahoo！、优酷土豆、网易、腾讯、淘宝等公司的实际Spark案例”

第六阶级：提供Spark解决方案

1. 彻底掌握 Spark 框架源码的每一个细节；
2. 根据不同的业务场景的需要提供 Spark 在不同场景的下的解决方案；
3. 根据实际需要，在 Spark 框架基础上进行二次开发，打造自己的 Spark 框架；

推荐课程： “精通Spark：Spark内核剖析、源码解读、性能优化和商业案例实战”

《第三章：Spark 架构设计与编程模型》

Spark 是大数据时代通用而高效的计算平台 基于 RDD 成功实现了“One stack to rule them all” 理念。

目前 SPARK 已经构建了自己的整个大数据处理生态系统，如流处理、图技术、机器学习、NoSQL 查询等方面都有自己的技术 ,并且是 Apache 顶级 Project ,可以预计的是 2014 年下半年到 2015 年在社区和商业应用上会有爆发式的增长。

国外一些大型互联网公司已经部署了 Spark。甚至连 Hadoop 的早期主要贡献者 Yahoo 现在也在多个项目中部署使用 Spark ;国内的淘宝、优酷土豆、网易、搜狐、Baidu、腾讯等已经使用 Spark 技术用于自己的商业生产系统中，国内外的应用开始越来越广泛。Spark 正在逐渐走向成熟，并在这个领域扮演更加重要的角色。

刚刚结束的 2014 Spark Summit 上的信息，Spark 已经获得世界 20 家顶级公司的支持，这些公司中包括 Intel、IBM 等，同时更重要的是包括了最大的四个 Hadoop 发行商都提供了对非常强有力的支持 Spark 的支持。

本章首先介绍为什么 Spark 是大数据必然的现在和未来，接着讲解 Spark 架构和生态系统，然后细致解析 Spark 的编程模型，最后通过众多的案例动手实践 Spark 编程，从零开始，循序渐进，希望助力诸位 Spark 爱好者能够顺利入门 Spark。

Spark 架构设计与编程模型实战共分四个部分：

- 第一部分：为什么 Spark 是大数据必然的现在和未来？！
- 第二部分：Spark 架构设计
- 第三部分：Spark 编程模型
- 第四部分：动手实战 Spark 编程

本将是 **Spark** 架构设计与编程模型的第二部分：**Spark** 架构设计 ,具体内容如下所示：

- 1，到底什么是 Spark？
- 2，Spark 的速度为何如此之快？
- 3，Spark 的 RDD；
- 4，Spark 高容错机制 Lineage

不需任何前置知识，从零开始，循序渐进，成为 Spark 高手！

目录

一、到底什么是Spark？ 8

二， Spark的速度为何如此之快？ 10

三， Spark的RDD 14

四， Spark的高容错机制lineage 17

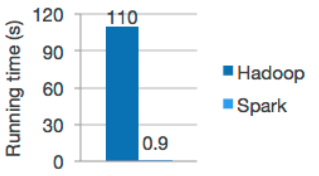
一、到底什么是 Spark？

Apache Spark™ is a fast and general engine for large-scale data processing.

Speed

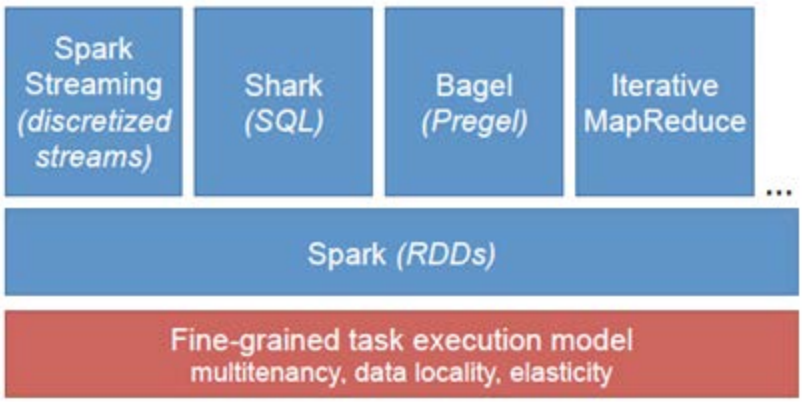
Run programs up to 100x faster than Hadoop MapReduce in memory, or 10x faster on disk.

Spark has an advanced DAG execution engine that supports cyclic data flow and in-memory computing.

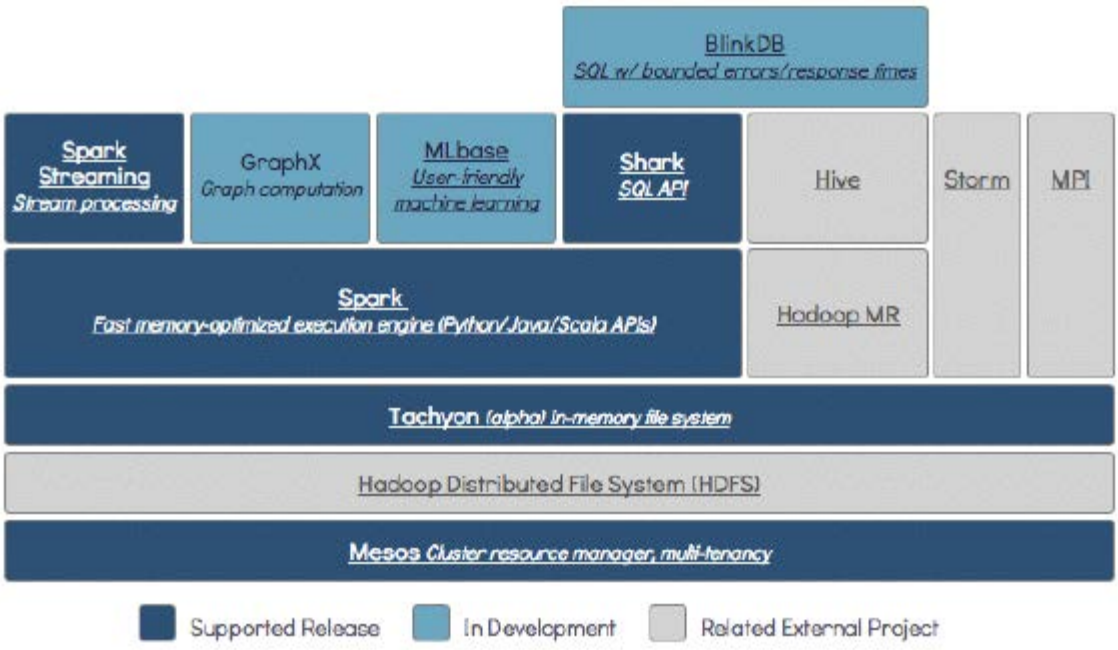


Logistic regression in Hadoop and Spark

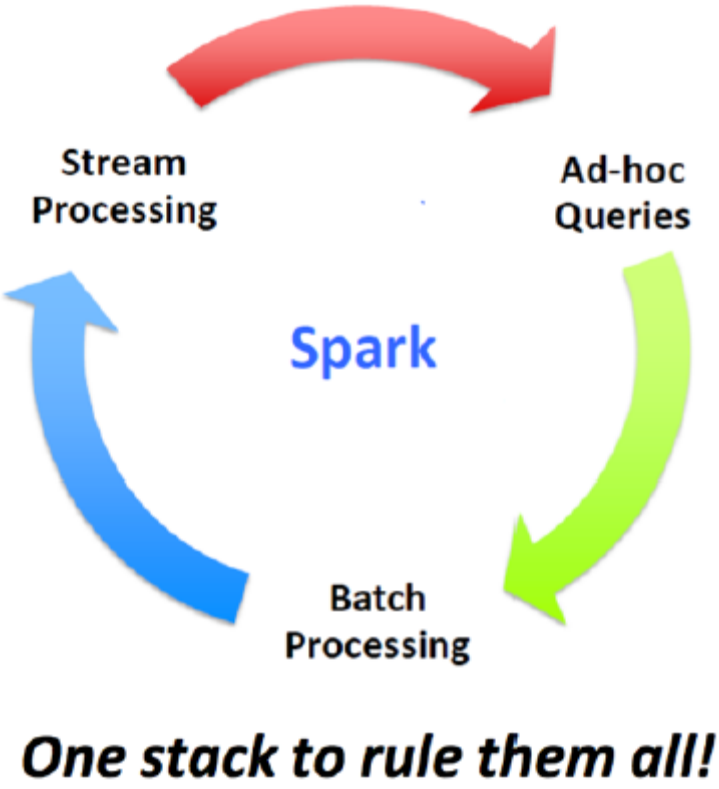
Spark 是一个通用的大数据计算平台，基于“One Stack to rule them all”的理念成功成为了一体化多元化的大数据处理平台，轻松应对大数据处理中的实时流计算、SQL 交互式查询、机器学习和图计算等：



Spark 源于 BDAS:

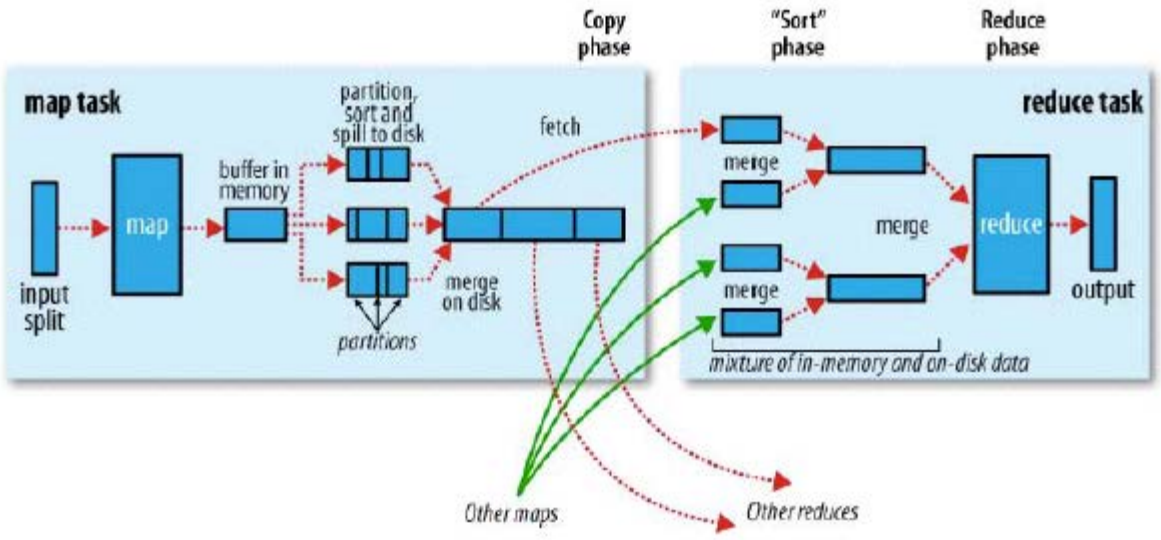


基于该技术堆栈，Spark 目前已经成为大数据通用计算平台：

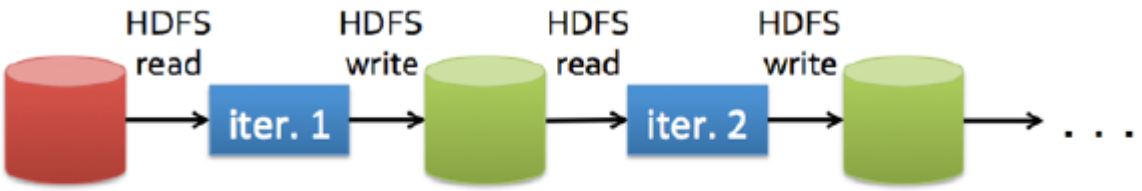


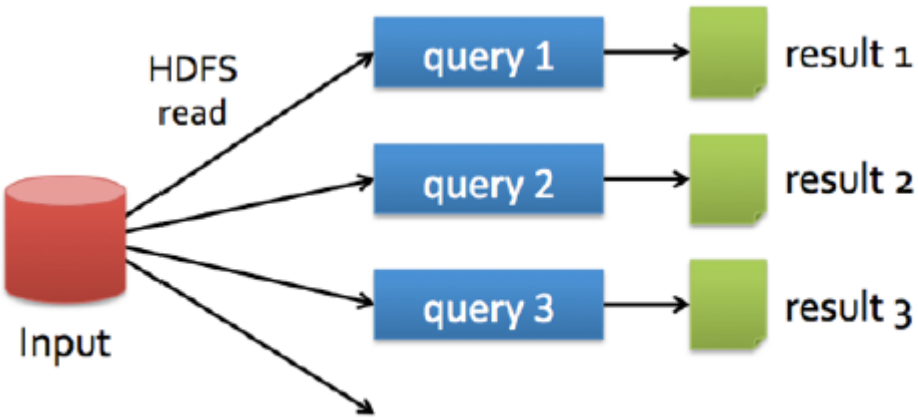
二， Spark 的速度为何如此之快？

首先我们看一下 Hadoop 经典的处理过程：

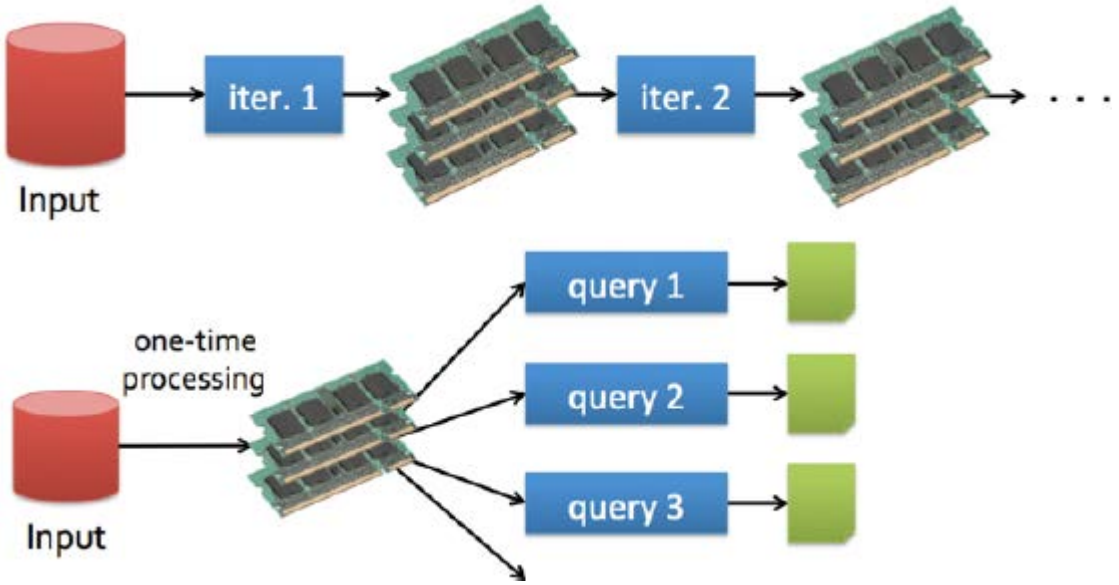


MapReduce 在每次执行的时候都要从磁盘读数据，计算完毕后都要把数据存放到磁盘上：

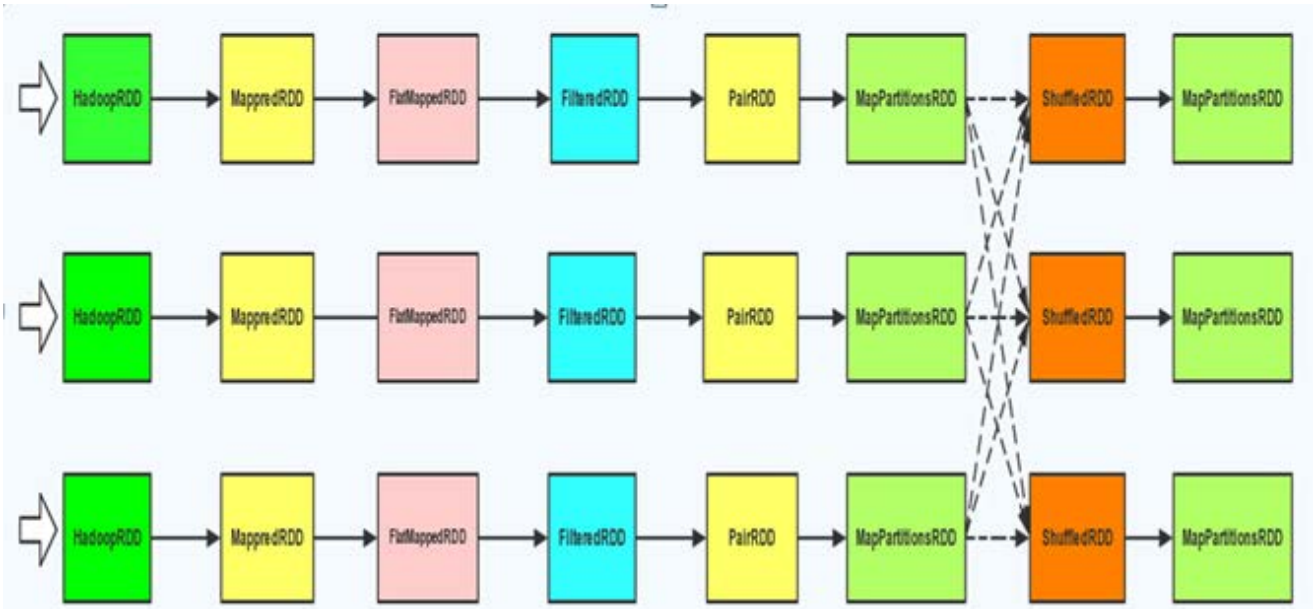




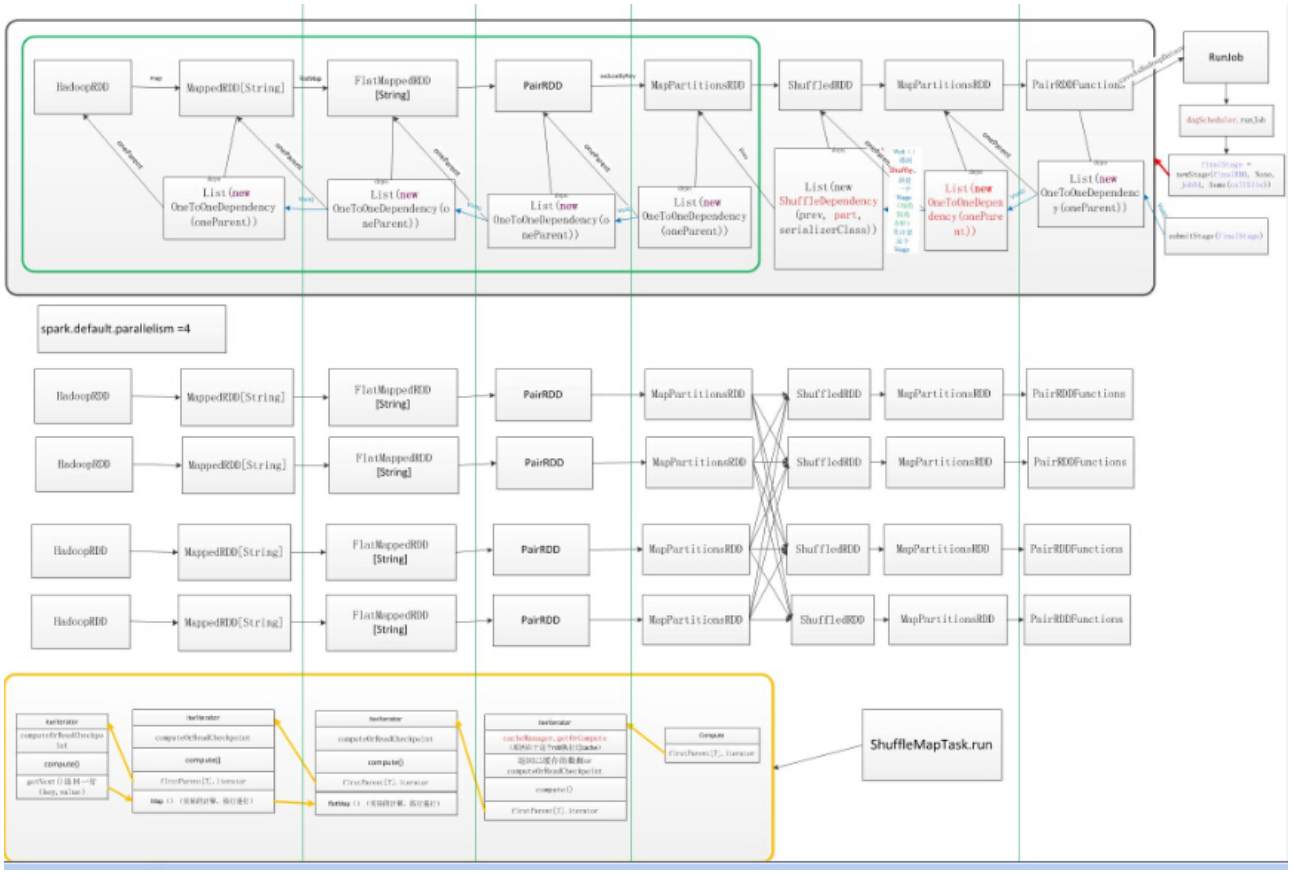
而 Spark 是基于内存的：



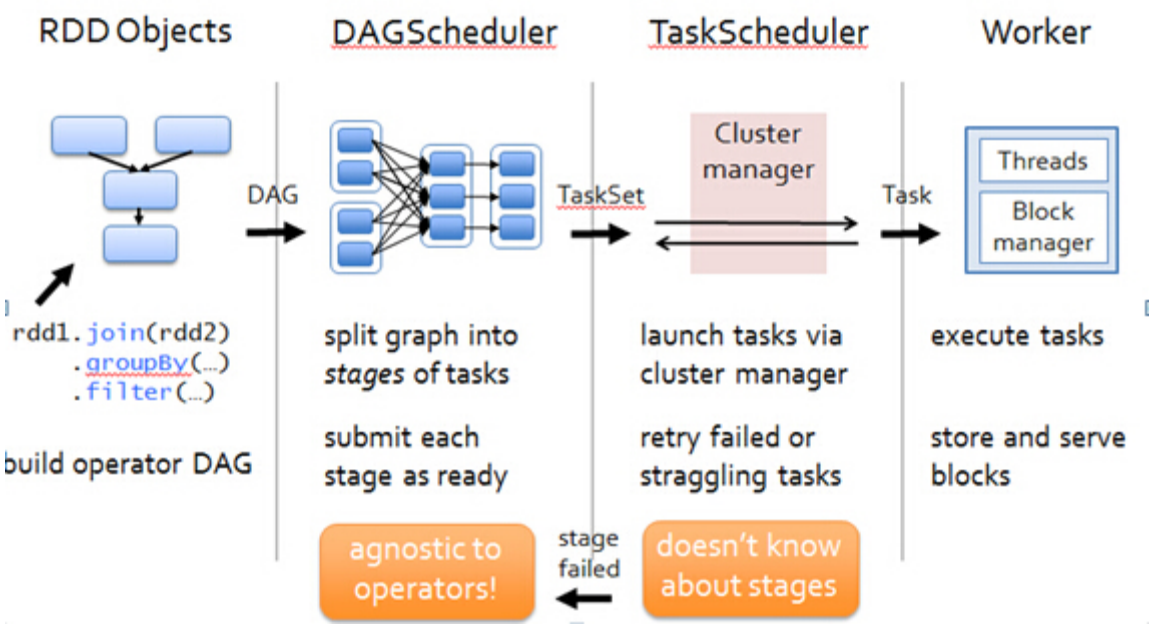
另外一方面，DAG 也是 Spark 快的极为重要的原因，下面是一张 DAG 图的示例：



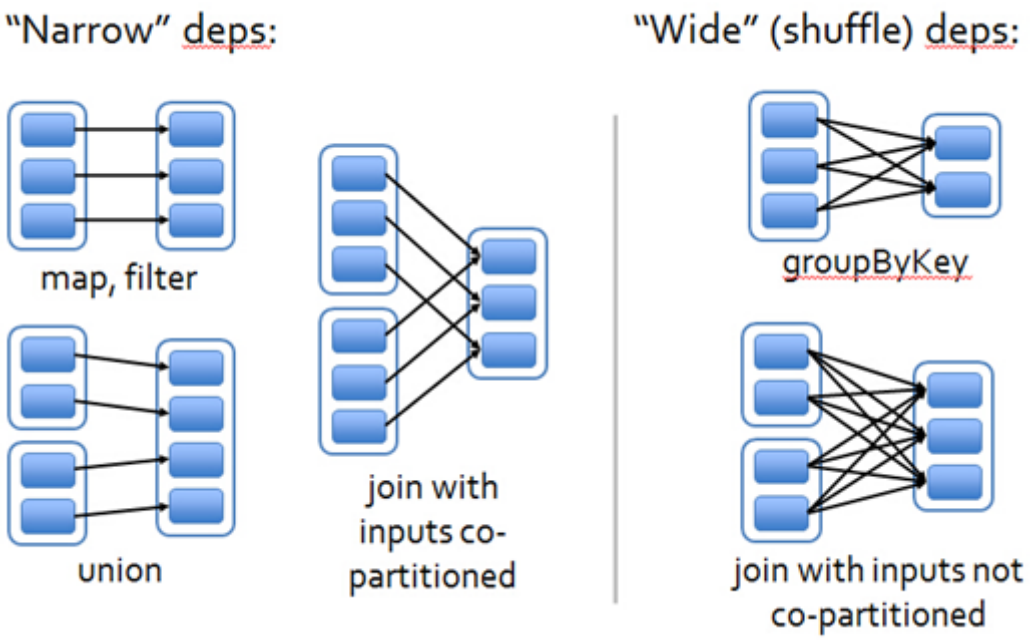
大家也可以看一下网络上一张描述 DAG 更多细节的图片：



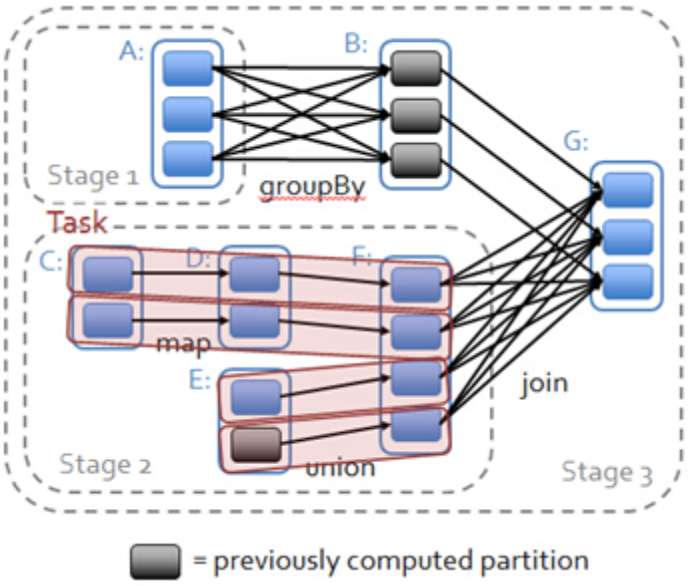
基于 DAG，Spark 具备了非常精致的作业调度系统：



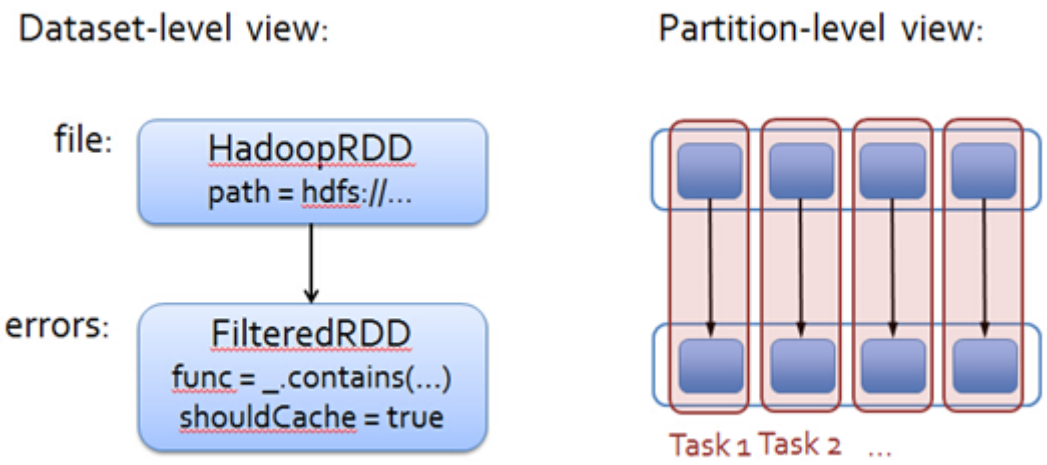
DAG 中的依赖有宽依赖和窄依赖之分：



在 DAG 图中可以根据依赖对 pipeline 等优化操作：



基于 RDD 和 DAG，并行计算整个 Job：



Spark 之所以快 ,还有一个原因就是其容错机制 ,这个我们会在本讲的后面和大家分享。

三， Spark 的 RDD

在 Spark 中一切都是以 RDD 为基础和核心的：

A Resilient Distributed Dataset (RDD), the basic abstraction in Spark. Represents an immutable, partitioned collection of elements that can be operated on in parallel. This class contains the basic operations available on all RDDs, such as `map`, `filter`, and `persist`. In addition, org.apache.spark.rdd.PairRDDFunctions contains operations available only on RDDs of key-value pairs, such as `groupByKey` and `join`; org.apache.spark.rdd.DoubleRDDFunctions contains operations available only on RDDs of Doubles; and org.apache.spark.rdd.SequenceFileRDDFunctions contains operations available on RDDs that can be saved as SequenceFiles. These operations are automatically available on any RDD of the right type (e.g. RDD[(Int, Int)] through implicit conversions when you `import org.apache.spark.SparkContext._`.

Internally, each RDD is characterized by five main properties:

- A list of partitions
- A function for computing each split
- A list of dependencies on other RDDs
- Optionally, a Partitioner for key-value RDDs (e.g. to say that the RDD is hash-partitioned)
- Optionally, a list of preferred locations to compute each split on (e.g. block locations for an HDFS file)

All of the scheduling and execution in Spark is done based on these methods, allowing each RDD to implement its own way of computing itself. Indeed, users can implement custom RDDs (e.g. for reading data from a new storage system) by overriding these functions. Please refer to the Spark paper for more details on RDD internals.

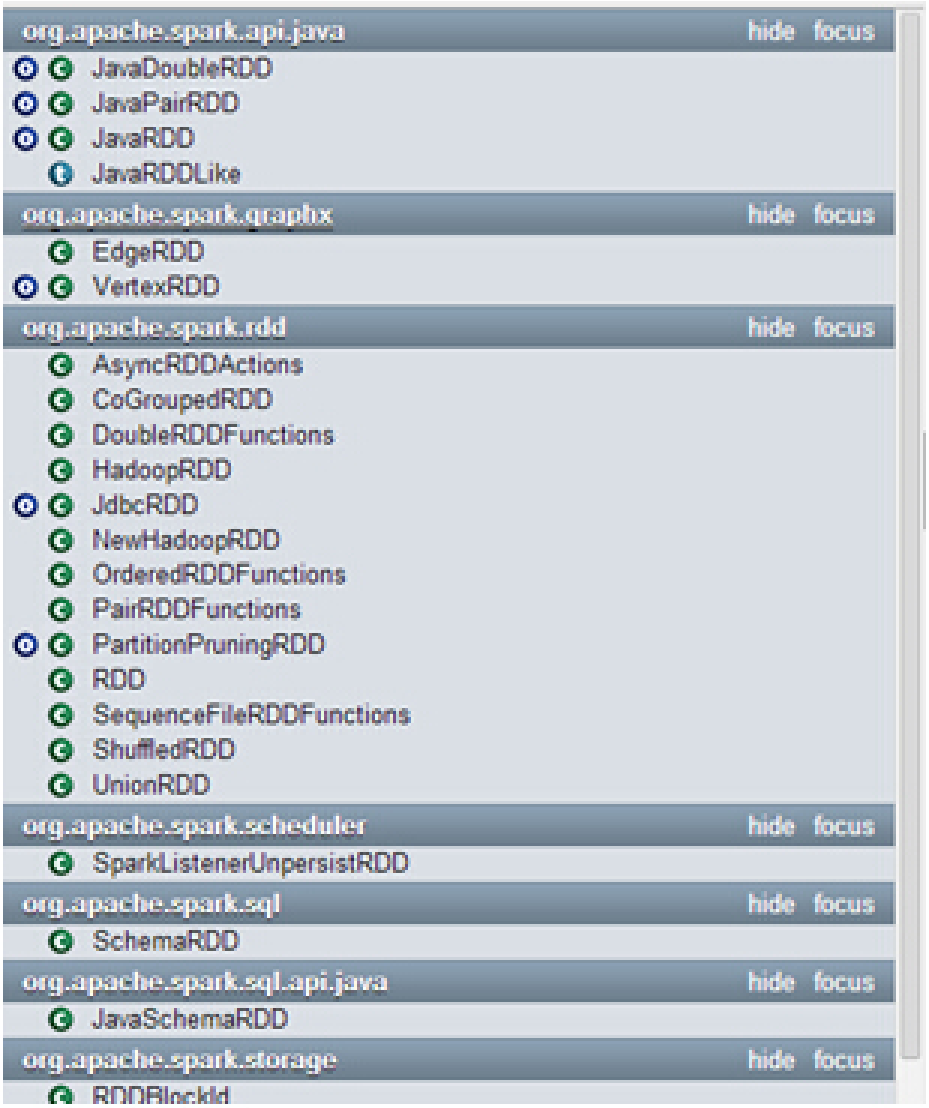
每个 RDD 的 API 如下所示：

```
// 计算某个分区
def compute(split: Partition, context: TaskContext): Iterator[T]

protected def getPartitions: Array[Partition]
// 依赖的父RDD，默认就是返回整个dependency序列
protected def getDependencies: Seq[Dependency[_]] = deps

protected def getPreferredLocations(split: Partition): Seq[String] = Nil
```

Spark 官方文档中给出了的众多的 RDD：



RDD 中的操作分为 transformations 和 actions 两种：

Transformations	*map*(*f* : T ⇒ U)	:	RDD[T] ⇒ RDD[U]
	filter(*f* : T ⇒ Bool)	:	RDD[T] ⇒ RDD[T]
	flatMap(*f* : T ⇒ Seq[U])	:	RDD[T] ⇒ RDD[U]
	sample(*fraction* : Float)	:	RDD[T] ⇒ RDD[T] (Deterministic sampling)
	groupByKey()	:	RDD[(K, V)] ⇒ RDD[(K, Seq[V])]
	reduceByKey(*f* : (V,V) ⇒ V)	:	RDD[(K, V)] ⇒ RDD[(K, V)]
	union()	:	(RDD[T], RDD[T]) ⇒ RDD[T]
	join()	:	(RDD[(K, V)], RDD[(K, W)]) ⇒ RDD[(K, (V, W))]
	cogroup()	:	(RDD[(K, V)], RDD[(K, W)]) ⇒ RDD[(K, (Seq[V], Seq[W]))]
	crossProduct()	:	(RDD[T], RDD[U]) ⇒ RDD[(T, U)]
	mapValues(*f* : V ⇒ W)	:	RDD[(K, V)] ⇒ RDD[(K, W)] (Preserves partitioning)
	sort(*c* : Comparator[K])	:	RDD[(K, V)] ⇒ RDD[(K, V)]
	partitionBy(*p* : Partitioner[K])	:	RDD[(K, V)] ⇒ RDD[(K, V)]
Actions	*count*()	:	RDD[T] ⇒ Long
	collect()	:	RDD[T] ⇒ Seq[T]
	reduce(*f* : (T,T) ⇒ T)	:	RDD[T] ⇒ T
	lookup(*k* : K)	:	RDD[(K, V)] ⇒ Seq[V] (On hash/range partitioned RDDs)
	save(*path* : String)	:	Outputs RDD to a storage system, *e.g.*, HDFS

Table 2: Transformations and actions available on RDDs in Spark. Seq[T] denotes a sequence of elements of type T.

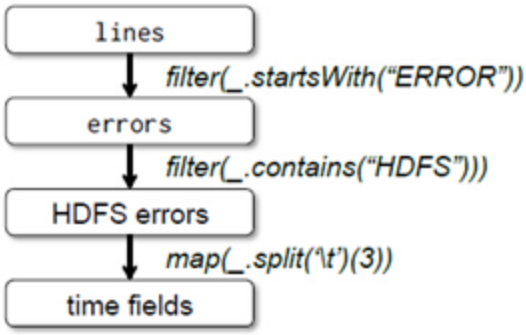
下面举一个例子来说明 RDD 的使用：

```
lines = spark.textFile("hdfs://...")
errors = lines.filter(_.startsWith("ERROR"))
errors.persist()

errors.count()

// Count errors mentioning MySQL:
errors.filter(_.contains("MySQL")).count()

// Return the time fields of errors mentioning HDFS as an array, assuming
// time is field number 3 in a tab-separated format:
errors.filter(_.contains("HDFS"))
      .map(_.split('\t')(3))
      .collect()
```

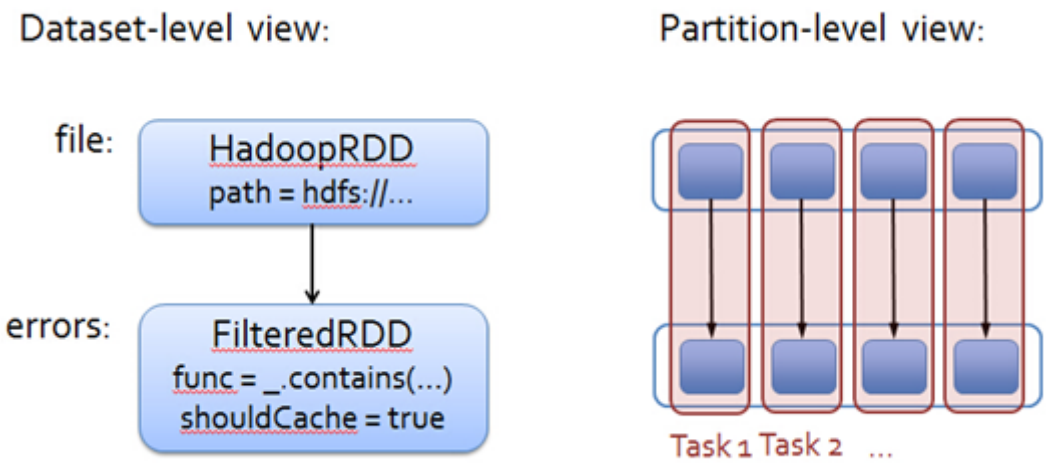


另外有两个特殊的 RDD:

他们都是 controlling operations：

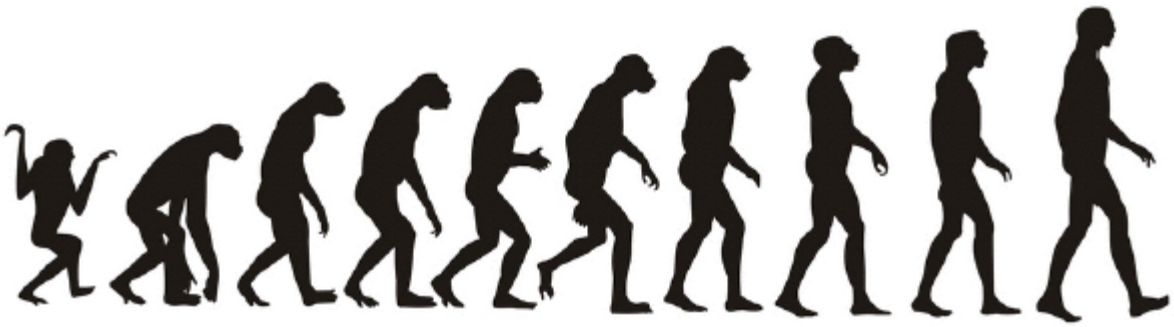
Persist:持久化RDD，修改了RDD的meta info中的storage level
Checkpoint:持久化RDD的同时切断Lineage，修改了RDD的meta info中的lineage
均返回经过修改的RDD对象自身而非新的RDD对象，均属Lazy操作

RDD 在执行的时候都是并行的：

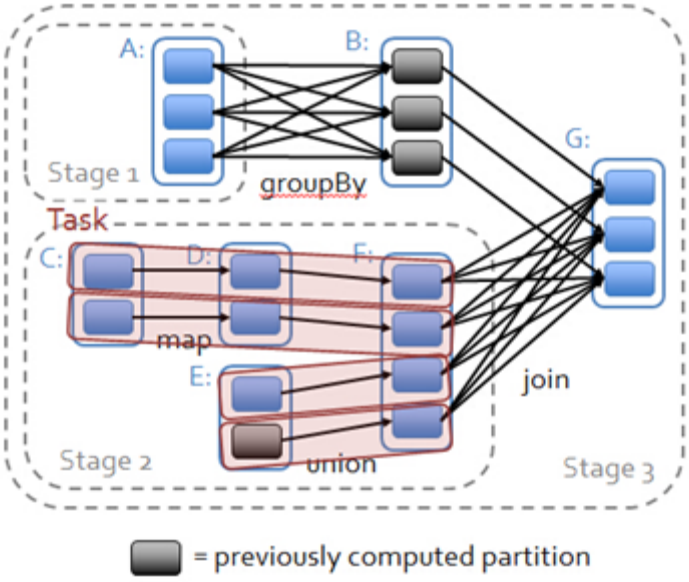


四， Spark 的高容错机制 lineage

基于 DAG 图，lineage 是轻量级而高效的：



操作之间相互具备 lineage 的关系，每个操作只关心其父操作，各个分片的数据之间互不影响，出现错误的时候只要恢复单个 Split 的特定部分即可：



Spark 亚太研究院

Spark 亚太研究院，提供 Spark、Hadoop、Android、Html5、云计算和移动互联网一站式解决方案。以帮助企业规划、部署、开发、培训和使用为核心，并规划和实施人才培训完整路径，提供源码研究和应用技术训练。

近期活动及相关课程

1、决战云计算大数据时代 Spark 亚太研究院 100 期公益大奖堂

每周四晚上 20：00—21:00

课程介绍：http://edu.51cto.com/course/course_id-1659.html#showDesc

报名参与：http://ke.qq.com/cgi-bin/courseDetail?course_id=6167

2、大数据 Spark 实战高手之路—熟练掌握 Scala 语言视频课程

国内第一个 Scala 视频学习课程！
成为 Spark 高手必备技能 必修课程
现在购买，即可享受套餐优惠！

课程地址： http://edu.51cto.com/pack/view/id-124.html