

# 《大数据技术原理与应用》

<http://www.icourse163.org/course/XMU-1002335004>

中国大学MOOC 2018年春季学期

## 第9讲 Hadoop再探讨

林子雨

厦门大学计算机科学系

E-mail: [ziyulin@xmu.edu.cn](mailto:ziyulin@xmu.edu.cn)



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





中国大学MOOC《大数据技术原理与应用》课程地址：  
<http://www.icourse163.org/course/XMU-1002335004>

 中国大学MOOC

课程 名校 学·问 学校云 考研 新

客户端

搜索感兴趣的课程

登录 | 注册

 **廈門大學**  
XIAMEN UNIVERSITY

### 大数据技术原理与应用

入门级大数据精品课程，适合初学者，完备的课程在线服务体系，可以帮助初学者实现“零基础”学习大数据。课程指导思想是“构建知识体系、阐明基本原理、引导初级实践、了解相关应用”。课程由国内高校知名大数据教师厦门大学林子雨老师主讲。配套的《大数据技术原理与应用》教材已经被众多高校采用。

## 大数据技术原理与应用

BIGDATA TECHNOLOGY APPLICATION

打开大数据之门，遨游大数据世界

播放视频简介



# 课程重要资料



## 大数据软件安装和编程指南

读者在学习《大数据技术原理与应用》MOOC课程时，需要进行安装Linux系统和各种大数据软件，并开展基础编程实践。这个实践过程，如果没有配套的指南，将会耗费读者大量的时间，而且在实践过程中的大量障碍，网络上都没有现成的答案，会给读者带来很大的挫折感，感觉学习大数据是一件“痛苦的事情”。为了帮助读者实现“零基础”学习大数据并顺利完成实验环境搭建和开展基础编程，课程团队建设与本课程配套的《大数据软件安装和基础编程实践指南》，读者即使没有Linux系统知识，没有学习过任何大数据软件的使用，也可以在自己熟悉的Windows操作系统上顺利安装Linux虚拟机和各种大数据软件，顺利完成基础的编程实践，让学习入门级大数据技术变得“相对容易”。课程团队在过去5年时间里建设的在线免费资源，大大降低了大数据技术学习门槛，已经较好地帮助很多大数据初学者顺利完成了大数据基础实践，获得了读者的好评，目前，厦门大学数据库实验室网站上的这些在线免费大数据教学资源每年访问量超过100万次，在国内高校形成了广泛的影响力。

**重要提示：**读者在学习《大数据技术原理与应用》MOOC课程时，在中国大学MOOC课程的栏目中，有一个名称为“大数据软件安装和编程指南”的子栏目，进入这个栏目，可以帮助读者顺利完成大数据上机环境的安装和开展基础编程实践。在观看每个章节的MOOC视频时，可以充分利用该栏目辅助自己完成上机实验操作。



# 欢迎访问教材官网获取教学资源

## 《大数据技术原理与应用——大数据概念、存储、处理、分析与应用》

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

厦门大学 林子雨编著, 人民邮电出版社, 2017年1月第2版  
ISBN:978-7-115-44330-4

- 国内高校第一本系统介绍大数据知识专业教材
- 京东、当当等各大网店畅销书籍
- 大数据入门教材精品
- 国内多所高校采用本教材开课
- 配套目前国内高校最完备的课程公共服务平台
- 福建省精品在线开放课程





# 提纲

- 9.1 Hadoop的优化与发展
- 9.2 HDFS2.0的新特性
- 9.3 新一代资源管理调度框架YARN
- 9.4 Hadoop生态系统中具有代表性的功能组件

本PPT是如下教材的配套讲义：

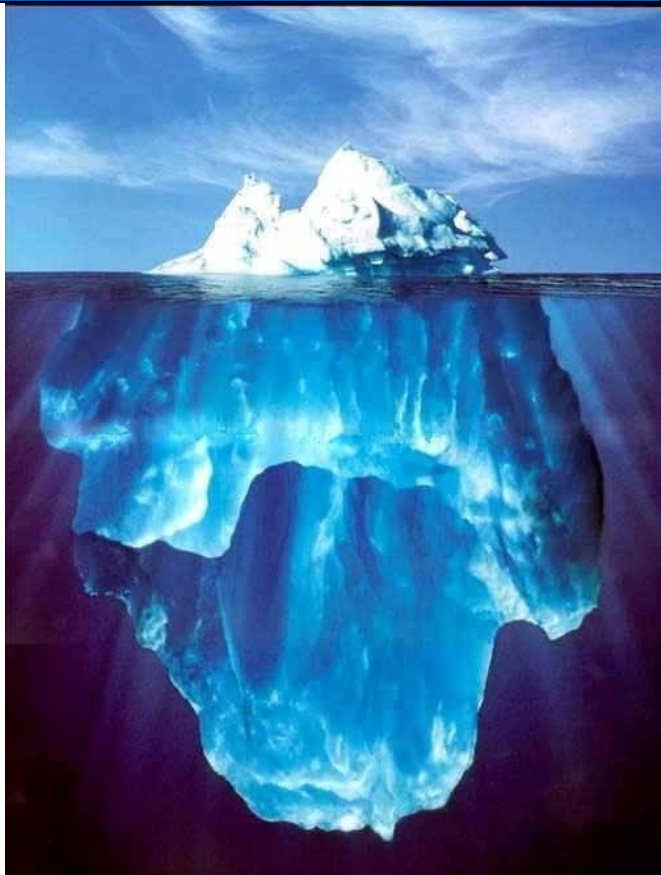
《大数据技术原理与应用——概念、存储、处理、分析与应用》  
(2017年1月第2版)

厦门大学 林子雨 编著，人民邮电出版社

ISBN:978-7-115-44330-4

欢迎访问《大数据技术原理与应用》教材官方网站，免费  
获取教材配套资源：

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





# 9.1 Hadoop的优化与发展

## 9.1.1 Hadoop的局限与不足

## 9.1.2 针对Hadoop的改进与提升



## 9.1.1 Hadoop的局限与不足

Hadoop1.0的核心组件（仅指MapReduce和HDFS，不包括Hadoop生态系统内的Pig、Hive、HBase等其他组件），主要存在以下不足：

- 抽象层次低，需人工编码
- 表达能力有限
- 开发者自己管理作业（Job）之间的依赖关系
- 难以看到程序整体逻辑
- 执行迭代操作效率低
- 资源浪费（Map和Reduce分两阶段执行）
- 实时性差（适合批处理，不支持实时交互式）



## 9.1.2针对Hadoop的改进与提升

Hadoop的优化与发展主要体现在两个方面：

- 一方面是Hadoop自身两大核心组件MapReduce和HDFS的架构设计改进
- 另一方面是Hadoop生态系统其它组件的不断丰富，加入了Pig、Tez、Spark和Kafka等新组件





## 9.1.2针对Hadoop的改进与提升

表9-1 Hadoop框架自身的改进：从1.0到2.0

组件	Hadoop1.0的问题	Hadoop2.0的改进
HDFS	单一名称节点，存在单点失效问题	设计了HDFS HA，提供名称节点热备机制
HDFS	单一命名空间，无法实现资源隔离	设计了HDFS Federation，管理多个命名空间
MapReduce	资源管理效率低	设计了新的资源管理框架YARN



## 9.1.2针对Hadoop的改进与提升

表9-2 不断完善的Hadoop生态系统

组件	功能	解决Hadoop中存在的问题
Pig	处理大规模数据的脚本语言，用户只需要编写几条简单的语句，系统会自动转换为MapReduce作业	抽象层次低，需要手工编写大量代码
Spark	基于内存的分布式并行编程框架，具有较高的实时性，并且较好支持迭代计算	延迟高，而且不适合执行迭代计算
Oozie	工作流和协作服务引擎，协调Hadoop上运行的不同任务	没有提供作业（Job）之间依赖关系管理机制，需要用户自己处理作业之间依赖关系
Tez	支持DAG作业的计算框架，对作业的操作进行重新分解和组合，形成一个大的DAG作业，减少不必要操作	不同的MapReduce任务之间存在重复操作，降低了效率
Kafka	分布式发布订阅消息系统，一般作为企业大数据分析平台的数据交换枢纽，不同类型的分布式系统可以统一接入到Kafka，实现和Hadoop各个组件之间的不同类型数据的实时高效交换	Hadoop生态系统中各个组件和其他产品之间缺乏统一的、高效的数据交换中介



## 9.2HDFS2.0的新特性

### 9.2.1HDFS HA

### 9.2.2HDFS Federation



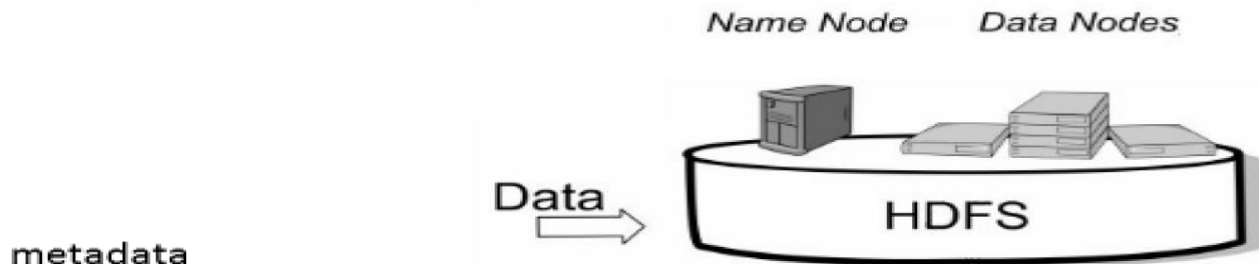
## 9.2.1 HDFS HA

### HDFS1.0组件及其功能回顾（具体请参见第3章HDFS）

名称节点保存元数据：

(1) 在磁盘上：FsImage和EditLog

(2) 在内存中：映射信息，即文件包含哪些块，每个块存储在哪个数据节点



#### metadata

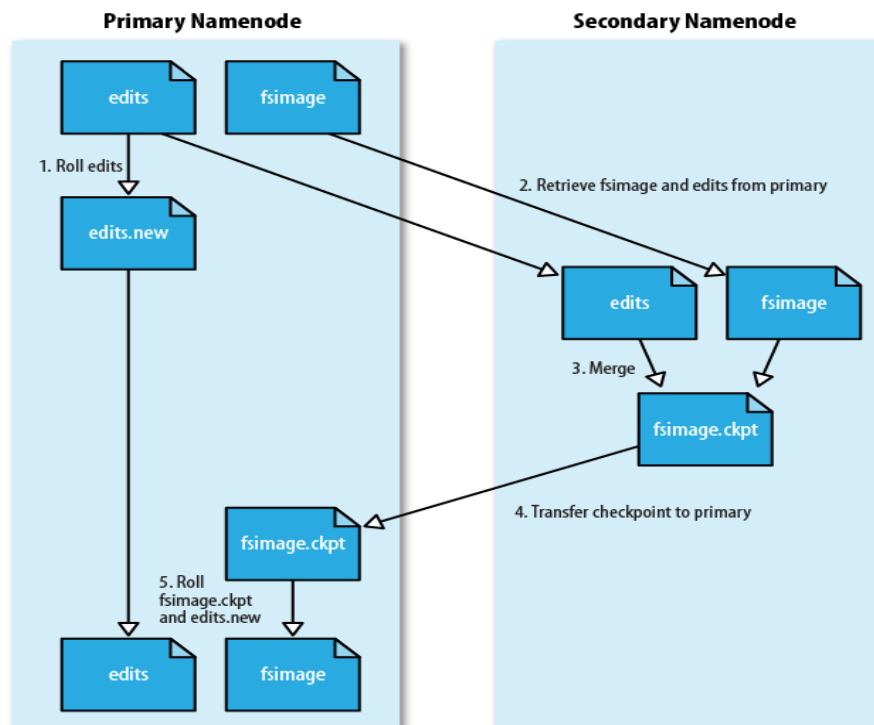
File.txt=  
Blk A:  
DN1, DN5, DN6  
  
Blk B:  
DN7, DN1, DN2  
  
Blk C:  
DN5, DN8, DN9

NameNode	DataNode
• 存储元数据	• 存储文件内容
• 元数据保存在内存中	• 文件内容保存在磁盘
• 保存文件,block , datanode 之间的映射关系	• 维护了block id到datanode本地文件的映射关系



## 9.2.1 HDFS HA

- HDFS 1.0存在单点故障问题
- 第二名称节点（SecondaryNameNode）无法解决单点故障问题



- SecondaryNameNode会定期和NameNode通信
  - 从NameNode上获取到FsImage和EditLog文件，并下载到本地的相应目录下
  - 执行EditLog和FsImage文件合并
  - 将新的FsImage文件发送到NameNode节点上
  - NameNode使用新的FsImage和EditLog（缩小了）
- 第二名称节点用途：
- 不是热备份
  - 主要是防止日志文件EditLog过大，导致名称节点失败恢复时消耗过多时间
  - 附带起到冷备份功能



## 9.2.1 HDFS HA

- HDFS HA（High Availability）是为了解决单点故障问题
- HA集群设置两个名称节点，“活跃（Active）”和“待命（Standby）”
- 两种名称节点的状态同步，可以借助于一个共享存储系统来实现
- 一旦活跃名称节点出现故障，就可以立即切换到待命名称节点
- Zookeeper确保一个名称节点在对外服务
- 名称节点维护映射信息，数据节点同时向两个名称节点汇报信息

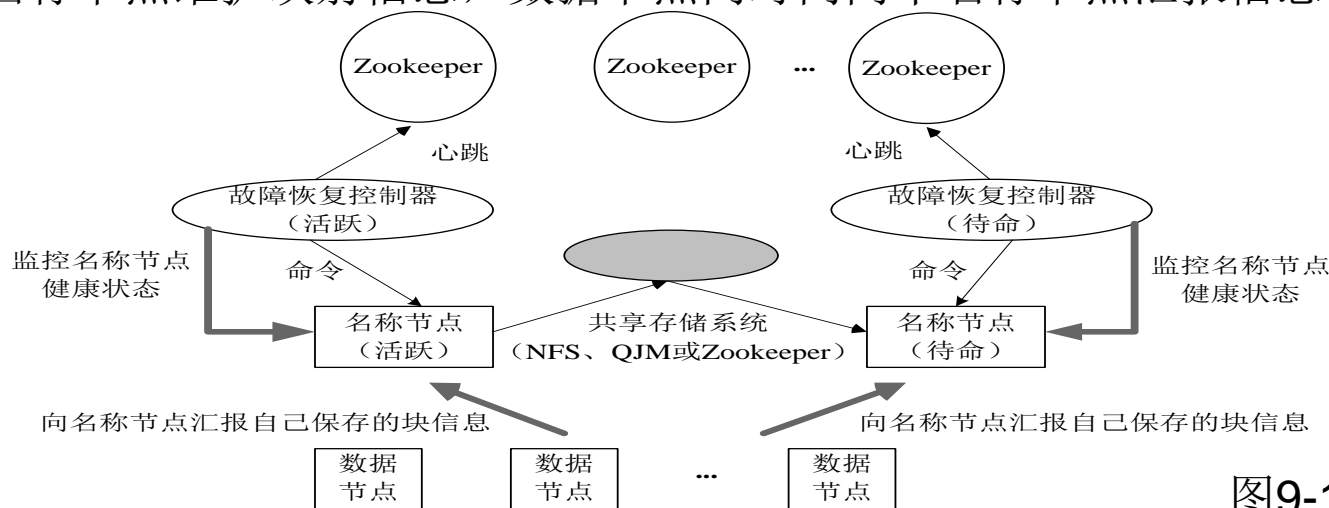


图9-1 HDFS HA架构



## 9.2.2 HDFS Federation

### 1. HDFS 1.0中存在的问题

- 单点故障问题
- 不可以水平扩展（是否可以通过纵向扩展来解决？）
- 系统整体性能受限于单个名称节点的吞吐量
- 单个名称节点难以提供不同程序之间的隔离性
- HDFS HA是热备份，提供高可用性，但是无法解决可扩展性、系统性能和隔离性

### 2. HDFS Federation的设计

• 在HDFS Federation中，设计了多个相互独立的名称节点，使得HDFS的命名服务能够水平扩展，这些名称节点分别进行各自命名空间和块的管理，相互之间是联盟（Federation）关系，不需要彼此协调。并且向后兼容

• HDFS Federation中，所有名称节点会共享底层的数据节点存储资源，数据节点向所有名称节点汇报

• 属于同一个命名空间的块构成一个“块池”

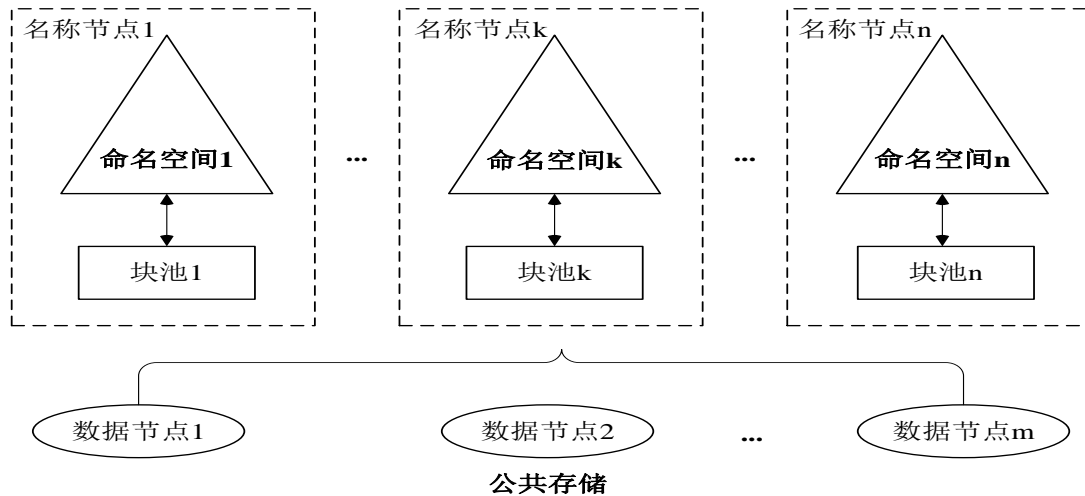


图9-2 HDFS Federation架构



## 9.2.2 HDFS Federation

### 3. HDFS Federation的访问方式

- 对于Federation中的多个命名空间，可以采用客户端挂载表（Client Side Mount Table）方式进行数据共享和访问
- 客户可以访问不同的挂载点来访问不同的子命名空间
- 把各个命名空间挂载到全局“挂载表”（mount-table）中，实现数据全局共享
- 同样的命名空间挂载到个人的挂载表中，就成为应用程序可见的命名空间

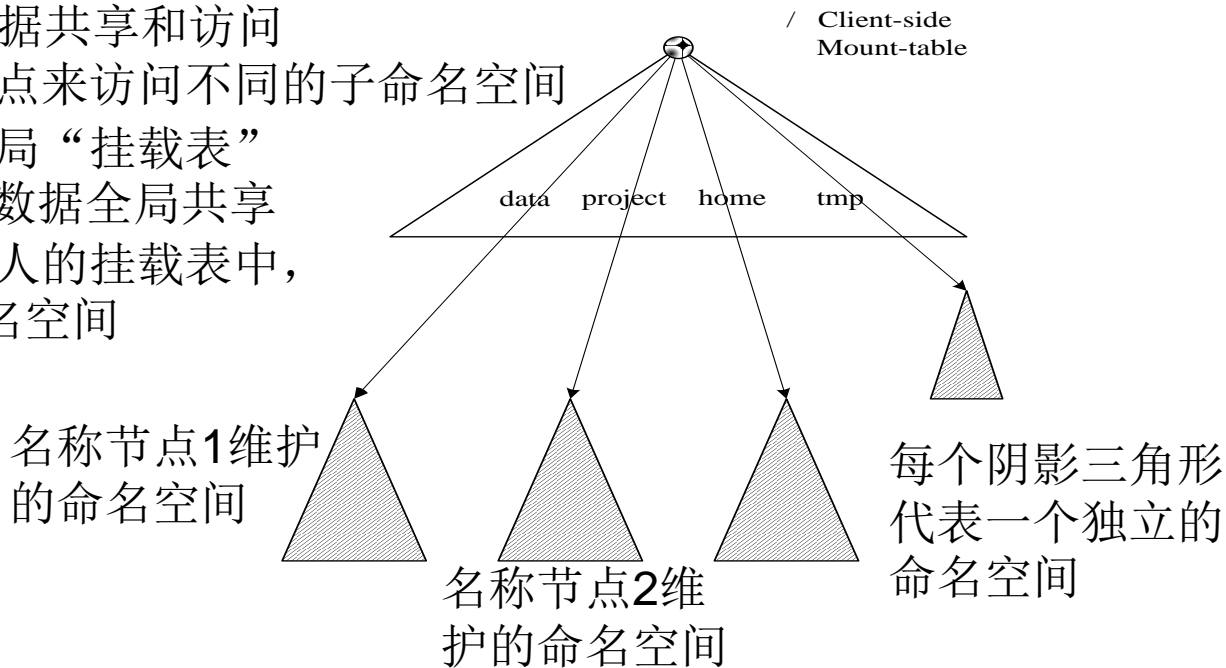


图9-3 客户端挂载表方式访问多个命名空间





## 9.2.2HDFS Federation

### 4.HDFS Federation相对于HDFS1.0的优势

HDFS Federation设计可解决单名称节点存在的以下几个问题:

(1) **HDFS集群扩展性**。多个名称节点各自分管一部分目录,使得一个集群可以扩展到更多节点,不再像HDFS1.0中那样由于内存的限制制约文件存储数目

(2) **性能更高效**。多个名称节点管理不同的数据,且同时对外提供服务,将为用户提供更高的读写吞吐率

(3) **良好的隔离性**。用户可根据需要将不同业务数据交由不同名称节点管理,这样不同业务之间影响很小

需要注意的, HDFS Federation并不能解决单点故障问题,也就是说,每个名称节点都存在在单点故障问题,需要为每个名称节点部署一个后备名称节点,以应对名称节点挂掉对业务产生的影响



## 9.3新一代资源管理调度框架YARN

9.3.1 MapReduce1.0的缺陷

9.3.2 YARN设计思路

9.3.3 YARN体系结构

9.3.4 YARN工作流程

9.3.5 YARN框架与MapReduce1.0框架的对比分析

9.3.6 YARN的发展目标



## 9.3.1 MapReduce1.0的缺陷

- (1) 存在单点故障
- (2) **JobTracker** “大包大揽” 导致任务过重（任务多时内存开销大，上限4000节点）
- (3) 容易出现内存溢出（分配资源只考虑MapReduce任务数，不考虑CPU、内存）
- (4) 资源划分不合理（强制划分为slot，包括Map slot和Reduce slot）

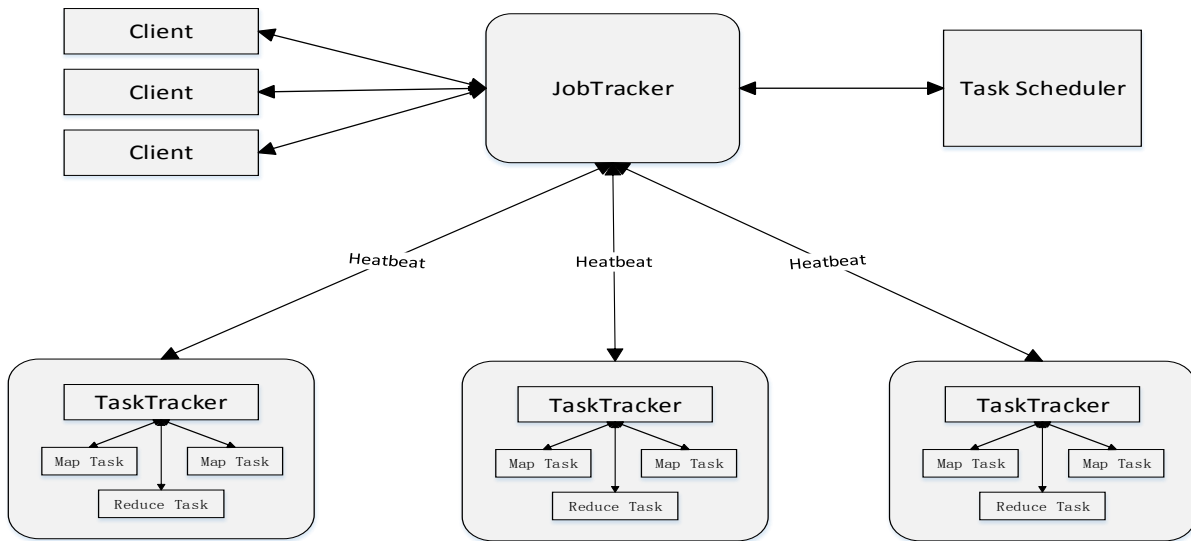
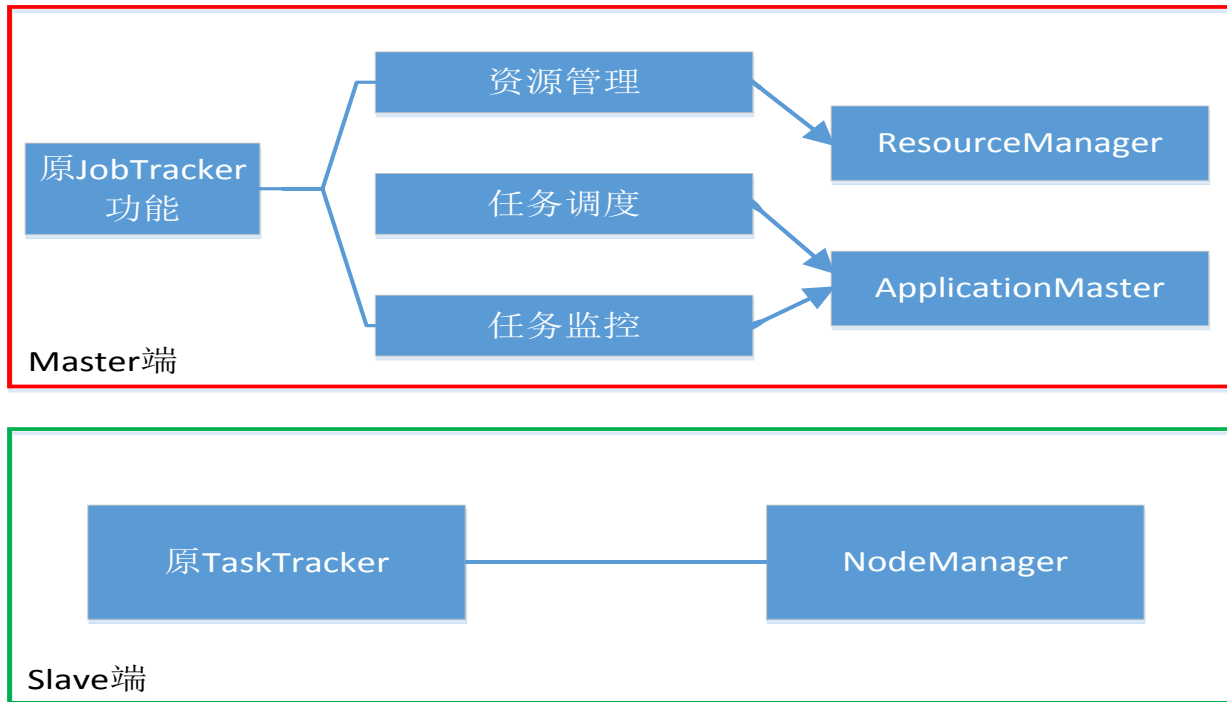


图9-4 MapReduce1.0体系结构



## 9.3.2 YARN设计思路

YARN架构思路：将原JobTracker三大功能拆分



• MapReduce 1.0 既是一个计算框架，也是一个资源管理调度框架

• 到了Hadoop 2.0以后，MapReduce 1.0中的资源管理调度功能，被单独分离出来形成了YARN，它是一个纯粹的资源管理调度框架，而不是一个计算框架

• 被剥离了资源管理调度功能的MapReduce 框架就变成了MapReduce 2.0，它是运行在YARN之上的一个纯粹的计算框架，不再自己负责资源调度管理服务，而是由YARN为其提供资源管理调度服务



## 9.3.3 YARN体系结构

### ResourceManager

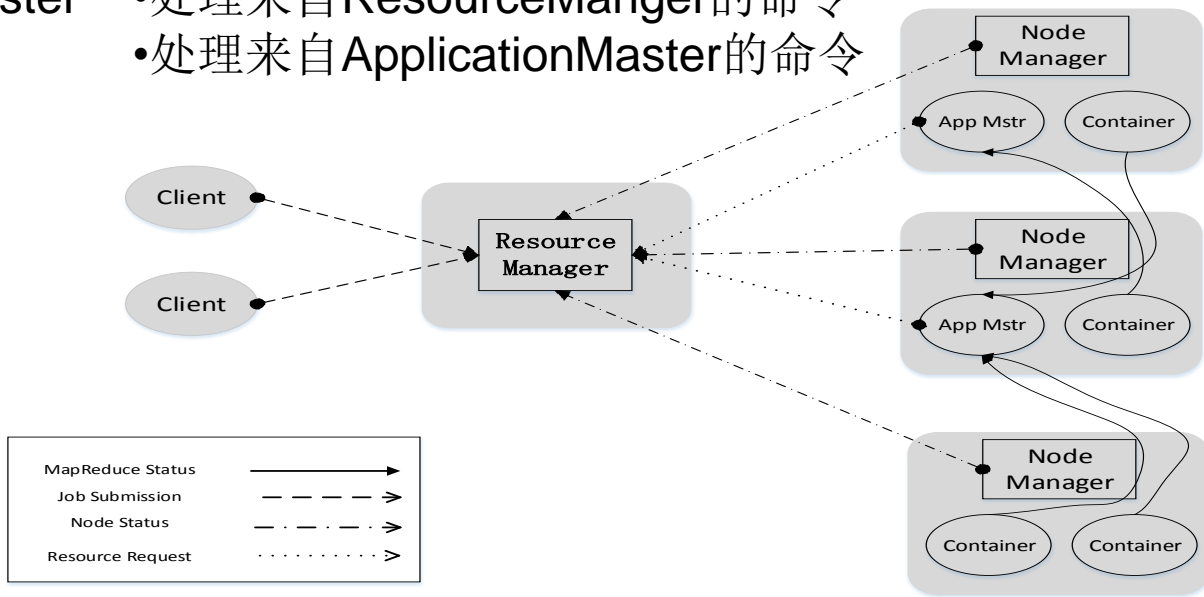
- 处理客户端请求
- 启动/监控ApplicationMaster
- 监控NodeManager
- 资源分配与调度

### ApplicationMaster

- 为应用程序申请资源，并分配给内部任务
- 任务调度、监控与容错

### NodeManager

- 单个节点上的资源管理
- 处理来自ResourceManager的命令
- 处理来自ApplicationMaster的命令





## 9.3.3 YARN体系结构

### ResourceManager

- **ResourceManager (RM)** 是一个全局的资源管理器，负责整个系统的资源管理和分配，主要包括两个组件，即调度器 (**Scheduler**) 和应用程序管理器 (**Applications Manager**)
- 调度器接收来自 **ApplicationMaster** 的应用程序资源请求，把集群中的资源以“容器”的形式分配给提出申请的应用程序，容器的选择通常会考虑应用程序所要处理的数据的位置，进行就近选择，从而实现“计算向数据靠拢”
- 容器 (**Container**) 作为动态资源分配单位，每个容器中都封装了一定数量的 **CPU**、内存、磁盘等资源，从而限定每个应用程序可以使用的资源量
- 调度器被设计成是一个可插拔的组件，**YARN** 不仅自身提供了许多种直接可用的调度器，也允许用户根据自己的需求重新设计调度器
- 应用程序管理器 (**Applications Manager**) 负责系统中所有应用程序的管理工作，主要包括应用程序提交、与调度器协商资源以启动 **ApplicationMaster**、监控 **ApplicationMaster** 运行状态并在失败时重新启动等



## 9.3.3 YARN体系结构

### ApplicationMaster

ResourceManager接收用户提交的作业，按照作业的上下文信息以及从NodeManager收集来的容器状态信息，启动调度过程，为用户作业启动一个ApplicationMaster

ApplicationMaster的主要功能是：

- (1) 当用户作业提交时，ApplicationMaster与ResourceManager协商获取资源，ResourceManager会以容器的形式为ApplicationMaster分配资源；
- (2) 把获得的资源进一步分配给内部的各个任务（Map任务或Reduce任务），实现资源的“二次分配”；
- (3) 与NodeManager保持交互通信进行应用程序的启动、运行、监控和停止，监控申请到的资源的使用情况，对所有任务的执行进度和状态进行监控，并在任务发生失败时执行失败恢复（即重新申请资源重启任务）；
- (4) 定时向ResourceManager发送“心跳”消息，报告资源的使用情况和应用的进度信息；
- (5) 当作业完成时，ApplicationMaster向ResourceManager注销容器，执行周期完成。



## 9.3.3 YARN体系结构

### NodeManager

NodeManager是驻留在一个YARN集群中的每个节点上的代理，主要负责：

- 容器生命周期管理
- 监控每个容器的资源（CPU、内存等）使用情况
- 跟踪节点健康状况
- 以“心跳”的方式与ResourceManager保持通信
- 向ResourceManager汇报作业的资源使用情况和每个容器的运行状态
- 接收来自ApplicationMaster的启动/停止容器的各种请求

需要说明的是，NodeManager主要负责管理抽象的容器，只处理与容器相关的事情，而不具体负责每个任务（Map任务或Reduce任务）自身状态的管理，因为这些管理工作是由ApplicationMaster完成的，ApplicationMaster会通过不断与NodeManager通信来掌握各个任务的执行状态





## 9.3.3 YARN体系结构

在集群部署方面，YARN的各个组件是和Hadoop集群中的其他组件进行统一部署的

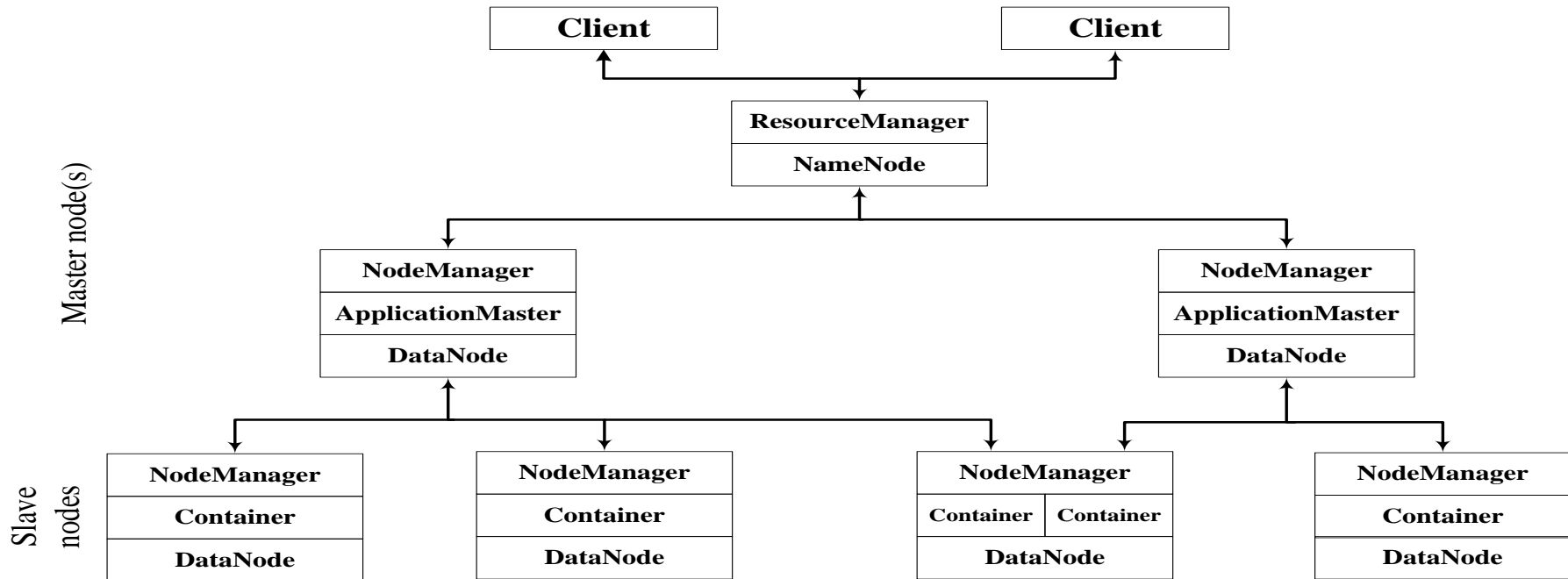
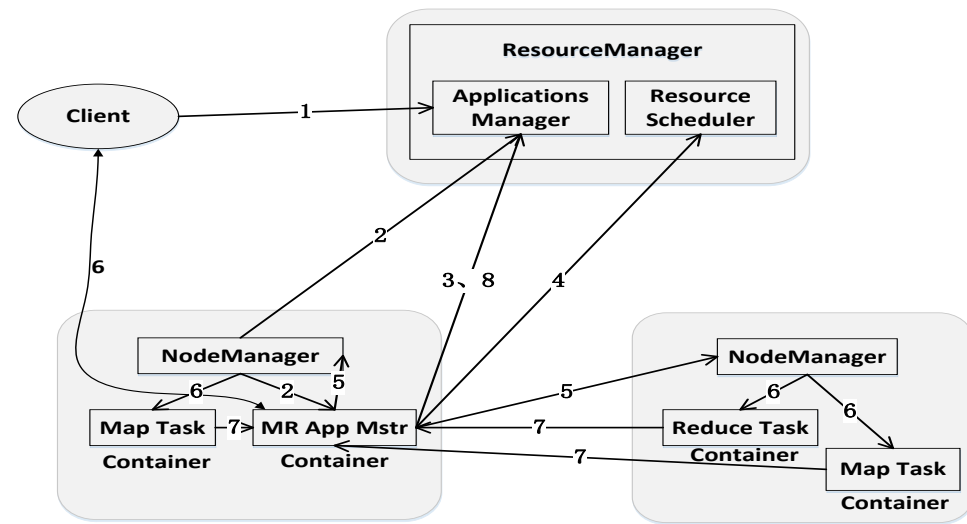


图9-7 YARN和Hadoop平台其他组件的统一部署



步骤2: YARN中的ResourceManager负责接收和处理来自客户端的请求, 为应用程序分配一个容器, 在该容器中启动一个ApplicationMaster



**步骤8:** 应用程序运行完成后，ApplicationMaster向ResourceManager的应用程序管理器注销并关闭自己



## 9.3.5 YARN框架与MapReduce1.0框架的对比分析

- 从MapReduce1.0框架发展到YARN框架，客户端并没有发生变化，其大部分调用API及接口都保持兼容，因此，原来针对Hadoop1.0开发的代码不用做大的改动，就可以直接放到Hadoop2.0平台上运行

总体而言，YARN相对于MapReduce1.0来说具有以下优势：

- 大大减少了承担中心服务功能的ResourceManager的资源消耗
  - ApplicationMaster来完成需要大量资源消耗的任务调度和监控
  - 多个作业对应多个ApplicationMaster，实现了监控分布化
- MapReduce1.0既是一个计算框架，又是一个资源管理调度框架，但是，只能支持MapReduce编程模型。而YARN则是一个纯粹的资源调度管理框架，在它上面可以运行包括MapReduce在内的不同类型的计算框架，只要编程实现相应的ApplicationMaster
- YARN中的资源管理比MapReduce1.0更加高效
  - 以容器为单位，而不是以slot为单位



## 9.3.6 YARN的发展目标

### YARN的目标就是实现“一个集群多个框架”，为什么？

- 一个企业当中同时存在各种不同的业务应用场景，需要采用不同的计算框架
  - MapReduce实现离线批处理
  - 使用Impala实现实时交互式查询分析
  - 使用Storm实现流式数据实时分析
  - 使用Spark实现迭代计算
- 这些产品通常来自不同的开发团队，具有各自的资源调度管理机制
- 为了避免不同类型应用之间互相干扰，企业就需要把内部的服务器拆分成多个集群，分别安装运行不同的计算框架，即“一个框架一个集群”
- 导致问题
  - 集群资源利用率低
  - 数据无法共享
  - 维护代价高



## 9.3.6 YARN的发展目标

- YARN的目标就是实现“一个集群多个框架”，即在一个集群上部署一个统一的资源调度管理框架YARN，在YARN之上可以部署其他各种计算框架
- 由YARN为这些计算框架提供统一的资源调度管理服务，并且能够根据各种计算框架的负载需求，调整各自占用的资源，实现集群资源共享和资源弹性收缩
- 可以实现一个集群上的不同应用负载混搭，有效提高了集群的利用率
- 不同计算框架可以共享底层存储，避免了数据集跨集群移动

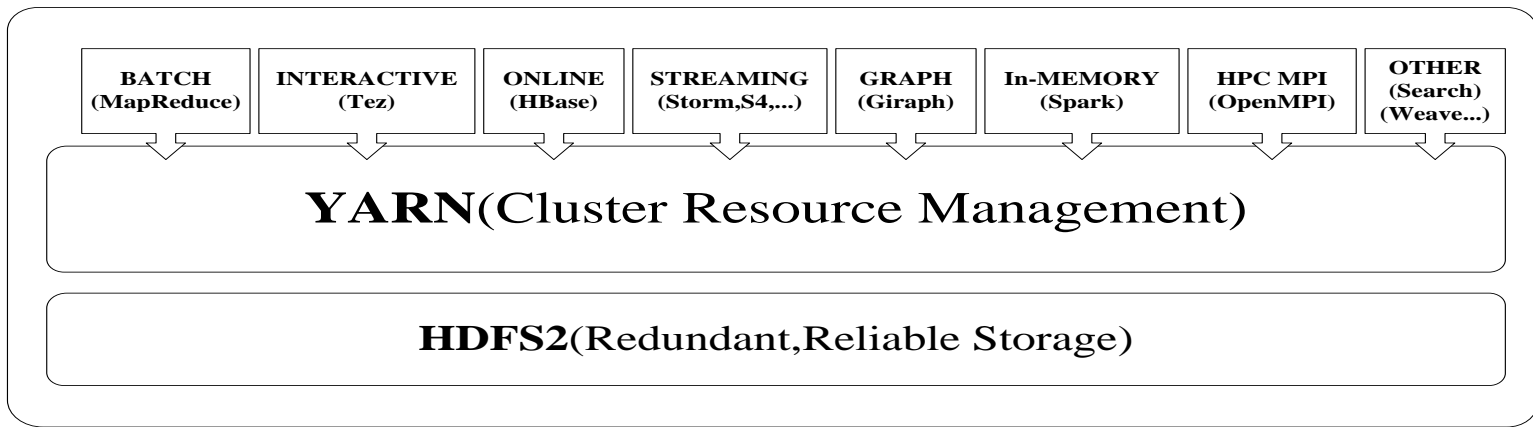


图9-9 在YARN上部署各种计算框架



## 9.4 Hadoop生态系统中具有代表性的功能组件

**9.4.1 Pig**

**9.4.2 Tez**

**9.4.3 Spark**

**9.4.4 Kafka**



## 9.4.1 Pig

- Pig是Hadoop生态系统的一个组件
- 提供了类似SQL的Pig Latin语言（包含Filter、GroupBy、Join、OrderBy等操作，同时也支持用户自定义函数）
- 允许用户通过编写简单的脚本来实现复杂的数据分析，而不需要编写复杂的MapReduce应用程序
- Pig会自动把用户编写的脚本转换成MapReduce作业在Hadoop集群上运行，而且具备对生成的MapReduce程序进行自动优化的功能
- 用户在编写Pig程序的时候，不需要关心程序的运行效率，这就大大减少了用户编程时间
- 通过配合使用Pig和Hadoop，在处理海量数据时就可以实现事半功倍的效果，比使用Java、C++等语言编写MapReduce程序的难度要小很多，并且用更少的代码量实现了相同的数据处理分析功能



## 9.4.1 Pig

Pig可以加载数据、表达转换数据以及存储最终结果

Pig语句通常按照如下的格式来编写:

- 通过LOAD语句从文件系统读取数据
- 通过一系列“转换”语句对数据进行处理
- 通过一条STORE语句把处理结果输出到文件系统中，或者使用DUMP语句把处理结果输出到屏幕上



数据收集



数据加工(Pig)



数据仓库(Hive)

图9-10 Pig在企业数据分析系统中的作用





## 9.4.1 Pig

- 下面是一个采用Pig Latin语言编写的应用程序实例，实现对用户访问网页情况的统计分析：

```
visits          = load '/data/visits' as (user, url, time);

gVisits         = group visits by url;
visitCounts    = foreach gVisits generate url, count(visits);
//得到的表的结构visitCounts(url,visits)
urlInfo        = load '/data/urlInfo' as (url, category, pRank);
visitCounts    = join visitCounts by url, urlInfo by url;
//得到的连接结果表的结构visitCounts(url,visits,category,pRank)
gCategories    = group visitCounts by category;
topUrls        = foreach gCategories generate top(visitCounts,10);

store topUrls into '/data/topUrls';
```



## 9.4.1 Pig

Pig Latin是通过编译为MapReduce在Hadoop集群上执行的。统计用户访问量程序被编译成MapReduce时，会产生如图所示的Map和Reduce

```
1 visits      = load '/data/visits' as (user, url, time);
2 gVisits     = group visits by url;
3 visitCounts = foreach gVisits generate url, count(visits);
//得到的表的结构visitCounts(url,visits)

4 urlInfo     = load '/data/urlInfo' as (url, category, pRank);
5 visitCounts = join visitCounts by url, urlInfo by url;
//得到的连接结果表的结构visitCounts(url,visits,category,pRank)

6 gCategories = group visitCounts by category;
7 topUrls    = foreach gCategories generate top(visits,10);
8 store topUrls into '/data/topUrls';
```

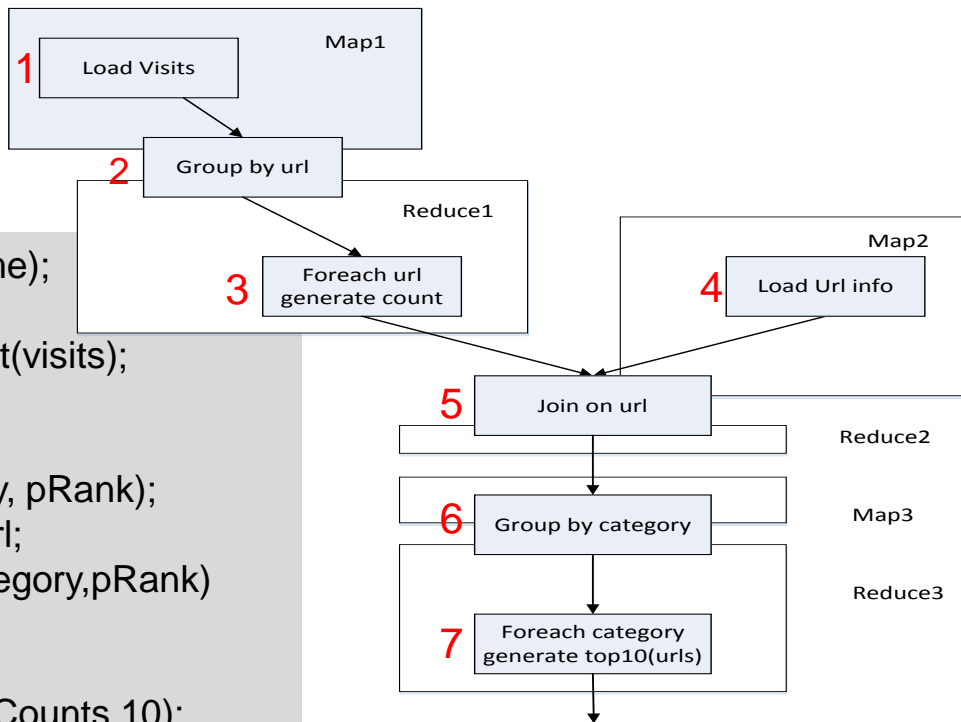
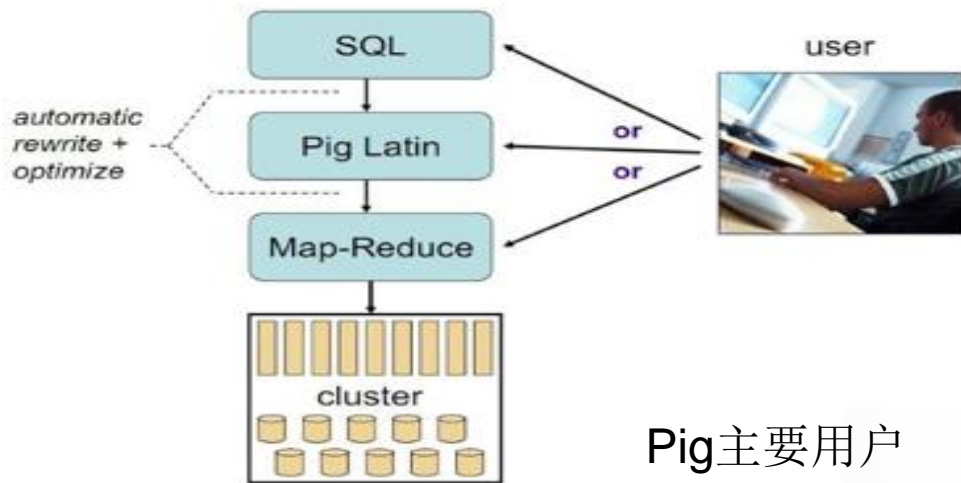


图9-11 从Pig Latin脚本转化得到的MapReduce作业



## 9.4.1 Pig



### Pig的应用场景

- 数据查询只面向相关技术人员
- 即时性的数据处理需求，这样可以通过pig很快写一个脚本开始运行处理，而不需要创建表等相关的事先准备工作

### Pig主要用户

- Yahoo!: 90%以上的MapReduce作业是Pig生成的
- Twitter: 80%以上的MapReduce作业是Pig生成的
- Linkedin: 大部分的MapReduce作业是Pig生成的
- 其他主要用户: Salesforce, Nokia, AOL, comScore



## 9.4.2 Tez

- Tez是Apache开源的支持DAG作业的计算框架，它直接源于MapReduce框架
- 核心思想是将Map和Reduce两个操作进一步拆分
- Map被拆分成Input、Processor、Sort、Merge和Output
- Reduce被拆分成Input、Shuffle、Sort、Merge、Processor和Output等
- 分解后的元操作可以任意灵活组合，产生新的操作
- 这些操作经过一些控制程序组装后，可形成一个大的DAG作业
- 通过DAG作业的方式运行MapReduce作业，提供了程序运行的整体处理逻辑，就可以去除工作流当中多余的Map阶段，减少不必要的操作，提升数据处理的性能
- Hortonworks把Tez应用到数据仓库Hive的优化中，使得性能提升了约100倍



## 9.4.2 Tez

```
SELECT a.state, COUNT(*), AVERAGE(c.price)
FROM a
JOIN b ON (a.id = b.id)
JOIN c ON (a.itemId = c.itemId)
GROUP BY a.state
```

- Tez的优化主要体现在:
- 去除了连续两个作业之间的“写入HDFS”
- 去除了每个工作流中多余的Map阶段

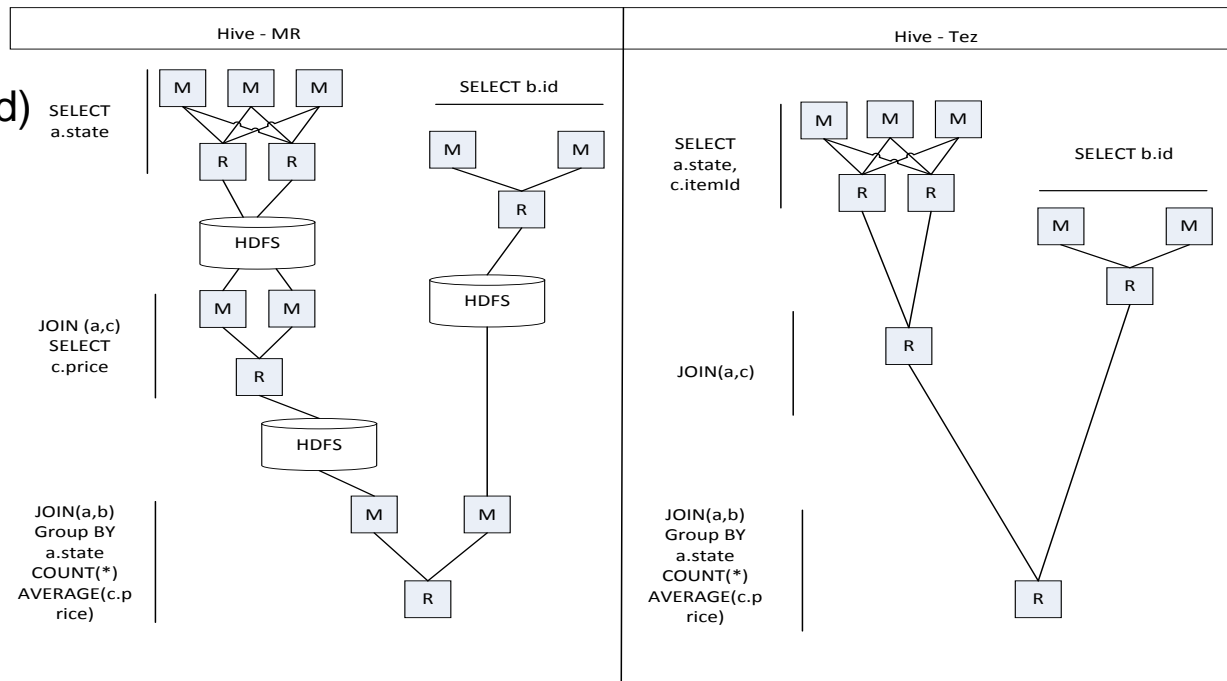


图9-12 HiveQL语句在MapReduce和Tez中的执行情况对比



## 9.4.2 Tez

- 在Hadoop2.0生态系统中，MapReduce、Hive、Pig等计算框架，都需要最终以MapReduce任务的形式执行数据分析，因此，Tez框架可以发挥重要的作用
- 借助于Tez框架实现对MapReduce、Pig和Hive等的性能优化
- 可以解决现有MR框架在迭代计算（如PageRank计算）和交互式计算方面的问题

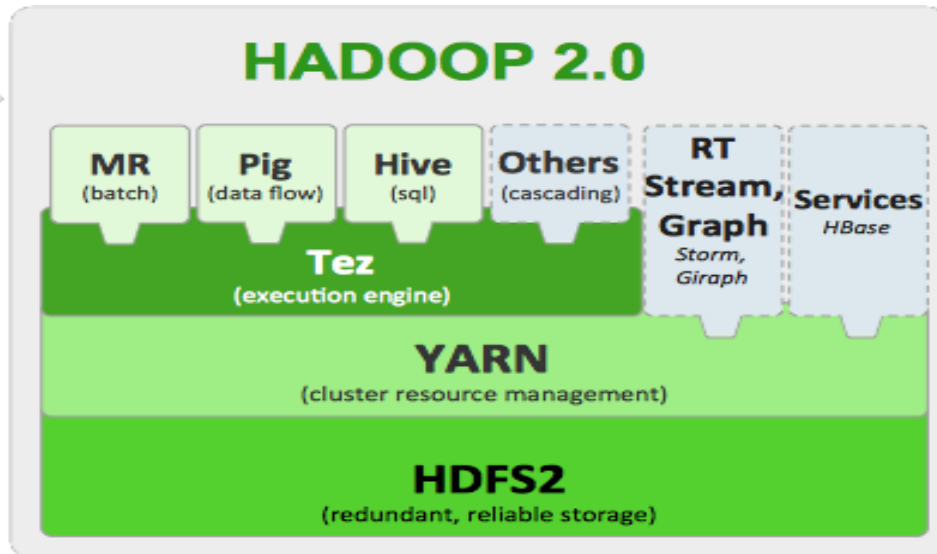


图9-13 Tez框架在Hadoop生态系统中的作用



## 9.4.2 Tez

### (Tez+Hive)与Impala、Dremel和Drill的区别？

- Tez在解决Hive、Pig延迟大、性能低等问题的思路，是和那些支持实时交互式查询分析的产品（如Impala、Dremel和Drill等）是不同的
- Impala、Dremel和Drill的解决问题思路是抛弃MapReduce计算框架，不再将类似SQL语句的HiveQL或者Pig语句翻译成MapReduce程序，而是采用与商用并行关系数据库类似的分布式查询引擎，可以直接从HDFS或者HBase中用SQL语句查询数据，而不需要把SQL语句转化成MapReduce任务来执行，从而大大降低了延迟，很好地满足了实时查询的要求
- Tez则不同，比如，针对Hive数据仓库进行优化的“Tez+Hive”解决方案，仍采用MapReduce计算框架，但是对DAG的作业依赖关系进行了裁剪，并将多个小作业合并成一个大作业，这样，不仅计算量减少了，而且写HDFS次数也会大大减少



## 9.4.3 Spark

- **Hadoop**缺陷，其**MapReduce**计算模型延迟过高，无法胜任实时、快速计算的需求，因而只适用于离线批处理的应用场景
  - 中间结果写入磁盘，每次运行都从磁盘读数据
  - 在前一个任务执行完成之前，其他任务无法开始，难以胜任复杂、多阶段的计算任务
- **Spark**最初诞生于伯克利大学的**APM**实验室，是一个可应用于大规模数据处理的快速、通用引擎，如今是**Apache**软件基金会下的顶级开源项目之一
- **Spark**在借鉴**Hadoop MapReduce**优点的同时，很好地解决了**MapReduce**所面临的问题
  - 内存计算，带来了更高的迭代运算效率
  - 基于**DAG**的任务调度执行机制，优于**MapReduce**的迭代执行机制
- 当前，**Spark**正以其结构一体化、功能多元化的优势，逐渐成为当今大数据领域最热门的大数据计算平台





## 9.4.4Kafka

- Kafka**是一种高吞吐量的分布式发布订阅消息系统，用户通过**Kafka**系统可以发布大量的消息，同时也能实时订阅消费消息
- Kafka**可以同时满足在线实时处理和批量离线处理
- 在公司的大数据生态系统中，可以把**Kafka**作为数据交换枢纽，不同类型的分布式系统（关系数据库、NoSQL数据库、流处理系统、批处理系统等），可以统一接入到**Kafka**，实现和**Hadoop**各个组件之间的不同类型数据的实时高效交换

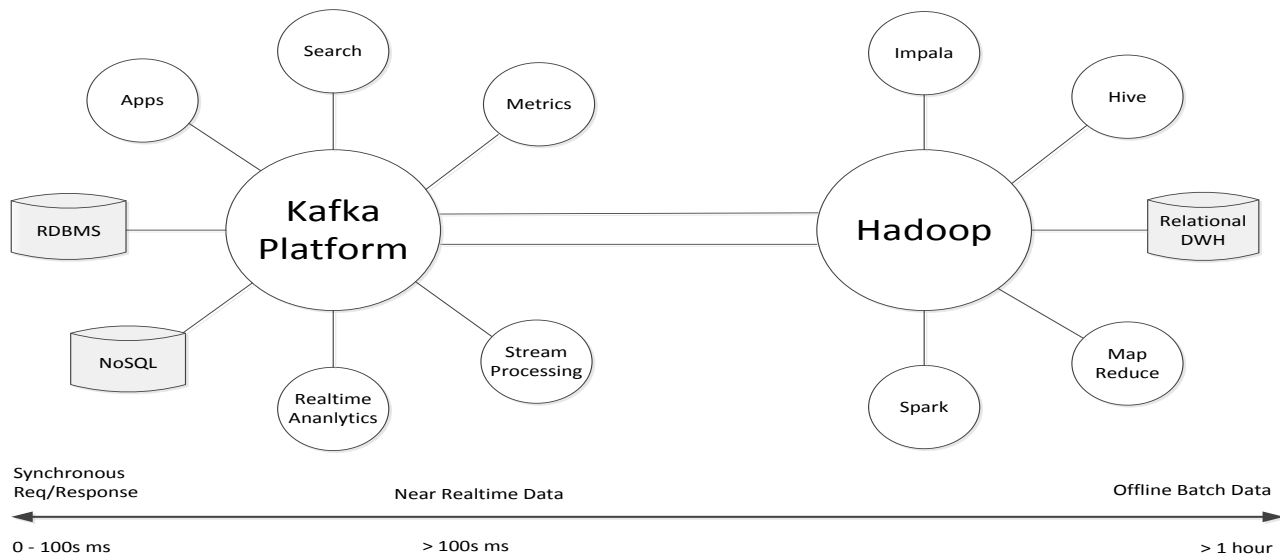


图9-14 Kafka作为数据交换枢纽



# 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](mailto: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、晋江市发展和改革局副局长。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。中国高校首个“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。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B：大数据学习路线图



大数据学习路线图访问地址：<http://dblab.xmu.edu.cn/post/10164/>



# 附录C：《大数据技术原理与应用》教材



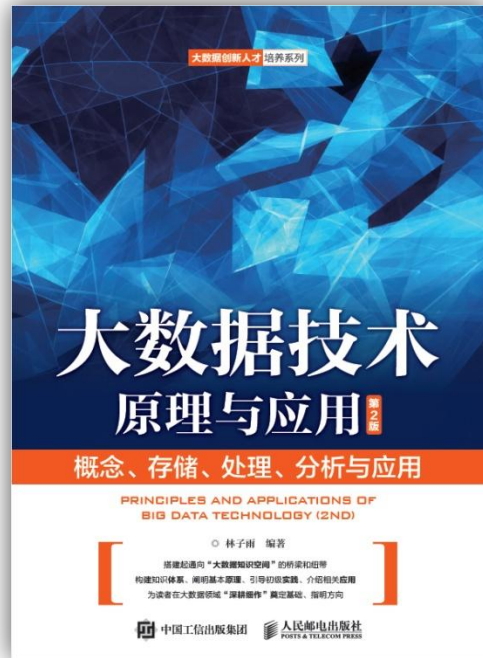
扫一扫访问教材官网

《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，是中国高校第一本系统介绍大数据知识的专业教材。

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





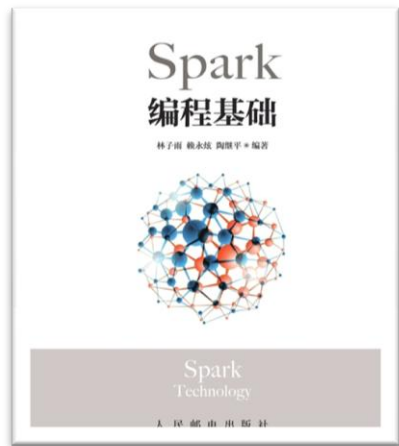
# 附录E：《Spark编程基础》教材

## 《Spark编程基础》

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:978-7-115-47598-5  
教材官网：<http://dblab.xmu.edu.cn/post/spark/>  
授课视频：<http://dblab.xmu.edu.cn/post/10482/>



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

国内高校首个大数据课程公共服务平台，为全国高校教师和学生提供大数据教学资源一站式“免费”在线服务，包括课程教材、讲义PPT、课程习题、实验指南、学习指南、备课指南、授课视频、技术资料、实验案例、在线教程等，目前平台每年访问量超过100万次，成为全国高校大数据教学知名品牌



## 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

The background is a solid blue color. It features several faint, light-blue silhouettes of people. In the top left, a group of people is holding hands in a circle. In the top center, a group of people is standing in a line, holding hands. In the bottom left, a person is sitting and looking towards the right. In the bottom right, a person is standing and looking towards the left. The text "Thank You!" is centered in the middle of the image.

# **Thank You!**

Department of Computer Science, Xiamen University, 2018