



《Spark编程基础（Scala版）》

教材官网: <http://dblab.xmu.edu.cn/post/spark/>

温馨提示: 编辑幻灯片母版, 可以修改每页PPT的厦大校徽和底部文字

第3章 Spark的设计与运行原理

(PPT版本号: 2018年7月版本)



扫一扫访问教材官网

林子雨

厦门大学计算机科学系

E-mail: ziyulin@xmu.edu.cn ➡

主页: <http://www.cs.xmu.edu.cn/linziyu>





课程教材

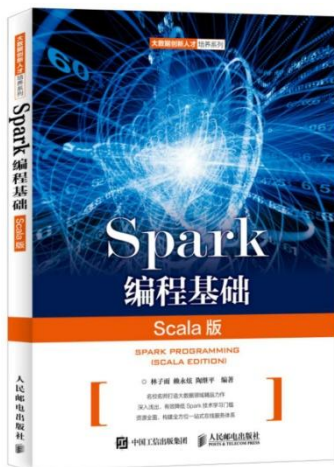
本套讲义PPT属于以下教材的配套材料

《Spark编程基础（Scala版）》

厦门大学 林子雨，赖永炫，陶继平 编著

披荆斩棘，在大数据丛林中开辟学习捷径
填沟削坎，为快速学习Spark技术铺平道路
深入浅出，有效降低Spark技术学习门槛
资源全面，构建全方位一站式在线服务体系

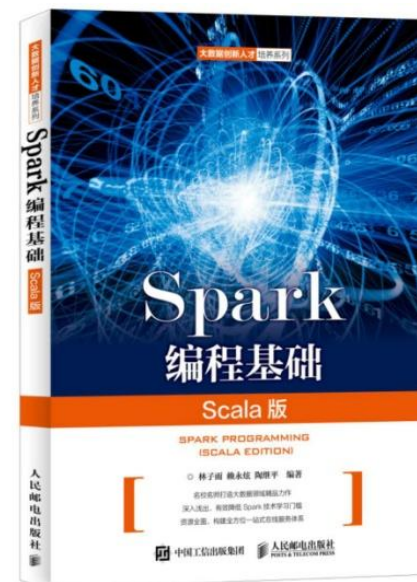
人民邮电出版社出版发行，ISBN:978-7-115-48816-9
教材官网：<http://dbllab.xmu.edu.cn/post/spark/>



本书以Scala作为开发Spark应用程序的编程语言，系统介绍了Spark编程的基础知识。全书共8章，内容包括大数据技术概述、Scala语言基础、Spark的设计与运行原理、Spark环境搭建和使用方法、RDD编程、Spark SQL、Spark Streaming、Spark MLlib等。本书每个章节都安排了入门级的编程实践操作，以便读者更好地学习和掌握Spark编程方法。本书官网免费提供了全套的在线教学资源，包括讲义PPT、习题、源代码、软件、数据集、授课视频、上机实验指南等。



课程配套授课视频



课程在线视频地址：<http://dblab.xmu.edu.cn/post/10482/>



提纲

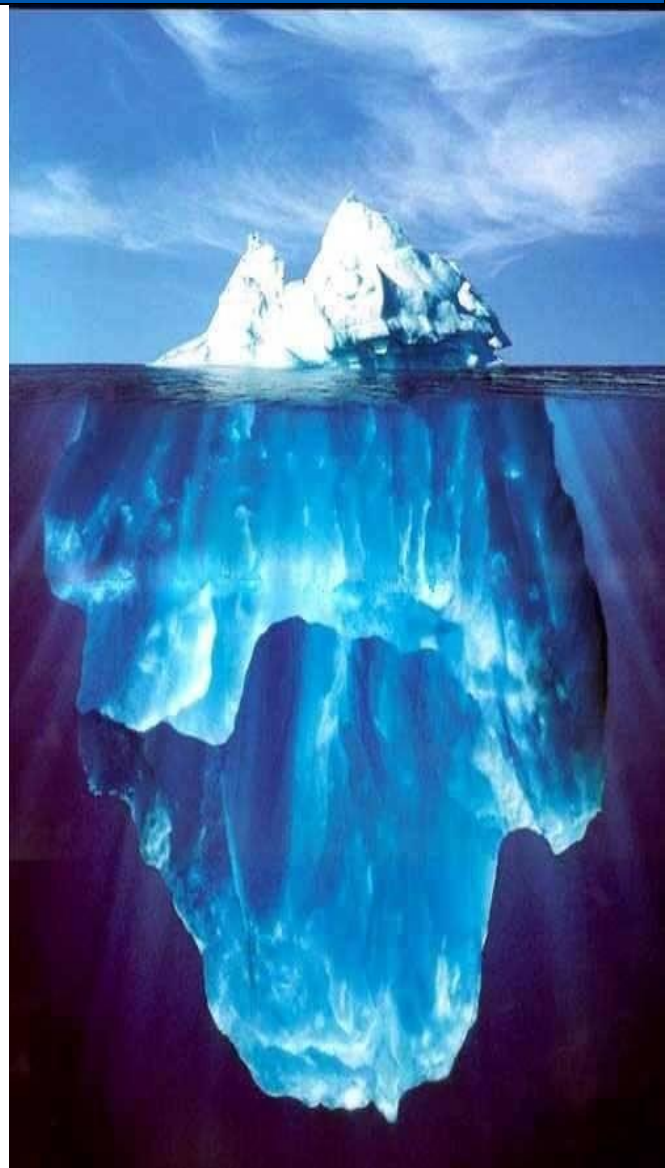
- 3.1 Spark概述
- 3.2 Spark生态系统
- 3.3 Spark运行架构
- 3.4 Spark的部署方式



高校大数据课程

公 共 服 务 平 台

百度搜索厦门大学数据库实验室网站访问平台





3.1 Spark概述

3.1.1 Spark简介

3.1.2 Scala简介

3.1.3 Spark与Hadoop的比较



3.1.1 Spark简介

- Spark最初由美国加州大学伯克利分校（UC Berkeley）的AMP实验室于2009年开发，是基于内存计算的大数据并行计算框架，可用于构建大型的、低延迟的数据分析应用程序
- 2013年Spark加入Apache孵化器项目后发展迅猛，如今已成为Apache软件基金会最重要的三大分布式计算系统开源项目之一（Hadoop、Spark、Storm）
- Spark在2014年打破了Hadoop保持的基准排序纪录
 - Spark/206个节点/23分钟/100TB数据
 - Hadoop/2000个节点/72分钟/100TB数据
 - Spark用十分之一的计算资源，获得了比Hadoop快3倍的速度



3.1.1 Spark简介

Spark具有如下几个主要特点：

- 运行速度快：使用DAG执行引擎以支持循环数据流与内存计算
- 容易使用：支持使用Scala、Java、Python和R语言进行编程，可以通过Spark Shell进行交互式编程
- 通用性：Spark提供了完整而强大的技术栈，包括SQL查询、流式计算、机器学习和图算法组件
- 运行模式多样：可运行于独立的集群模式中，可运行于Hadoop中，也可运行于Amazon EC2等云环境中，并且可以访问HDFS、Cassandra、HBase、Hive等多种数据源



3.1.1 Spark简介

Spark如今已吸引了国内外各大公司的注意，如腾讯、淘宝、百度、亚马逊等公司均不同程度地使用了Spark来构建大数据分析应用，并应用到实际的生产环境中



图 谷歌趋势：Spark与Hadoop对比



3.1.2 Scala简介

Scala是一门现代的多范式编程语言，运行于Java平台（JVM，Java 虚拟机），并兼容现有的Java程序

Scala的特性：

- Scala具备强大的并发性，支持函数式编程，可以更好地支持分布式系统
- Scala语法简洁，能提供优雅的API

Scala兼容Java，运行速度快，且能融合到Hadoop生态圈中

Scala是Spark的主要编程语言，但Spark还支持Java、Python、R作为编程语言

Scala的优势是提供了REPL（Read-Eval-Print Loop，交互式解释器），提高程序开发效率



3.1.3 Spark与Hadoop的对比

Hadoop存在如下一些缺点：

- 表达能力有限
- 磁盘IO开销大
- 延迟高
 - 任务之间的衔接涉及IO开销
 - 在前一个任务执行完成之前，其他任务就无法开始，难以胜任复杂、多阶段的计算任务



3.1.3 Spark与Hadoop的对比

Spark在借鉴Hadoop MapReduce优点的同时，很好地解决了MapReduce所面临的问题

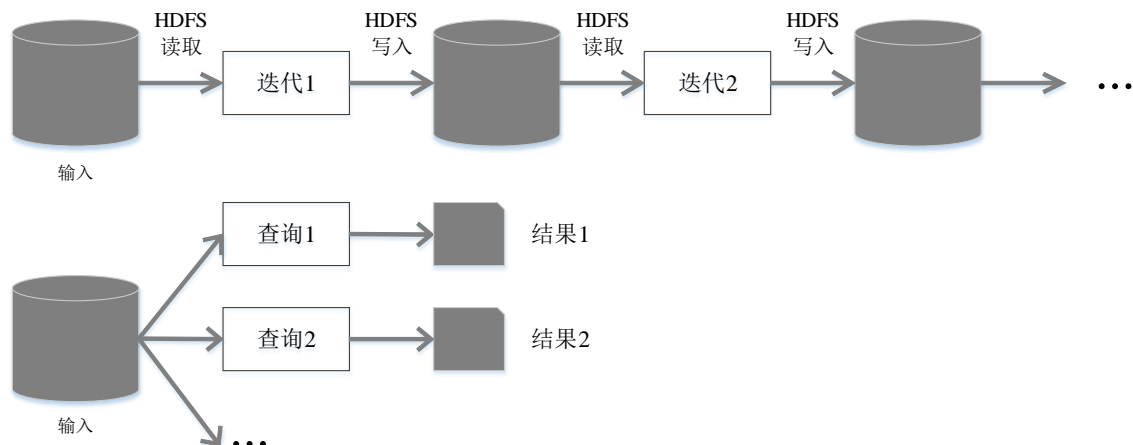
相比于Hadoop MapReduce，Spark主要具有如下优点：

- Spark的计算模式也属于MapReduce，但不局限于Map和Reduce操作，还提供了多种数据集操作类型，编程模型比Hadoop MapReduce更灵活
- Spark提供了内存计算，可将中间结果放到内存中，对于迭代运算效率更高

Spark基于DAG的任务调度执行机制，要优于Hadoop MapReduce的迭代执行机制



3.1.3 Spark与Hadoop的对比



(a) Hadoop MapReduce执行流程

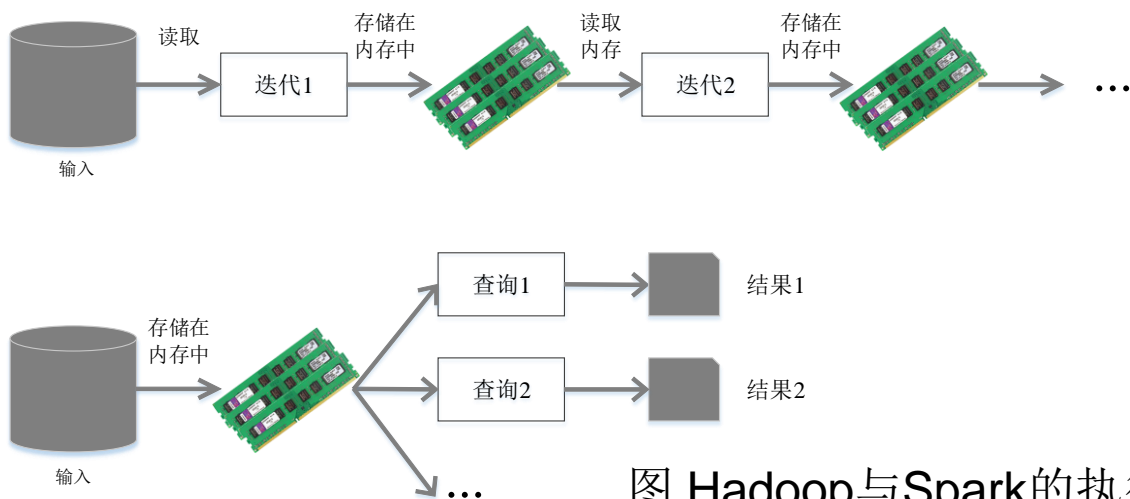


图 Hadoop与Spark的执行流程对比

(b) Spark执行流程



3.1.3 Spark与Hadoop的对比

- 使用Hadoop进行迭代计算非常耗资源
- Spark将数据载入内存后，之后的迭代计算都可以直接使用内存中的中间结果作运算，避免了从磁盘中频繁读取数据

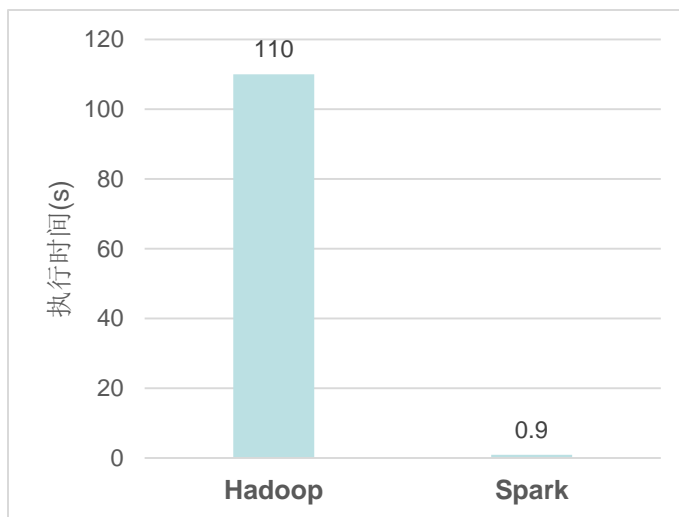


图 Hadoop与Spark执行逻辑回归的时间对比



3.2 Spark生态系统

在实际应用中，大数据处理主要包括以下三个类型：

- 复杂的批量数据处理：通常时间跨度在数十分钟到数小时之间
- 基于历史数据的交互式查询：通常时间跨度在数十秒到数分钟之间
- 基于实时数据流的数据处理：通常时间跨度在数百毫秒到数秒之间

当同时存在以上三种场景时，就需要同时部署三种不同的软件

- 比如: MapReduce / Impala / Storm

这样做难免会带来一些问题：

- 不同场景之间输入输出数据无法做到无缝共享，通常需要进行数据格式的转换
- 不同的软件需要不同的开发和维护团队，带来了较高的使用成本
- 比较难以对同一个集群中的各个系统进行统一的资源协调和分配



3.2 Spark生态系统

- Spark的设计遵循“一个软件栈满足不同应用场景”的理念，逐渐形成了一套完整的生态系统
- 既能够提供内存计算框架，也可以支持SQL即席查询、实时流式计算、机器学习和图计算等
- Spark可以部署在资源管理器YARN之上，提供一站式的大数据解决方案
- 因此，Spark所提供的生态系统足以应对上述三种场景，即同时支持批处理、交互式查询和流数据处理



3.2 Spark生态系统

Spark生态系统已经成为伯克利数据分析软件栈BDAS（Berkeley Data Analytics Stack）的重要组成部分

Access and Interfaces	Spark Streaming	BlinkDB	GraphX	MLBase
		Spark SQL		MLlib
Processing Engine	Spark Core			
Storage	Tachyon			
	HDFS, S3			
Resource Virtualization	Mesos		Hadoop Yarn	

图 BDAS架构

Spark的生态系统主要包含了Spark Core、Spark SQL、Spark Streaming（Structured Streaming）、MLLib和GraphX等组件



3.2 Spark生态系统

表1 Spark生态系统组件的应用场景

应用场景	时间跨度	其他框架	Spark生态系统中的组件
复杂的批量数据处理	小时级	MapReduce、Hive	Spark
基于历史数据的交互式查询	分钟级、秒级	Impala、Dremel、Drill	Spark SQL
基于实时数据流的数据处理	毫秒、秒级	Storm、S4	Spark Streaming Structured Streaming
基于历史数据的数据挖掘	-	Mahout	MLlib
图结构数据的处理	-	Pregel、Hama	GraphX



3.3 Spark运行架构

3.3.1 基本概念

3.3.2 架构设计

3.3.3 Spark运行基本流程

3.3.4 RDD的设计与运行原理



3.3.1 基本概念

- **RDD**: 是Resilient Distributed Dataset（弹性分布式数据集）的简称，是分布式内存的一个抽象概念，提供了一种高度受限的共享内存模型
- **DAG**: 是Directed Acyclic Graph（有向无环图）的简称，反映RDD之间的依赖关系
- **Executor**: 是运行在工作节点（WorkerNode）的一个进程，负责运行Task
- **应用（Application）**: 用户编写的Spark应用程序
- **任务（Task）**: 运行在Executor上的工作单元
- **作业（Job）**: 一个作业包含多个RDD及作用于相应RDD上的各种操作
- **阶段（Stage）**: 是作业的基本调度单位，一个作业会分为多组任务，每组任务被称为阶段，或者也被称为任务集合，代表了一组关联的、相互之间没有Shuffle依赖关系的任务组成的任务集



3.3.2 架构设计

- Spark运行架构包括集群资源管理器（**Cluster Manager**）、运行作业任务的工作节点（**Worker Node**）、每个应用的任务控制节点（**Driver**）和每个工作节点上负责具体任务的执行进程（**Executor**）
- 资源管理器可以自带或Mesos或YARN

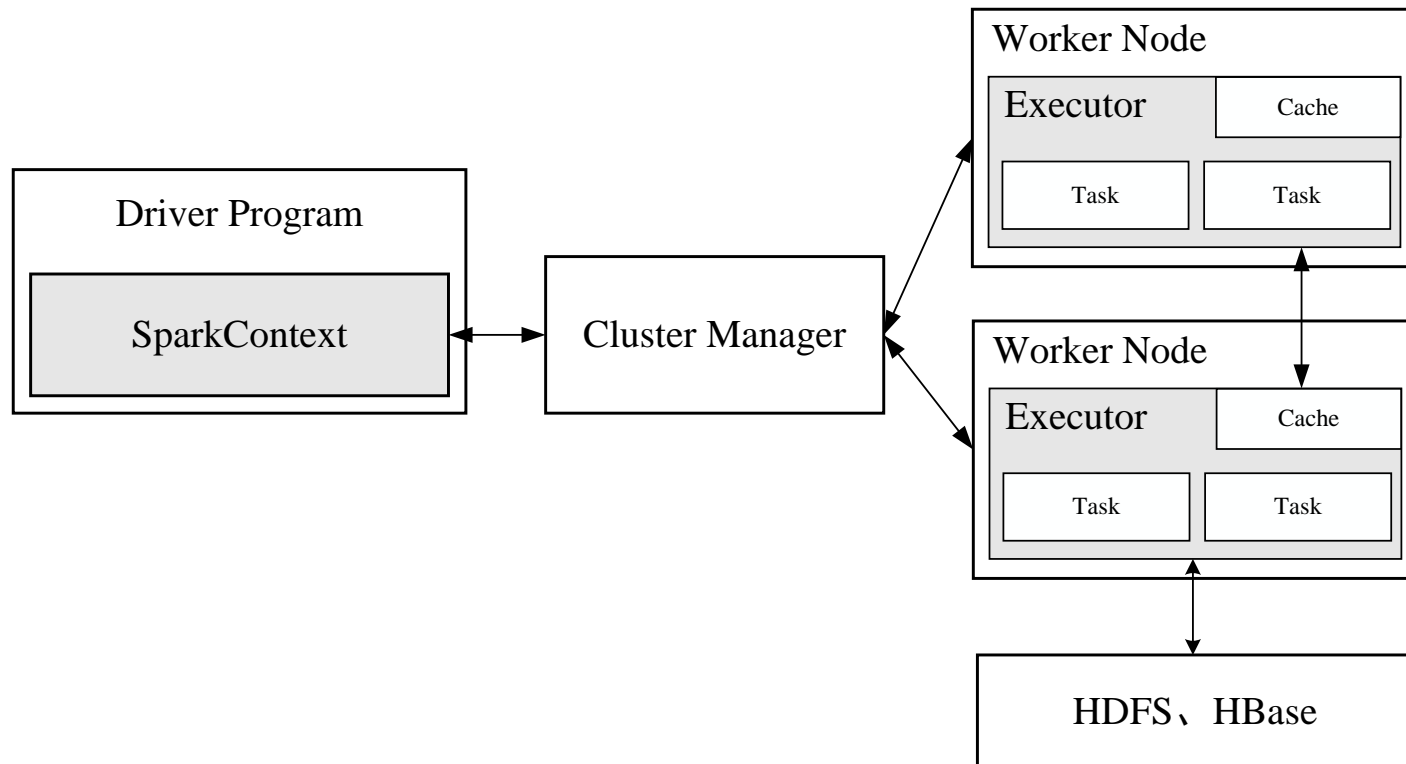


图 Spark运行架构



3.3.2 架构设计

- 一个应用由一个**Driver**和若干个作业构成，一个作业由多个阶段构成，一个阶段由多个没有**Shuffle**关系的任务组成
- 当执行一个应用时，**Driver**会向集群管理器申请资源，启动**Executor**，并向**Executor**发送应用程序代码和文件，然后在**Executor**上执行任务，运行结束后，执行结果会返回给**Driver**，或者写到**HDFS**或者其他数据库中

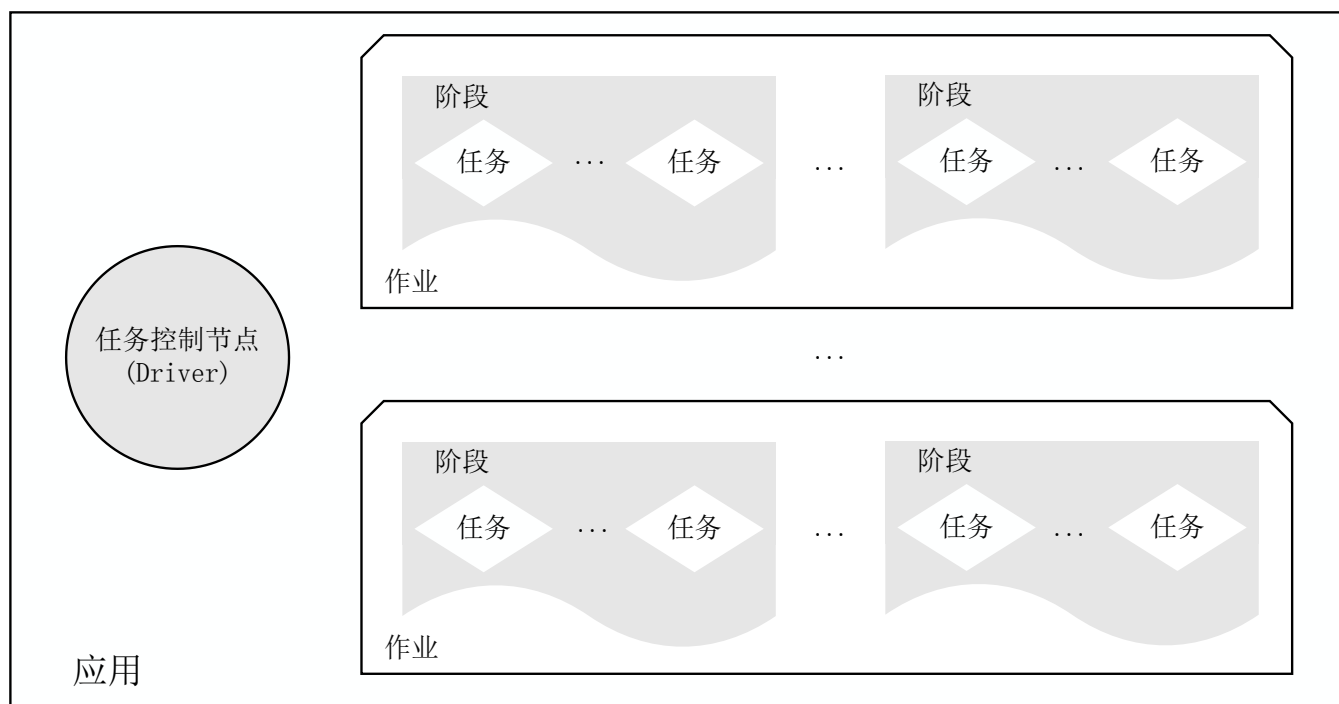


图 Spark中各种概念之间的相互关系



3.3.3 Spark运行基本流程

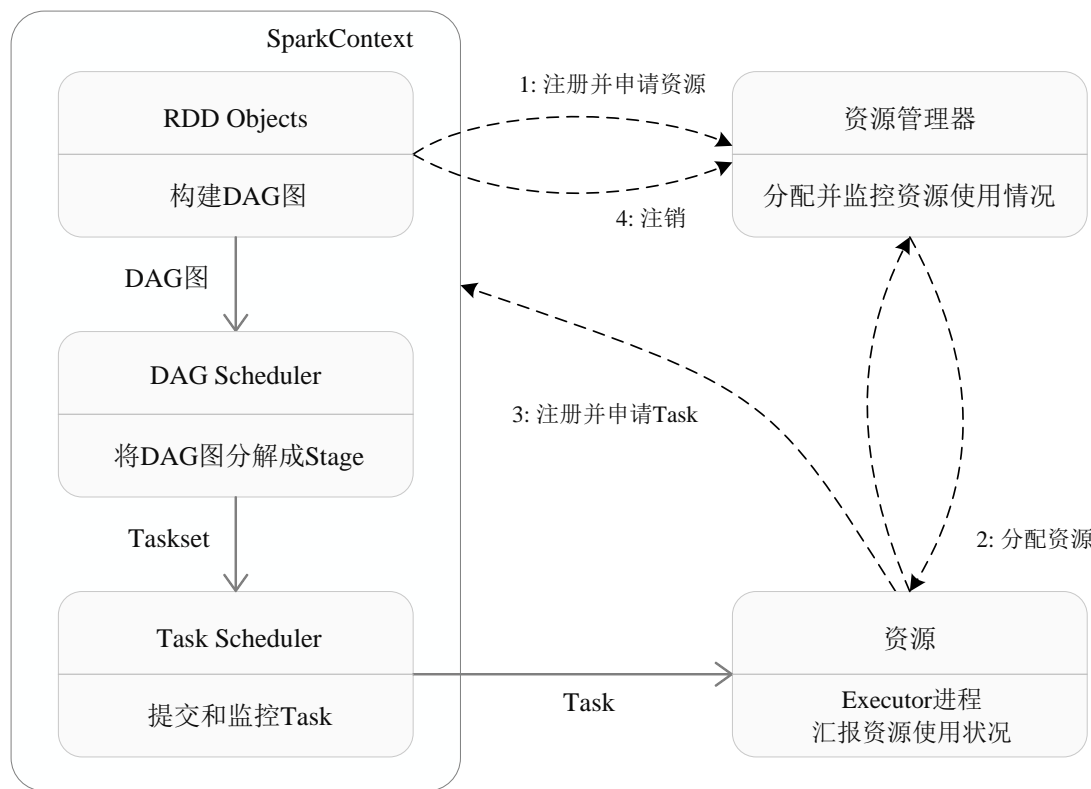


图 Spark运行基本流程图

SparkContext对象代表了和一个集群的连接

(1) 首先为应用构建起基本的运行环境，即由Driver创建一个SparkContext，进行资源的申请、任务的分配和监控

(2) 资源管理器为Executor分配资源，并启动Executor进程

(3) SparkContext根据RDD的依赖关系构建DAG图，DAG图提交给DAGScheduler解析成Stage，然后把一个个TaskSet提交给底层调度器TaskScheduler处理；Executor向SparkContext申请Task，Task Scheduler将Task发放给Executor运行，并提供应用程序代码

(4) Task在Executor上运行，把执行结果反馈给TaskScheduler，然后反馈给DAGScheduler，运行完毕后写入数据并释放所有资源



3.3.4 RDD运行原理

- 1.RDD设计背景
- 2.RDD概念
- 3.RDD特性
- 4.RDD之间的依赖关系
- 5.阶段的划分
- 6.RDD运行过程



3.3.4 RDD运行原理

1.RDD设计背景

- 许多迭代式算法（比如机器学习、图算法等）和交互式数据挖掘工具，共同之处是，不同计算阶段之间会重用中间结果
- 目前的MapReduce框架都是把中间结果写入到稳定存储（比如磁盘）中，带来了大量的数据复制、磁盘IO和序列化开销
- RDD就是为了满足这种需求而出现的，它提供了一个抽象的数据架构，我们不必担心底层数据的分布式特性，只需将具体的应用逻辑表达为一系列转换处理，不同RDD之间的转换操作形成依赖关系，可以实现管道化，避免中间数据存储



3.3.4 RDD运行原理

2.RDD概念

- 一个**RDD**就是一个分布式对象集合，本质上是一个只读的分区记录集合，每个**RDD**可分成多个分区，每个分区就是一个数据集片段，并且一个**RDD**的不同分区可以被保存到集群中不同的节点上，从而可以在集群中的不同节点上进行并行计算
- **RDD**提供了一种高度受限的共享内存模型，即**RDD**是只读的记录分区的集合，不能直接修改，只能基于稳定的物理存储中的数据集创建**RDD**，或者通过在其他**RDD**上执行确定的转换操作（如**map**、**join**和**group by**）而创建得到新的**RDD**



3.3.4 RDD运行原理

2.RDD概念

- RDD提供了一组丰富的操作以支持常见的数据运算，分为“动作”（Action）和“转换”（Transformation）两种类型
- RDD提供的转换接口都非常简单，都是类似map、filter、groupBy、join等粗粒度的数据转换操作，而不是针对某个数据项的细粒度修改（不适合网页爬虫）
- 表面上RDD的功能很受限、不够强大，实际上RDD已经被实践证明可以高效地表达许多框架的编程模型（比如MapReduce、SQL、Pregel）
- Spark用Scala语言实现了RDD的API，程序员可以通过调用API实现对RDD的各种操作



3.3.4 RDD运行原理

2.RDD概念

RDD典型的执行过程如下：

- RDD读入外部数据源进行创建
- RDD经过一系列的转换（Transformation）操作，每一次都会产生不同的RDD，供给下一个转换操作使用
- 最后一个RDD经过“动作”操作进行转换，并输出到外部数据源

这一系列处理称为一个Lineage（血缘关系），即DAG拓扑排序的结果

优点：惰性调用、管道化、避免同步等待、不需要保存中间结果、每次操作变得简单

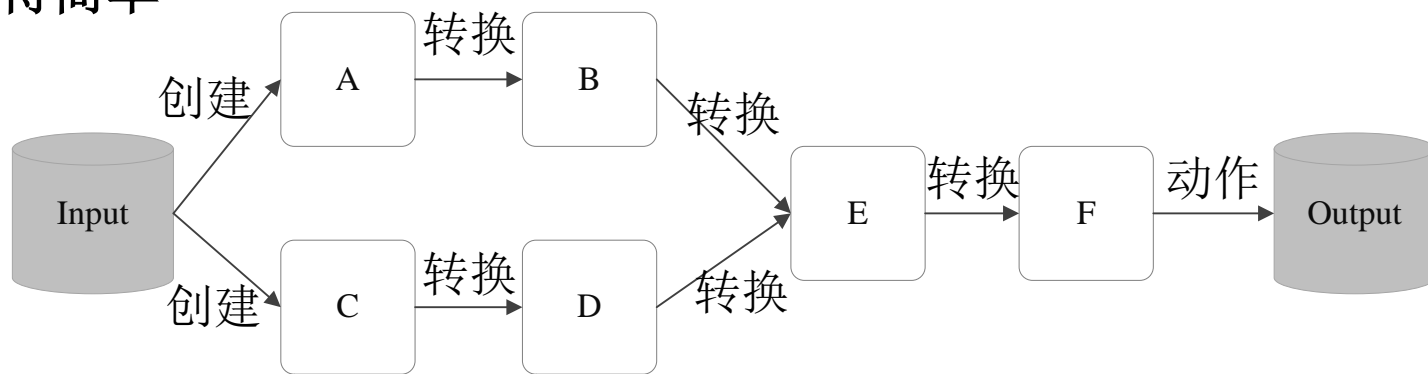


图 RDD执行过程的一个实例



3.3.4 RDD运行原理

3.RDD特性

Spark采用RDD以后能够实现高效计算的原因主要在于：

(1) 高效的容错性

- 现有容错机制：数据复制或者记录日志
- **RDD**：血缘关系、重新计算丢失分区、无需回滚系统、重算过程在不同节点之间并行、只记录粗粒度的操作

(2) 中间结果持久化到内存，数据在内存中的多个RDD操作之间进行传递，避免了不必要的读写磁盘开销

(3) 存放的数据可以是Java对象，避免了不必要的对象序列化和反序列化



3.3.4 RDD运行原理

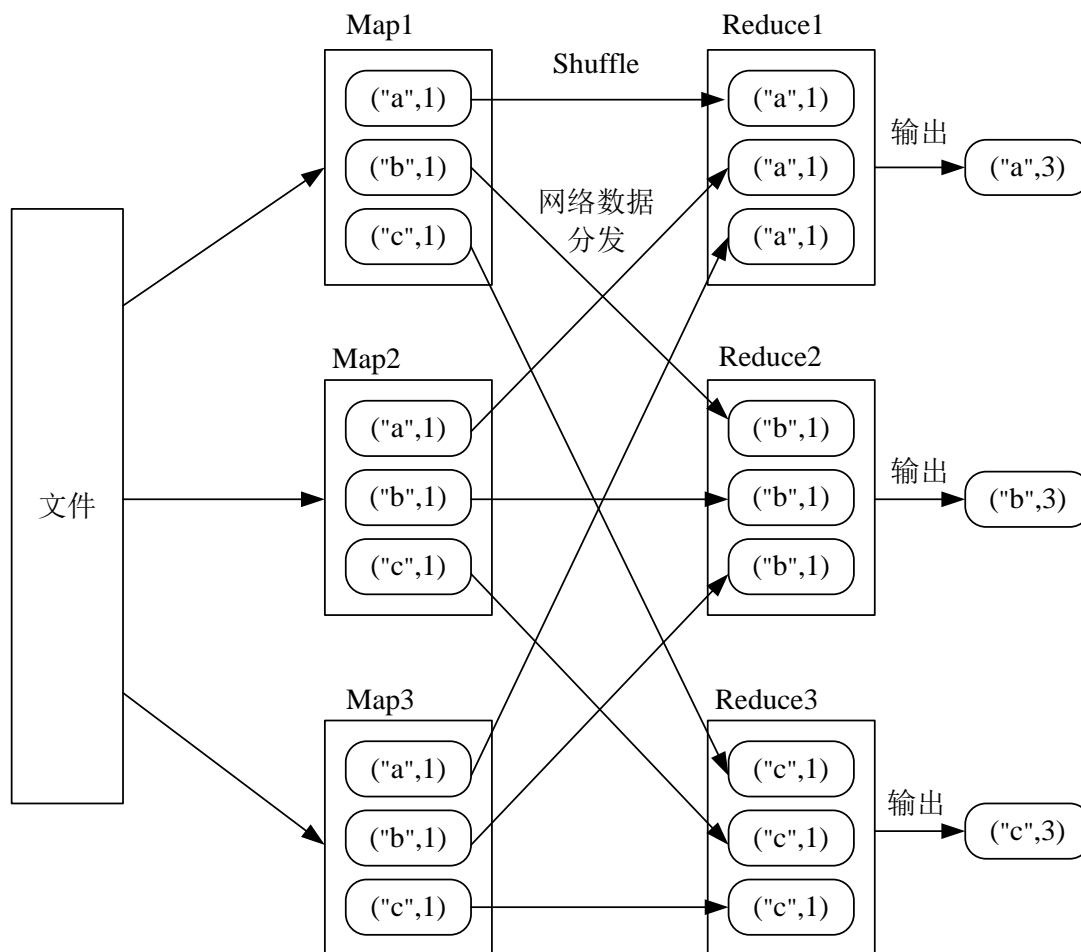
4. RDD之间的依赖关系

- Shuffle操作
 - 什么是Shuffle操作
- 窄依赖和宽依赖
 - 是否包含Shuffle操作是区分窄依赖和宽依赖的根据



3.3.4 RDD运行原理

4. RDD之间的依赖关系——Shuffle操作

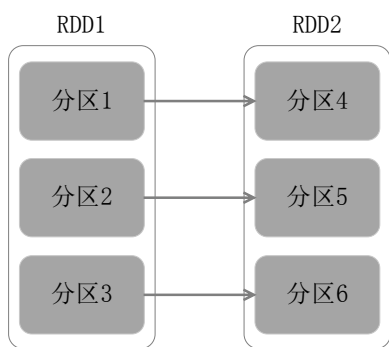


一个关于Shuffle 操作的简单实例

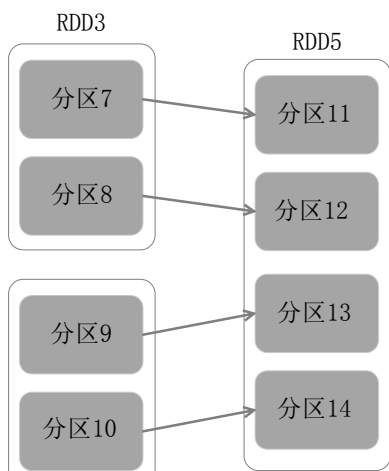


3.3.4 RDD运行原理

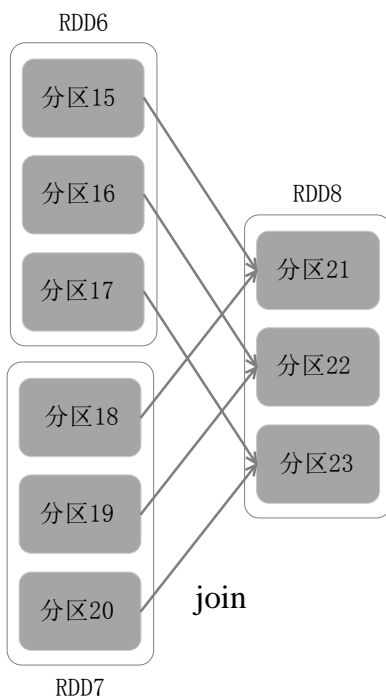
4. RDD之间的依赖关系——窄依赖和宽依赖



map, filter

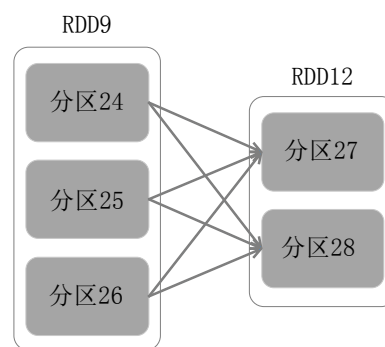


union

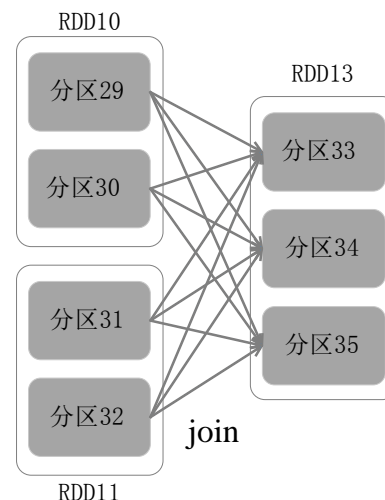


join

(a)窄依赖



groupByKey



join

(b)宽依赖

- 窄依赖表现为一个父RDD的分区对应于一个子RDD的分区或多个父RDD的分区对应于一个子RDD的分区
- 宽依赖则表现为存在一个父RDD的一个分区对应一个子RDD的多个分区

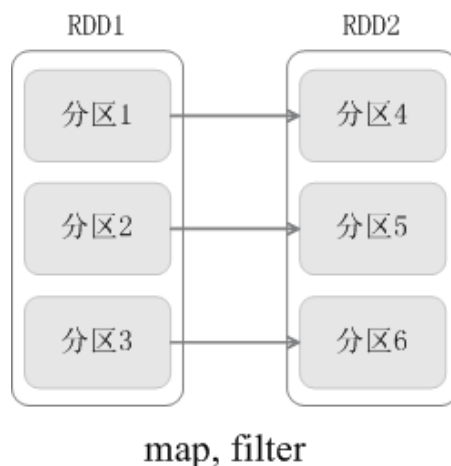
图 窄依赖与宽依赖的区别



3.3.4 RDD运行原理

5.阶段的划分

Spark 根据DAG 图中的RDD 依赖关系，把一个作业分成多个阶段。阶段划分的依据是窄依赖和宽依赖。对于宽依赖和窄依赖而言，窄依赖对于作业优化很有利，宽依赖无法优化。逻辑上，每个RDD 操作都是一个fork/join（一种用于并行执行任务的框架），把计算fork 到每个RDD 分区，完成计算后对各个分区得到的结果进行join 操作，然后fork/join下一个RDD 操作。



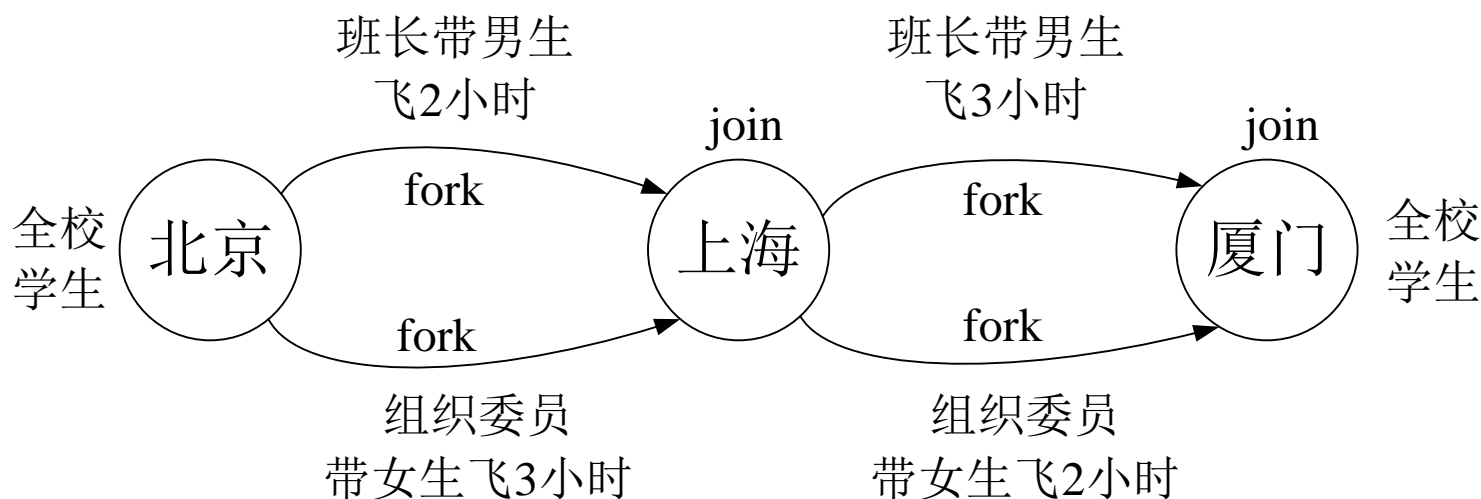


3.3.4 RDD运行原理

5.阶段的划分

fork/join的优化原理

举例：一个学校（含2个班级）完成从北京到厦门的长征

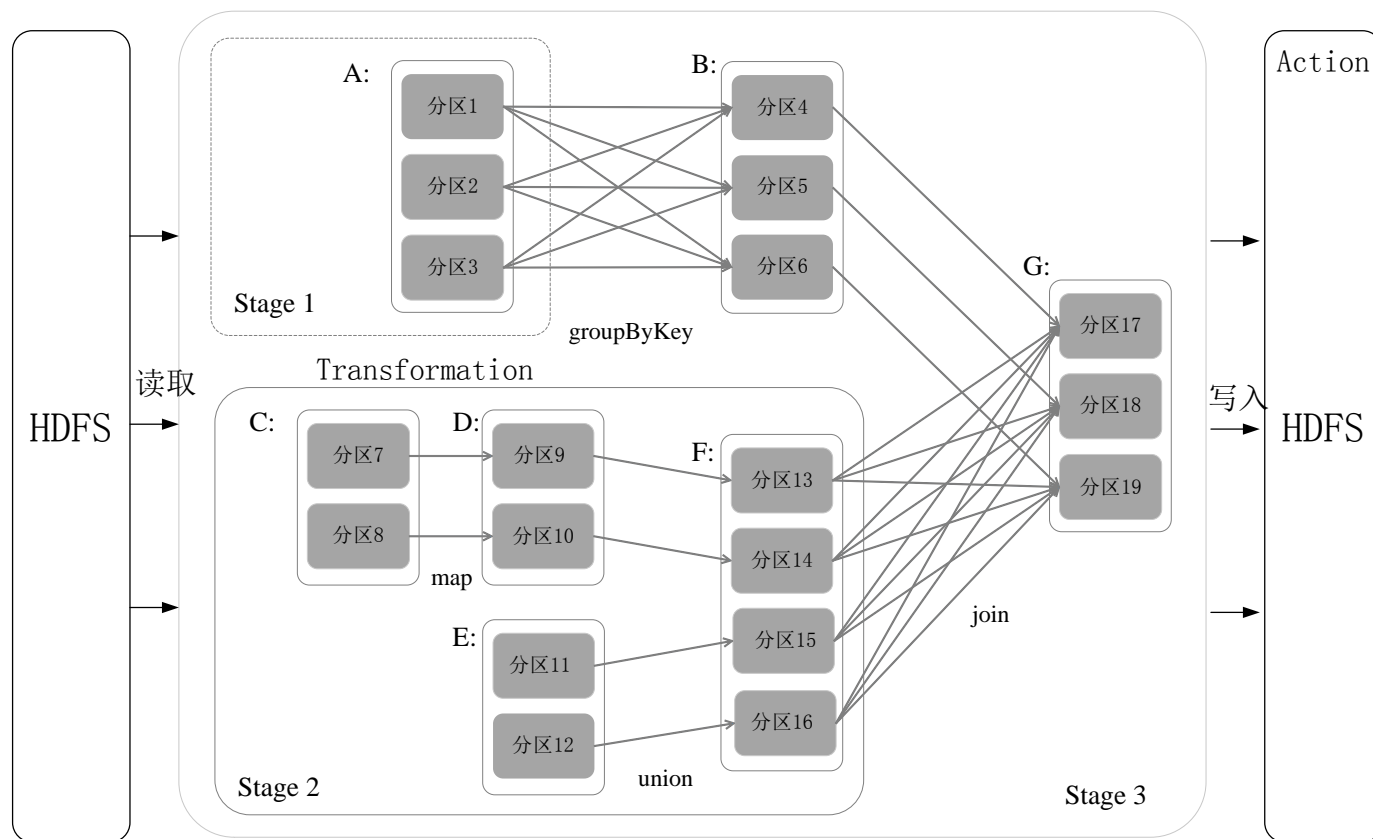




3.3.4 RDD运行原理

5.阶段的划分

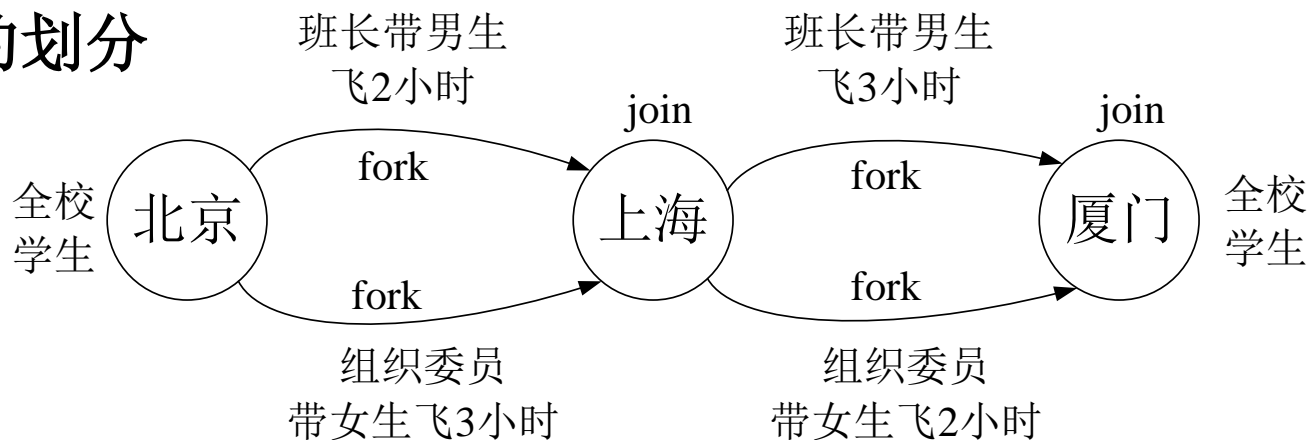
窄依赖可以实现“流水线”优化
宽依赖无法实现“流水线”优化



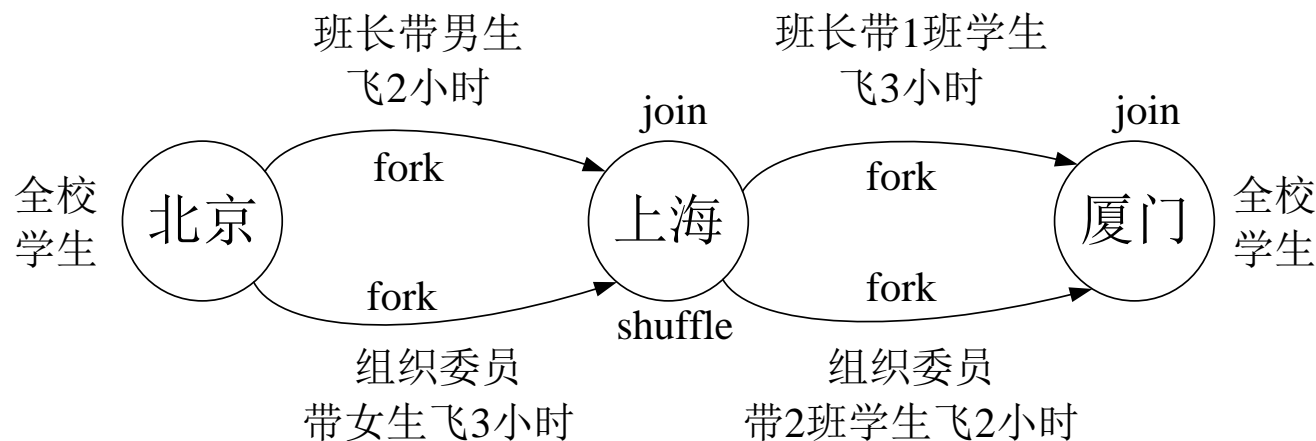


3.3.4 RDD运行原理

5.阶段的划分



(a) 窄依赖



(a) 宽依赖



3.3.4 RDD运行原理

5.阶段的划分

Spark根据DAG图中的RDD依赖关系，把一个作业分成多个阶段。对于宽依赖和窄依赖而言，窄依赖对于作业优化很有利。只有窄依赖可以实现流水线优化，宽依赖包含Shuffle过程，无法实现流水线方式处理。

Spark通过分析各个RDD的依赖关系生成了DAG，再通过分析各个RDD中的分区之间的依赖关系来决定如何划分

Stage，具体划分方法是：

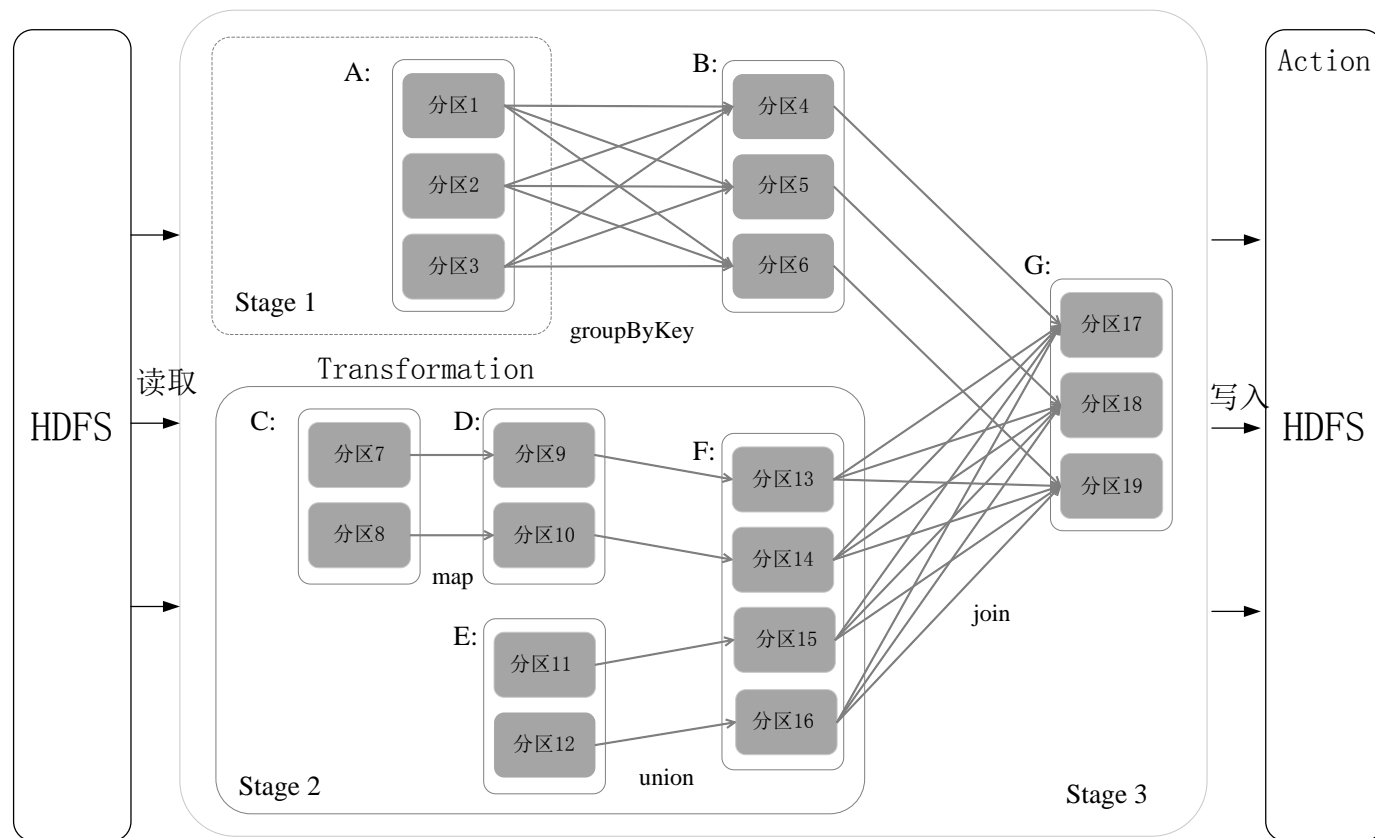
- 在DAG中进行反向解析，遇到宽依赖就断开
- 遇到窄依赖就把当前的RDD加入到Stage中
- 将窄依赖尽量划分在同一个Stage中，可以实现流水线计算



3.3.4 RDD运行原理

5.Stage的划分

被分成三个Stage，在Stage2中，从map到union都是窄依赖，这两步操作可以形成一个流水线操作



流水线操作实例
分区7通过map操作生成的分区9，可以不用等待分区8到分区10这个map操作的计算结束，而是继续进行union操作，得到分区13，这样流水线执行大大提高了计算的效率

图 根据RDD分区的依赖关系划分Stage



3.3.4 RDD运行原理

6.RDD运行过程

通过上述对RDD概念、依赖关系和Stage划分的介绍，结合之前介绍的Spark运行基本流程，再总结一下RDD在Spark架构中的运行过程：

- (1) 创建RDD对象；
- (2) SparkContext负责计算RDD之间的依赖关系，构建DAG；
- (3) DAGScheduler负责把DAG图分解成多个Stage，每个Stage中包含了多个Task，每个Task会被TaskScheduler分发给各个WorkerNode上的Executor去执行。

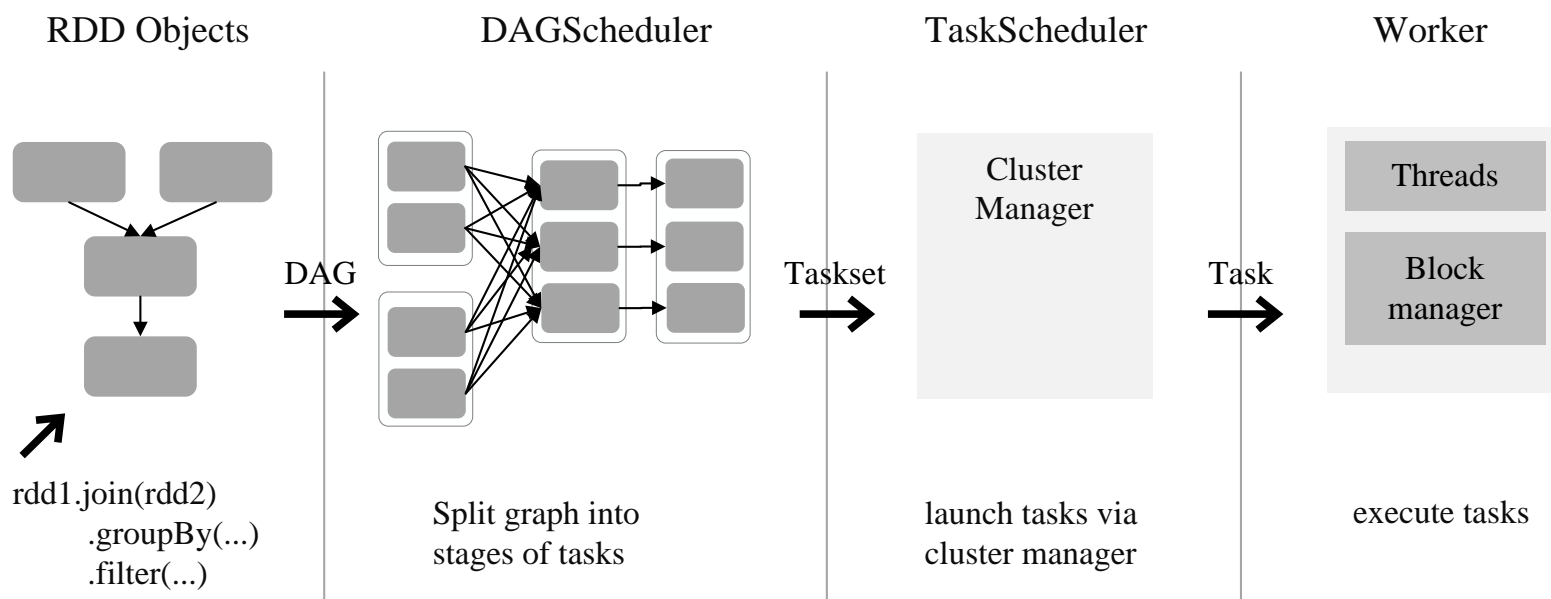


图 RDD在Spark中的运行过程



3.4 Spark的部署方式

Spark支持三种不同类型的部署方式，包括：

- Standalone（类似于MapReduce1.0，slot为资源分配单位）
- Spark on Mesos（和Spark有血缘关系，更好支持Mesos）
- Spark on YARN

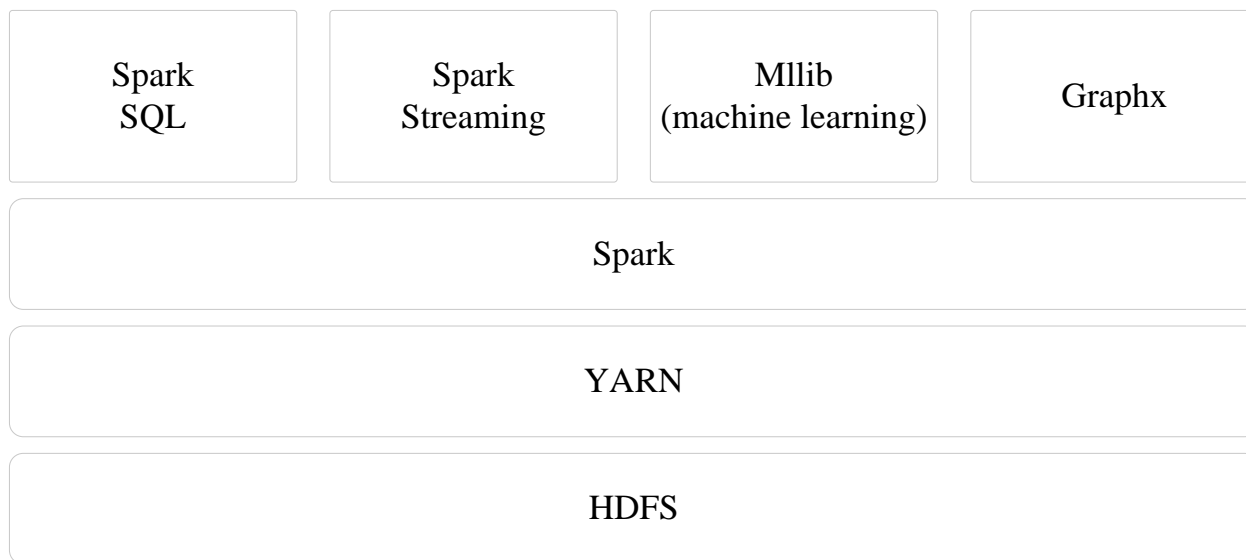


图 Spark on Yarn架构



讨论：Spark和Hadoop

- 虽然Spark很快，但现在在生产环境中仍然不尽人意，无论扩展性、稳定性、管理性等方面都需要进一步增强
- 同时，Spark在流处理领域能力有限，如果要实现亚秒级或大容量的数据获取或处理需要其他流处理产品。Cloudera宣布旨在让Spark流数据技术适用于80%的使用场合，就考虑到了这一缺陷。我们确实看到实时分析（而非简单数据过滤或分发）场景中，很多以前使用S4或Storm等流式处理引擎的实现已经逐渐被Kafka+Spark Streaming代替
- Spark的流行将逐渐让MapReduce、Tez走进博物馆
- Hadoop现在分三块HDFS/MR/YARN，Spark比Hadoop性能好，只是Spark作为一个计算引擎，比MR的性能要好。但它的存储和调度框架还是依赖于HDFS/YARN，Spark也有自己的调度框架，但仍然非常不成熟，基本不可商用



附录A：主讲教师林子雨简介



主讲教师：林子雨

单位：厦门大学计算机科学系

E-mail: ziyulin@xmu.edu.cn

个人网页: <http://www.cs.xmu.edu.cn/linziyu>

数据库实验室网站: <http://dblab.xmu.edu.cn>



扫一扫访问个人主页

林子雨，男，1978年出生，博士（毕业于北京大学），现为厦门大学计算机科学系助理教授（讲师），曾任厦门大学信息科学与技术学院院长助理、晋江市发展和改革委员会副局长。中国计算机学会数据库专业委员会委员，中国计算机学会信息系统专业委员会委员。国内高校首个“数字教师”提出者和建设者，厦门大学数据库实验室负责人，厦门大学云计算与大数据研究中心主要建设者和骨干成员，2013年度和2017年度厦门大学教学类奖教金获得者，荣获2017年福建省精品在线开放课程和2017年厦门大学高等教育成果二等奖。主要研究方向为数据库、数据仓库、数据挖掘、大数据、云计算和物联网，并以第一作者身份在《软件学报》《计算机学报》和《计算机研究与发展》等国家重点期刊以及国际学术会议上发表多篇学术论文。作为项目负责人主持的科研项目包括1项国家自然科学基金青年基金项目(No.61303004)、1项福建省自然科学基金青年基金项目(No.2013J05099)和1项中央高校基本科研业务费项目(No.2011121049)，主持的教改课题包括1项2016年福建省教改课题和1项2016年教育部产学协作育人项目，同时，作为课题负责人完成了国家发改委城市信息化重大课题、国家物联网重大应用示范工程区域试点泉州市工作方案、2015泉州市互联网经济调研等课题。中国高校首个“数字教师”提出者和建设者，2009年至今，“数字教师”大平台累计向网络免费发布超过500万字高价值的研究和教学资料，累计网络访问量超过500万次。打造了中国高校大数据教学知名品牌，编著出版了中国高校第一本系统介绍大数据知识的专业教材《大数据技术原理与应用》，并成为京东、当当网等网店畅销书籍；建设了国内高校首个大数据课程公共服务平台，为教师教学和学生学习大数据课程提供全方位、一站式服务，年访问量超过100万次。





附录C：《大数据技术原理与应用》教材

《大数据技术原理与应用——概念、存储、处理、分析与应用（第2版）》，由厦门大学计算机科学系林子雨博士编著，是国内高校第一本系统介绍大数据知识的专业教材。人民邮电出版社 ISBN:978-7-115-44330-4 定价：49.80元



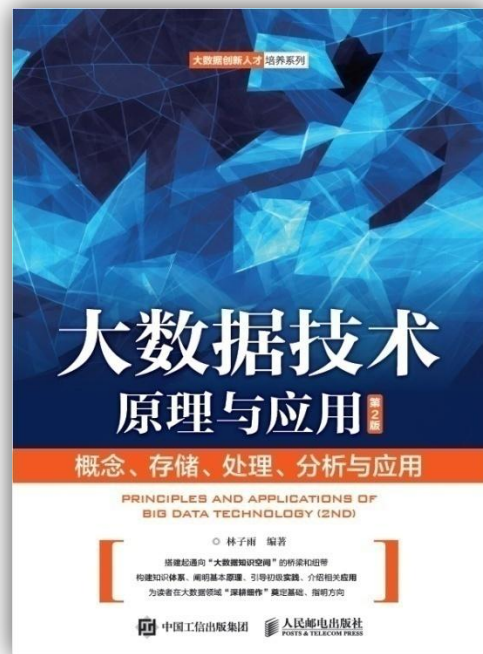
扫一扫访问教材官网

全书共有15章，系统地论述了大数据的基本概念、大数据处理架构Hadoop、分布式文件系统HDFS、分布式数据库HBase、NoSQL数据库、云数据库、分布式并行编程模型MapReduce、Spark、流计算、图计算、数据可视化以及大数据在互联网、生物医学和物流等各个领域的应用。在Hadoop、HDFS、HBase和MapReduce等重要章节，安排了入门级的实践操作，让读者更好地学习和掌握大数据关键技术。

本书可以作为高等院校计算机专业、信息管理等相关专业的大数据课程教材，也可供相关技术人员参考、学习、培训之用。

欢迎访问《大数据技术原理与应用——概念、存储、处理、分析与应用》教材官方网站：

<http://dbllab.xmu.edu.cn/post/bigdata>





附录D：《大数据基础编程、实验和案例教程》

本书是与《大数据技术原理与应用（第2版）》教材配套的唯一指定实验指导书

大数据教材



1+1黄金组合
厦门大学林子雨编著

配套实验指导书

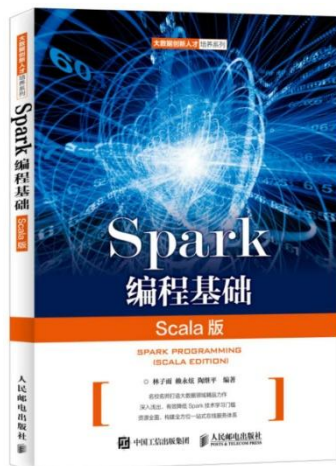


- 步步引导，循序渐进，详尽的安装指南为顺利搭建大数据实验环境铺平道路
- 深入浅出，去粗取精，丰富的代码实例帮助快速掌握大数据基础编程方法
- 精心设计，巧妙融合，五套大数据实验题目促进理论与编程知识的消化和吸收
- 结合理论，联系实际，大数据课程综合实验案例精彩呈现大数据分析全流程

清华大学出版社 ISBN:978-7-302-47209-4 定价：59元



附录E：《Spark编程基础（Scala版）》



《Spark编程基础（Scala版）》

厦门大学 林子雨，赖永炫，陶继平 编著

披荆斩棘，在大数据丛林中开辟学习捷径
填沟削坎，为快速学习Spark技术铺平道路
深入浅出，有效降低Spark技术学习门槛
资源全面，构建全方位一站式在线服务体系

人民邮电出版社出版发行，ISBN:978-7-115-48816-9
教材官网：<http://dblab.xmu.edu.cn/post/spark/>



本书以Scala作为开发Spark应用程序的编程语言，系统介绍了Spark编程的基础知识。全书共8章，内容包括大数据技术概述、Scala语言基础、Spark的设计与运行原理、Spark环境搭建和使用方法、RDD编程、Spark SQL、Spark Streaming、Spark MLlib等。本书每个章节都安排了入门级的编程实践操作，以便读者更好地学习和掌握Spark编程方法。本书官网免费提供了全套的在线教学资源，包括讲义PPT、习题、源代码、软件、数据集、授课视频、上机实验指南等。



附录F：高校大数据课程公共服务平台



高校大数据课程

公 共 服 务 平 台

<http://dblab.xmu.edu.cn/post/bigdata-teaching-platform/>



扫一扫访问平台主页



扫一扫观看3分钟FLASH动画宣传片



附录G：高校大数据实训课程系列案例教材

为了更好地满足高校开设大数据实训课程的教材需求，厦门大学数据库实验室林子雨老师团队联合企业共同开发了《高校大数据实训课程系列案例》，目前已经完成开发的系列案例包括：

《基于协同过滤算法的电影推荐》

《电信用户行为分析》

《实时日志流处理分析》

《微博用户情感分析》

《互联网广告预测分析》

《网站日志处理分析》

部分教材书稿已经完成写作，将于2019年陆续出版发行，教材相关信息，敬请关注网页后续更新！<http://dblab.xmu.edu.cn/post/shixunkecheng/>



扫一扫访问大数据实训课程系列案例教材主页

The background of the slide features several faint, light-blue silhouettes of people. In the top left, a group of three people is walking. In the top center, a group of seven people is standing in a line. In the bottom left, a person is sitting and talking on a mobile phone. On the right side, a person is standing and talking on a mobile phone. The text "Thank You!" is centered in the middle of the slide in a large, bold, white font.

Thank You!

Department of Computer Science, Xiamen University, 2018