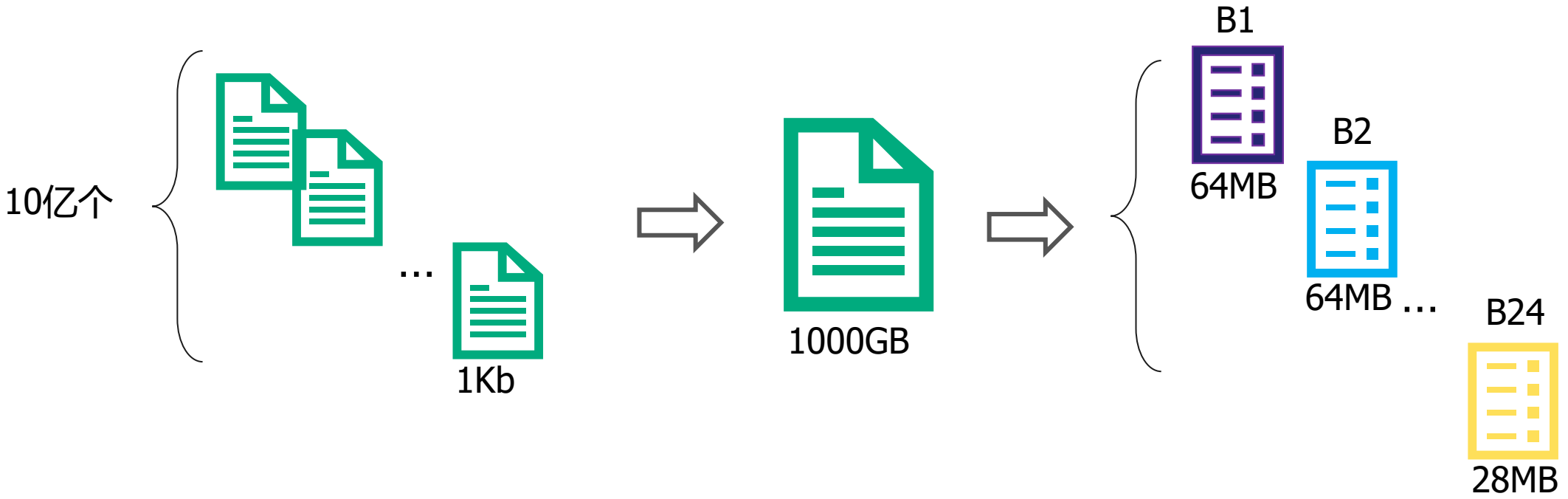
大数据分布式处理：Map-reduce计算模型

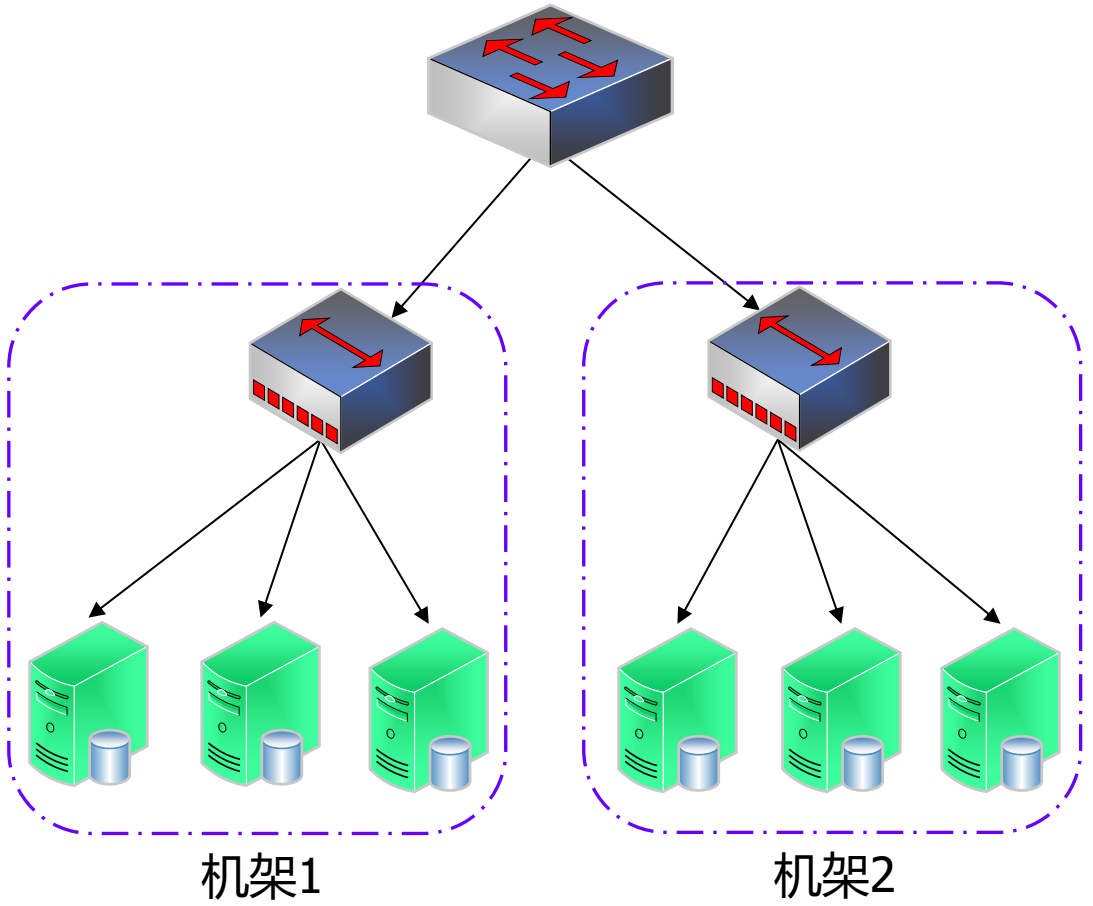
示例：海量网络评论的词频统计

- 用户需求：将10亿个网络评论存储到HDFS系统中：



回顾：大数据分布式处理的主要技术问题分析

- 计算任务（负载）如何分配?
- 分布式通信带来的额外开销问题?
  - 数据分发、处理结果收集
  - 大数据分布处理可带来的数据传输问题
- 硬件故障怎么办?
  - 节点故障、网络故障
- 系统的访问接口是否简单易用?
- 硬件性能需求及成本如何?

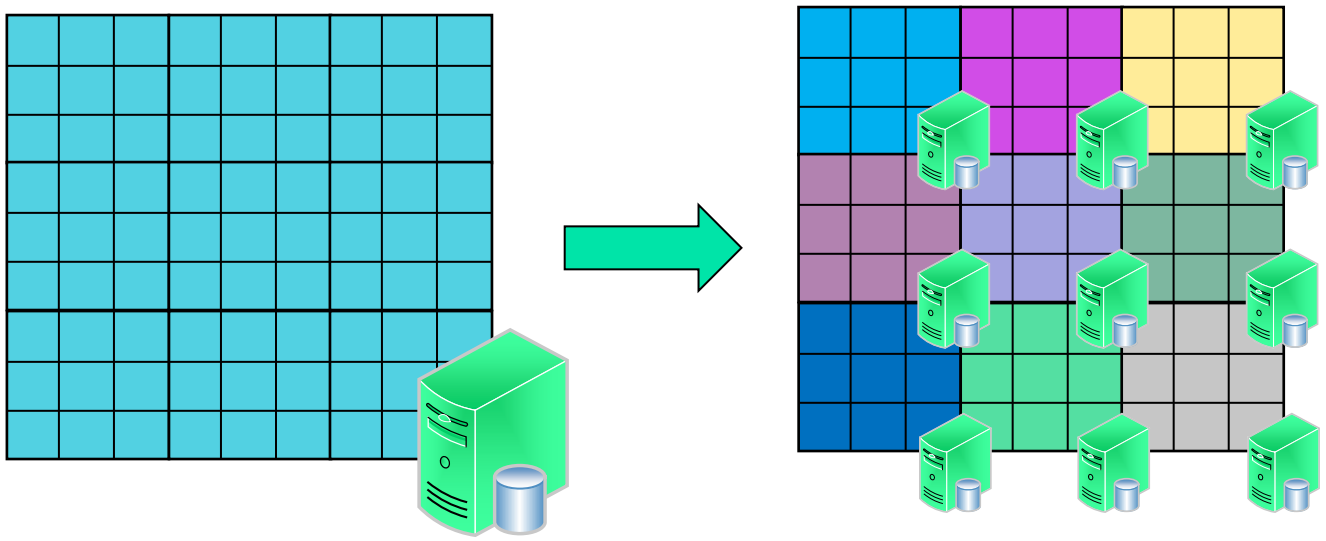


MapReduce简介

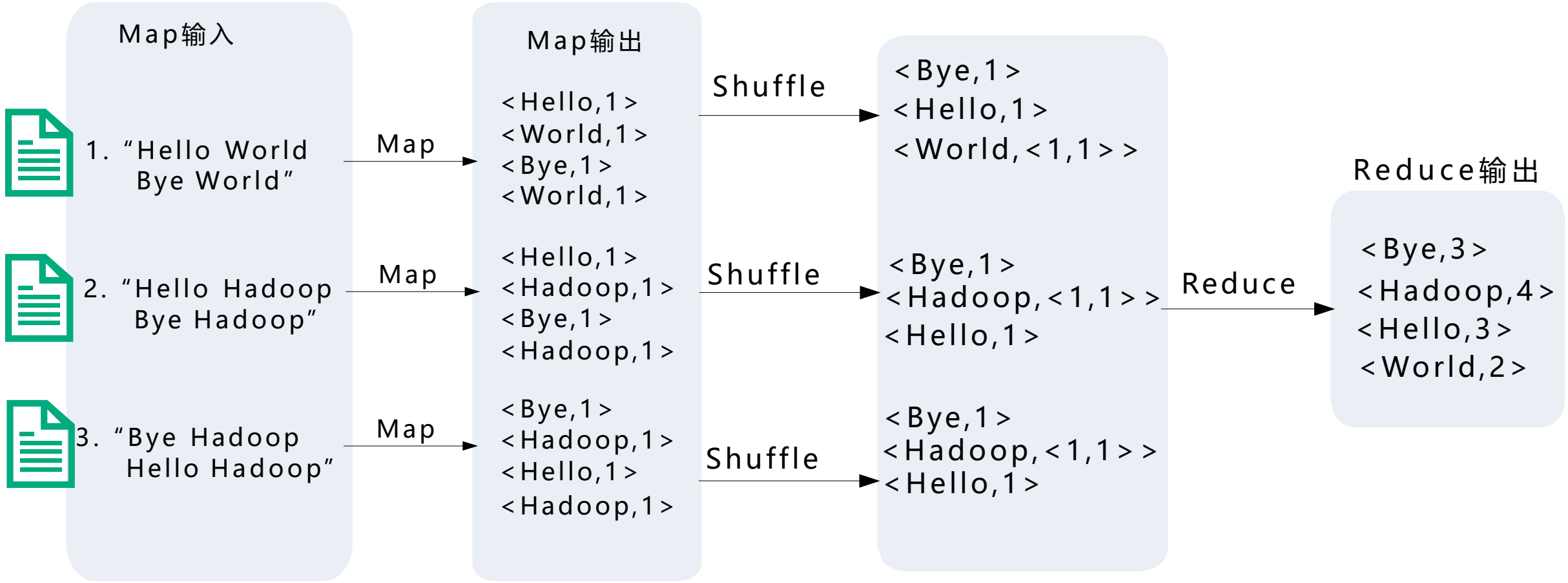
- MapReduce是一个统一的分布式并行计算软件框架，可以实现：
  - 计算任务的划分和调度
  - 数据的分布传输
  - 计算及处理结果的收集
  - 处理系统节点出错检测和失效恢复
  - 系统管理、负载平衡、计算性能优化
  - ……
  - 提供简单、易用的编程接口

MapReduce的基本思想

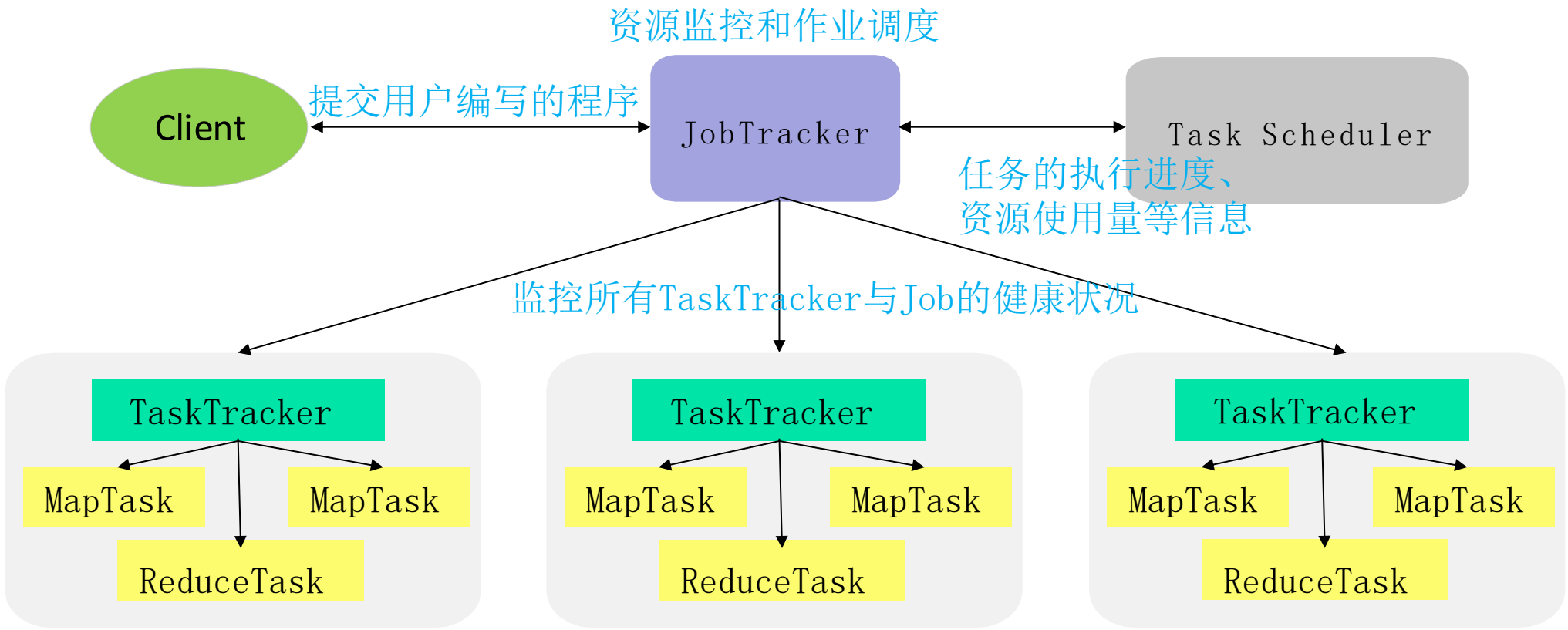
- 采用分而治之的思想实现大规模数据的并行运算
- Map函数：
  - 大量数据记录进行重复、简单处理
  - 只需要局部信息，获得中间结果
- Reduce函数：
  - 整理（全局的）中间结果
  - 产生最终结果输出



一个Map-Reduce任务分解示例



MapReduce的基本框架

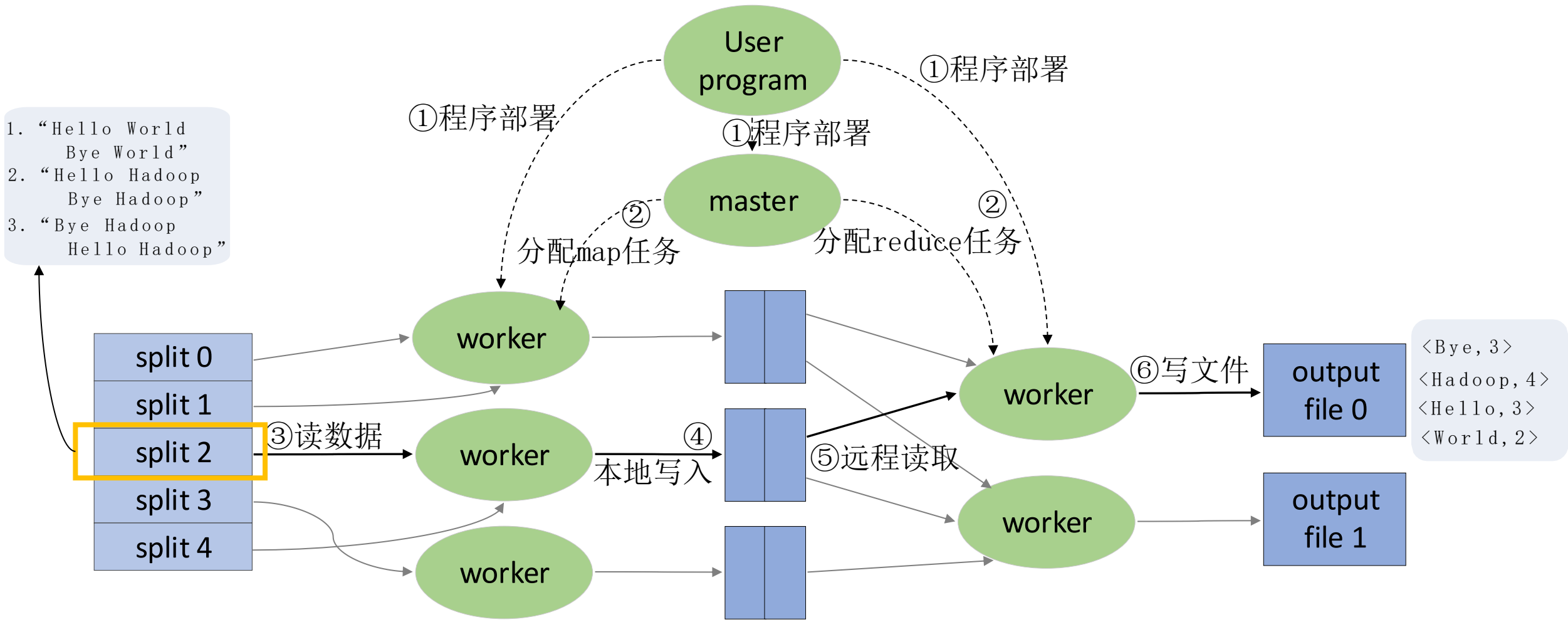


TaskTracker使用“slot”等量划分本节点上的资源量

slot 分为Map slot 和Reduce slot 两种，分别供MapTask 和Reduce Task 使用

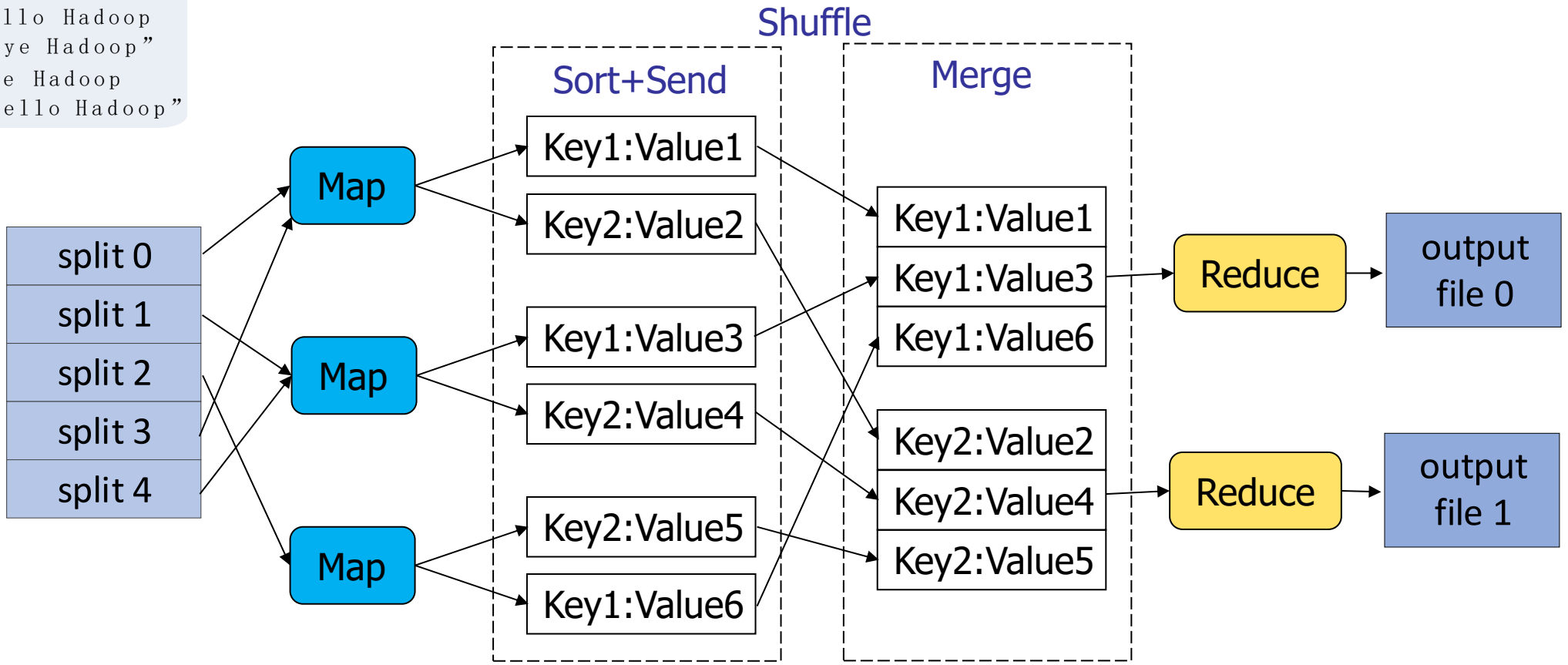
MapReduce的工作流程

“移动代码，而不是移动数据!”



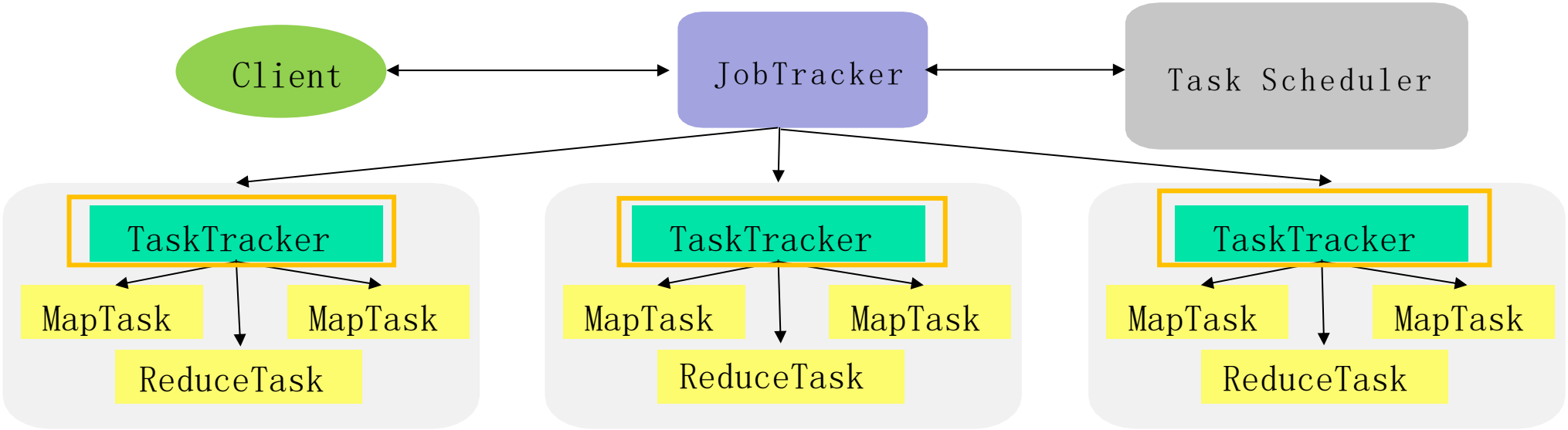
MapReduce中的“局部计算”和“全局计算”

1. “Hello World
   Bye World”
2. “Hello Hadoop
   Bye Hadoop”
3. “Bye Hadoop
   Hello Hadoop”



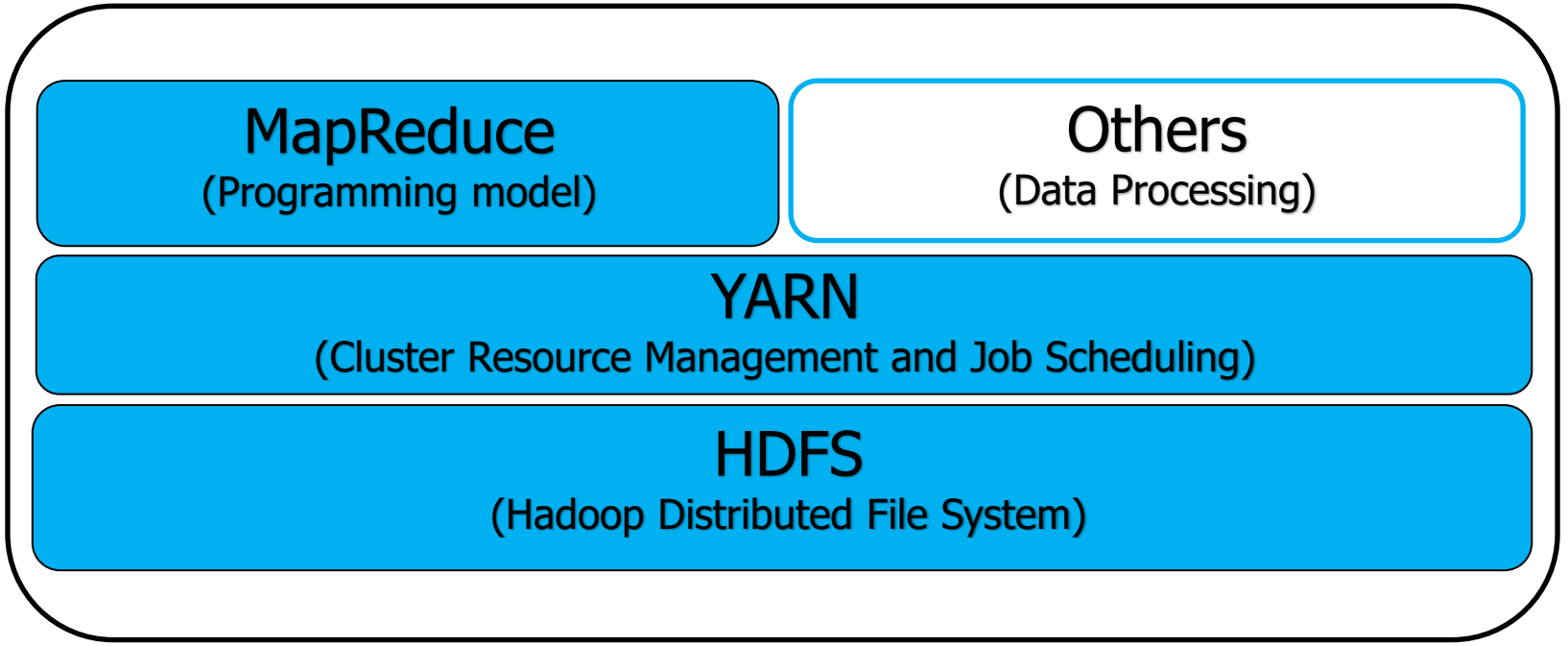
MapReduce 1.0计算框架的缺点

- 既是一个计算框架也是一个资源管理调度框架，存在一些缺陷：
  - 存在单点故障
  - JobTracker“大包大揽”导致任务过重（任务多时内存开销大，一般上限4000节点）
  - 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）
  - 资源划分不合理（强制划分Map slot和Reduce slot）



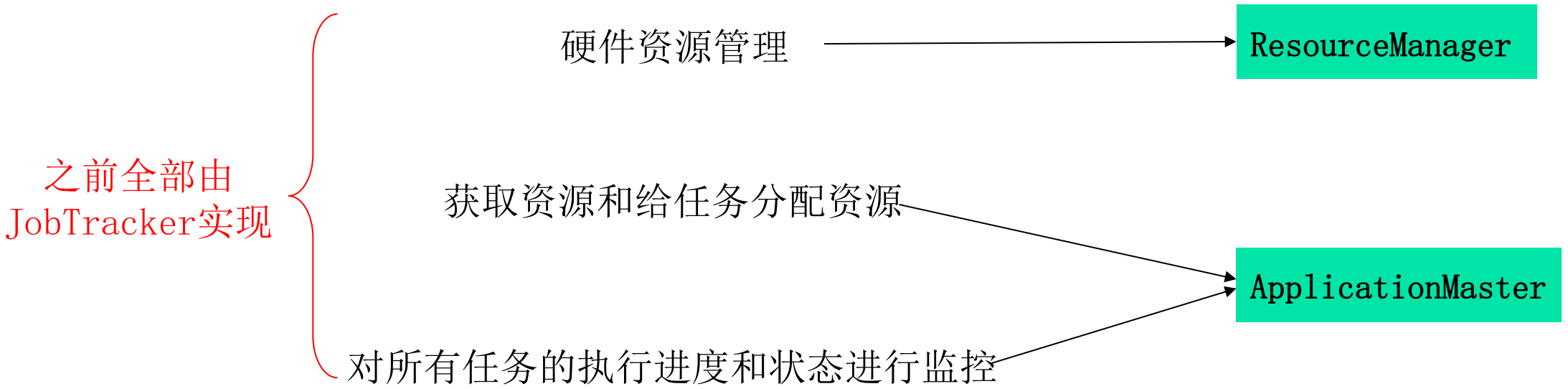
YARN的产生

- Hadoop2.0以后， MapReduce1.0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
- MapReduce2.0成为了运行在YARN之上的一个纯粹的计算框架



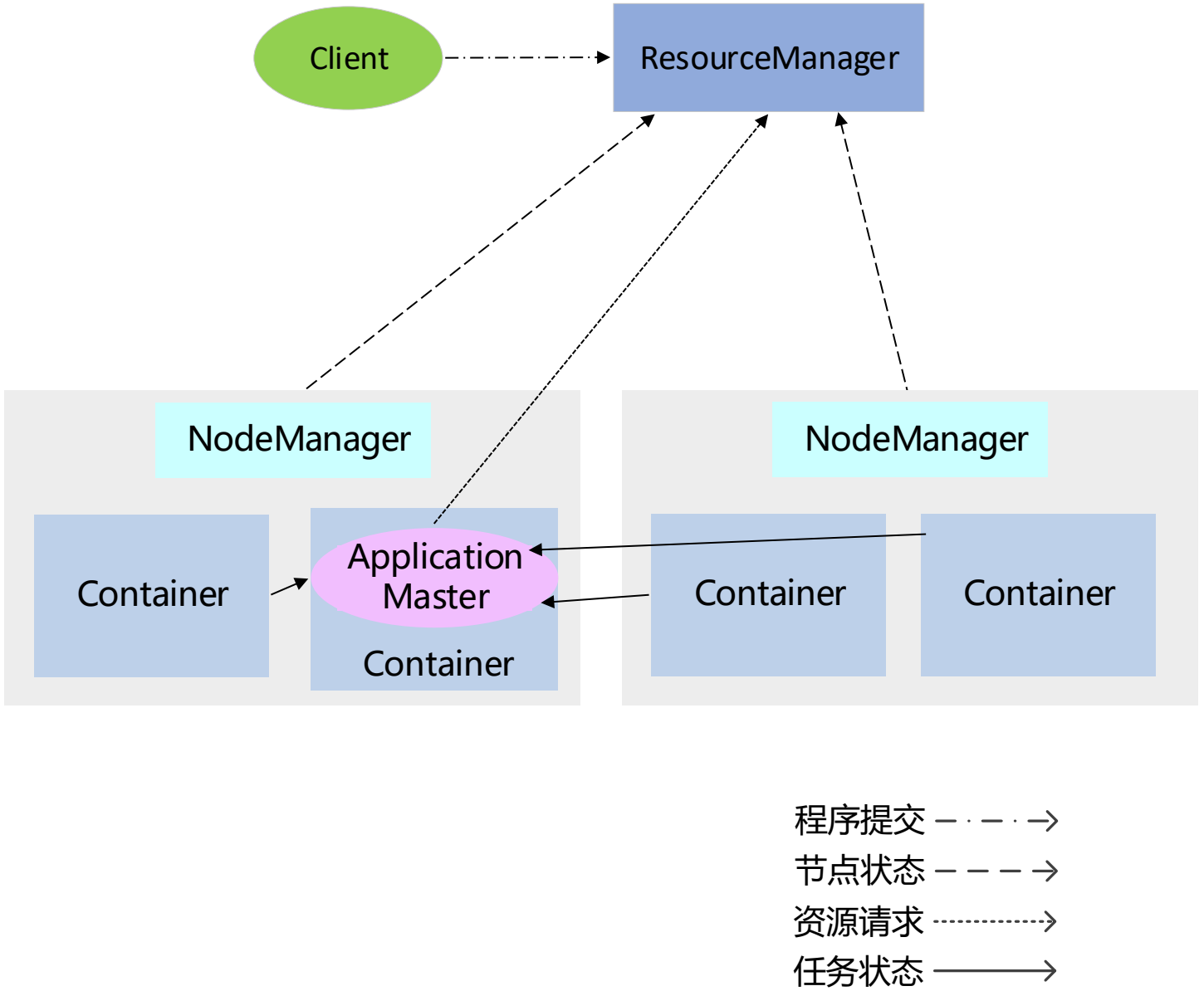
YARN的总体设计思路

- 将原JobTracker三大功能分层解耦：

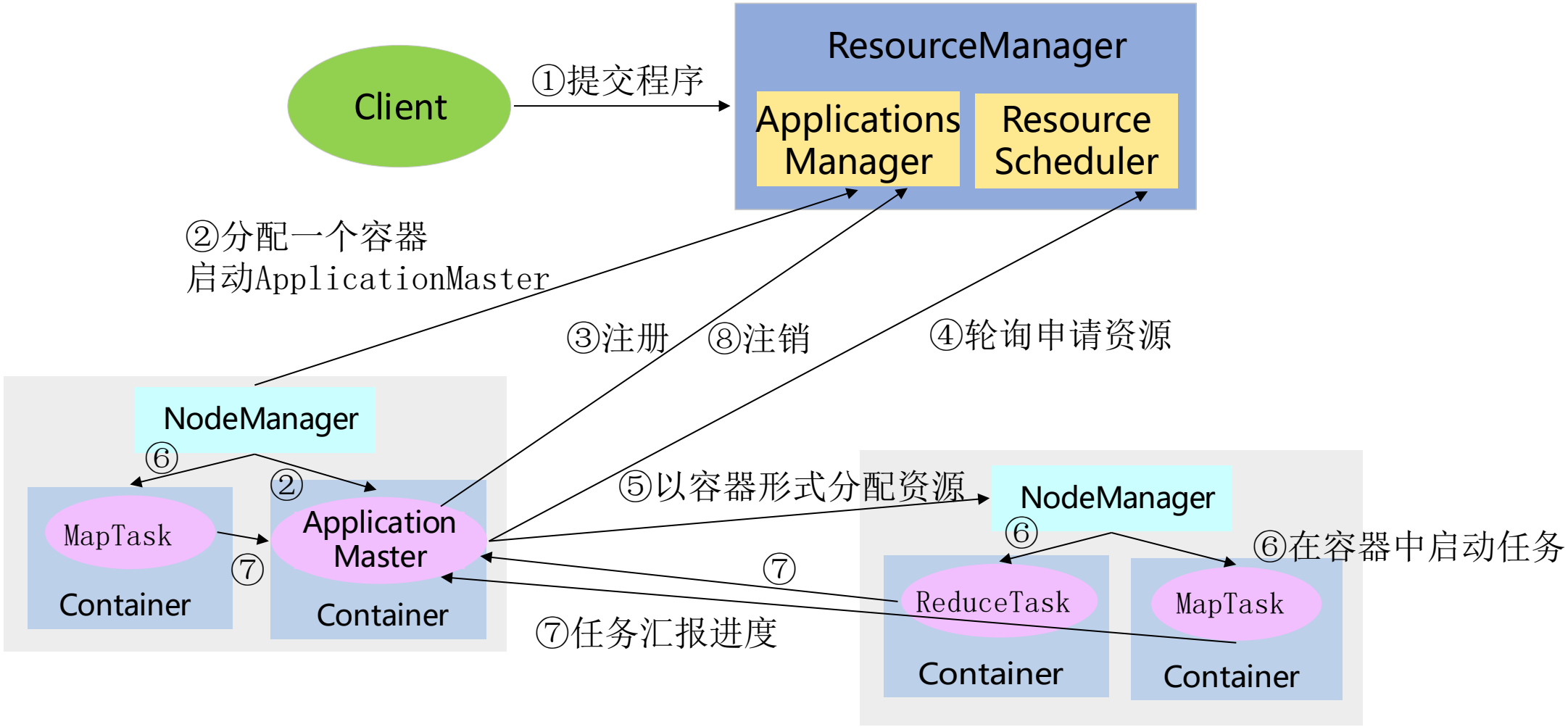


YARN的基本框架

- 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，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



YARN的工作流程


Client
①提交程序
ResourceManager
Applications Manager
Resource Scheduler
②分配一个容器
启动ApplicationMaster
③注册
⑧注销
④轮询申请资源
NodeManager
⑥
②
MapTask
Application Master
⑦
Container
Container
⑤以容器形式分配资源
NodeManager
⑥
⑥在容器中启动任务
⑦
ReduceTask
MapTask
⑦任务汇报进度
Container
Container

大数据处理框架Spark：基础原理

Spark简介

- Spark最初由美国加州伯克利大学（UCBerkeley）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

Reference: https://dblab.xmu.edu.cn/post/5633/

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

Reference: https://dblab.xmu.edu.cn/post/5633/

Spark与Hadoop Map-reduce计算框架的对比

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：

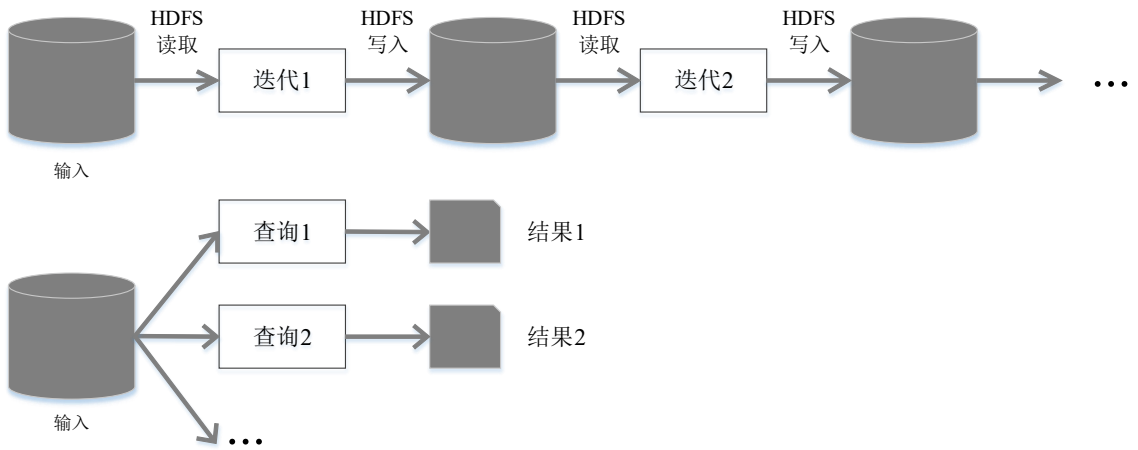
•Spark的计算模式也属于MapReduce，但不局限于Map和Reduce操作，还提供了多种数据集操作类型，编程模型比Hadoop MapReduce更灵活

•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

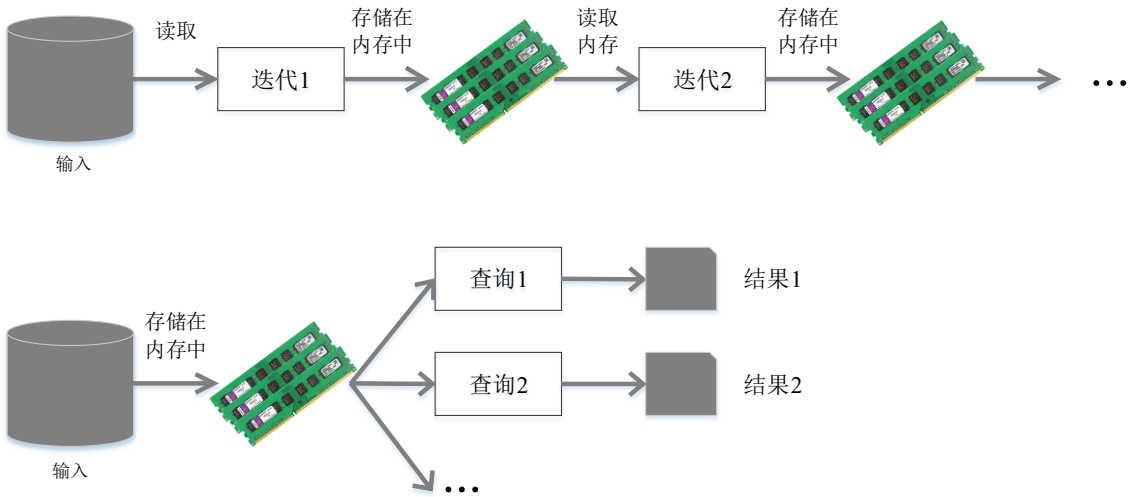
Spark基于DAG的任务调度执行机制，要优于Hadoop MapReduce的迭代执行机制

Reference: https://dblab.xmu.edu.cn/post/5633/

Spark与Hadoop Map-reduce计算框架的对比



(a) Hadoop MapReduce执行流程



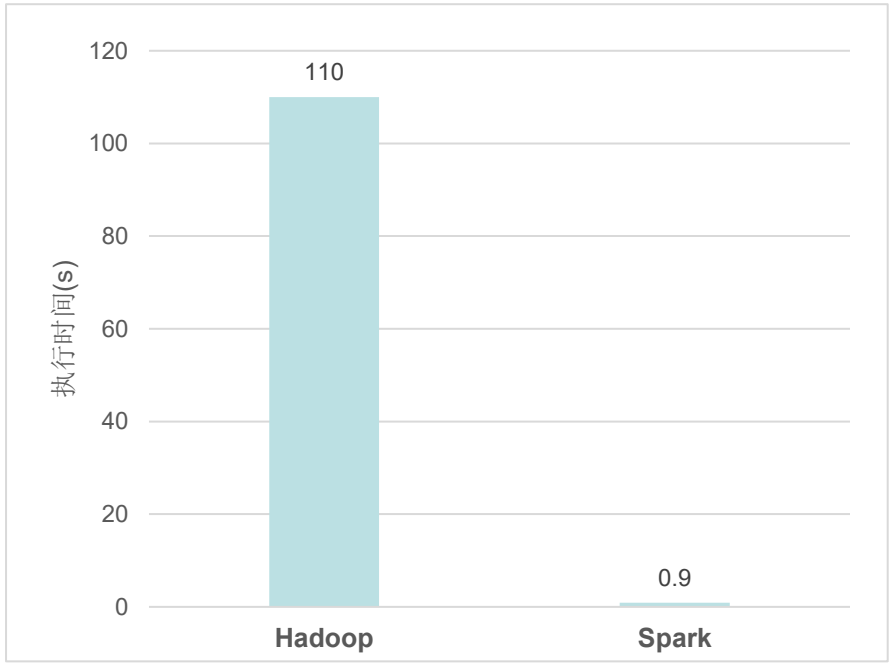
(b) Spark执行流程

Hadoop与Spark的执行流程对比

Reference: https://dblab.xmu.edu.cn/post/5633/

Spark与Hadoop Map-reduce计算框架的对比

- 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



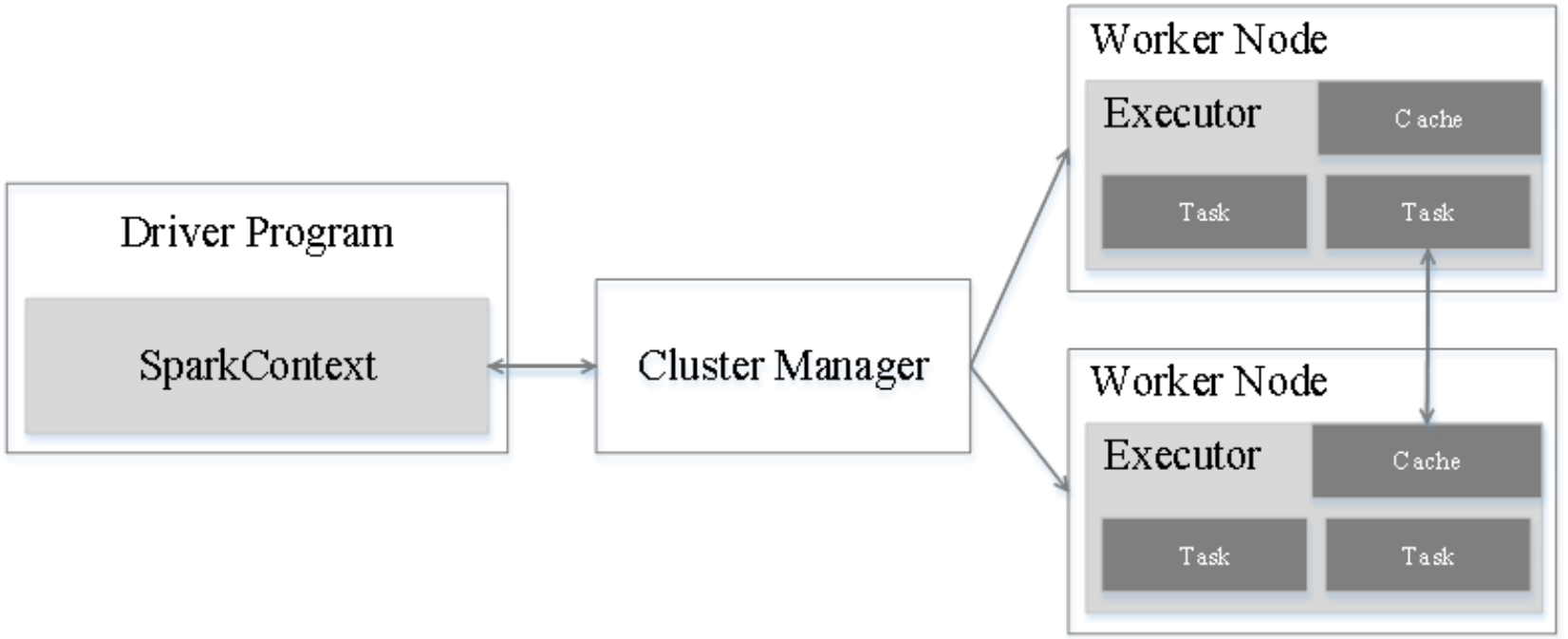
Hadoop与Spark执行逻辑回归的时间对比

Reference: https://dblab.xmu.edu.cn/post/5633/

Spark的运行框架

•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

与Hadoop MapReduce计算框架相比，Spark所采用的Executor有两个优点：
•一是利用多线程来执行具体的任务，减少任务的启动开销
•二是Executor中有一个BlockManager存储模块，会将内存和磁盘共同作为存储设备，有效减少IO开销

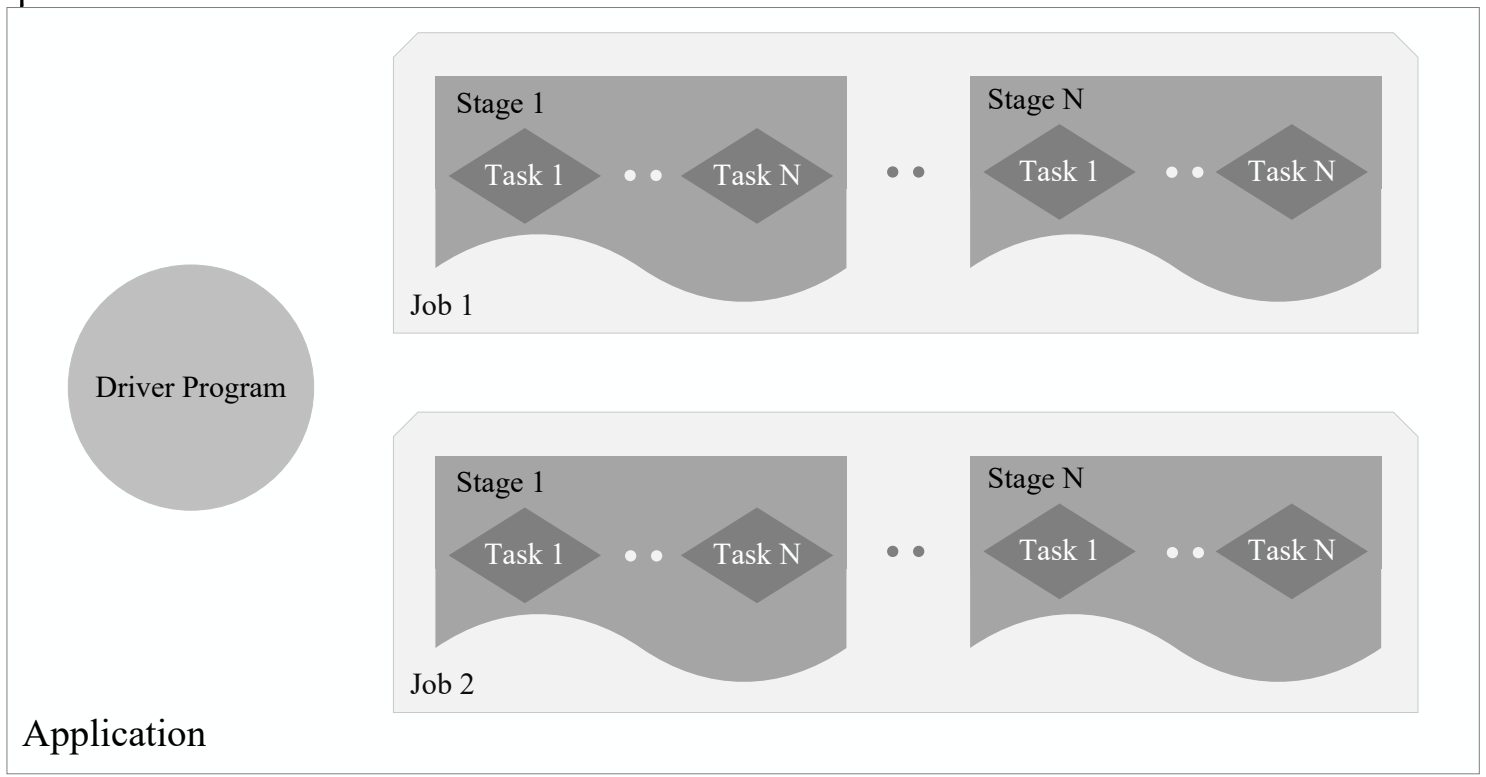


Reference: https://dblab.xmu.edu.cn/post/5633/

Spark的运行框架

•一个Application由一个Driver和若干个Job构成，一个Job由多个Stage构成，一个Stage由多个没有Shuffle关系的Task组成

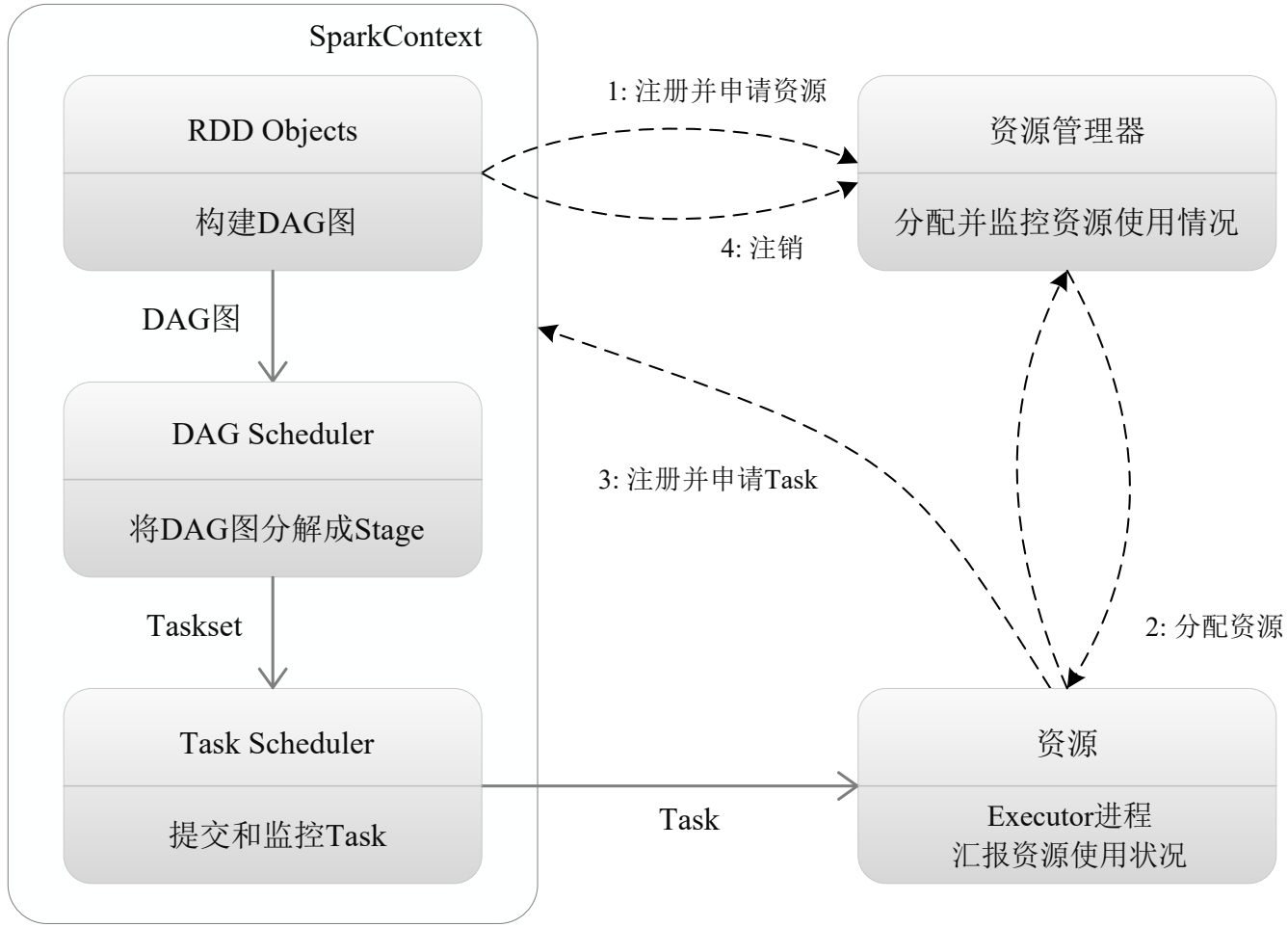
•当执行一个Application时，Driver会向集群管理器申请资源，启动Executor，并向Executor发送应用程序代码和文件，然后在Executor上执行Task，运行结束后，执行结果会返回给Driver，或者写到HDFS或者其他数据库中



Spark中各种概念之间的相互关系

Reference: https://dblab.xmu.edu.cn/post/5633/

Spark的运行框架



Spark运行基本流程图

（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

Reference: https://dblab.xmu.edu.cn/post/5633/

Spark的运行框架

总体而言，Spark运行架构具有以下特点：

（1）每个Application都有自己专属的Executor进程，并且该进程在Application运行期间一直驻留。Executor进程以多线程的方式运行Task

（2）Spark运行过程与资源管理器无关，只要能够获取Executor进程并保持通信即可

（3）Task采用了数据本地性和推测执行等优化机制

Reference: https://dblab.xmu.edu.cn/post/5633/

RDD的运行原理

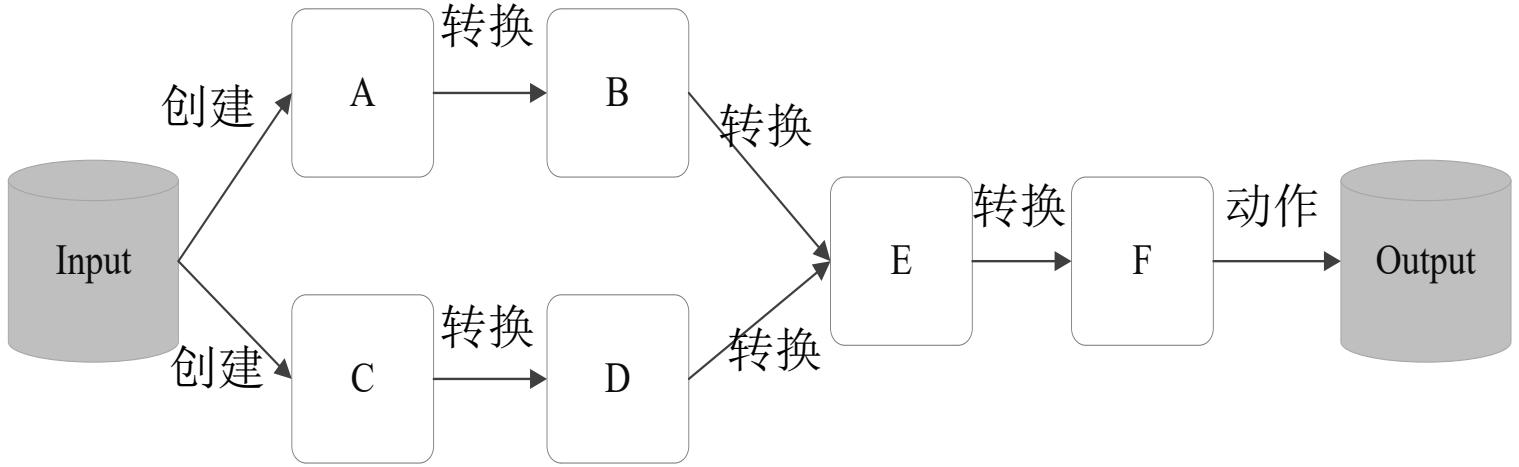
- 许多迭代式算法（比如机器学习、图算法等）和交互式数据挖掘工具，共同之处是，不同计算阶段之间会重用中间结果
- 一个RDD就是一个分布式对象集合，本质上是一个只读的分区记录集合，每个RDD可分成多个分区，每个分区就是一个数据集片段，并且一个RDD的不同分区可以被保存到集群中不同的节点上，从而可以在集群中的不同节点上进行并行计算
- RDD提供了一种高度受限的共享内存模型，即RDD是只读的记录分区的集合，不能直接修改，只能基于稳定的物理存储中的数据集创建RDD，或者通过在其他RDD上执行确定的转换操作（如map、join和group by）而创建得到新的RDD

Reference: https://dblab.xmu.edu.cn/post/5633/

RDD的运行原理

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

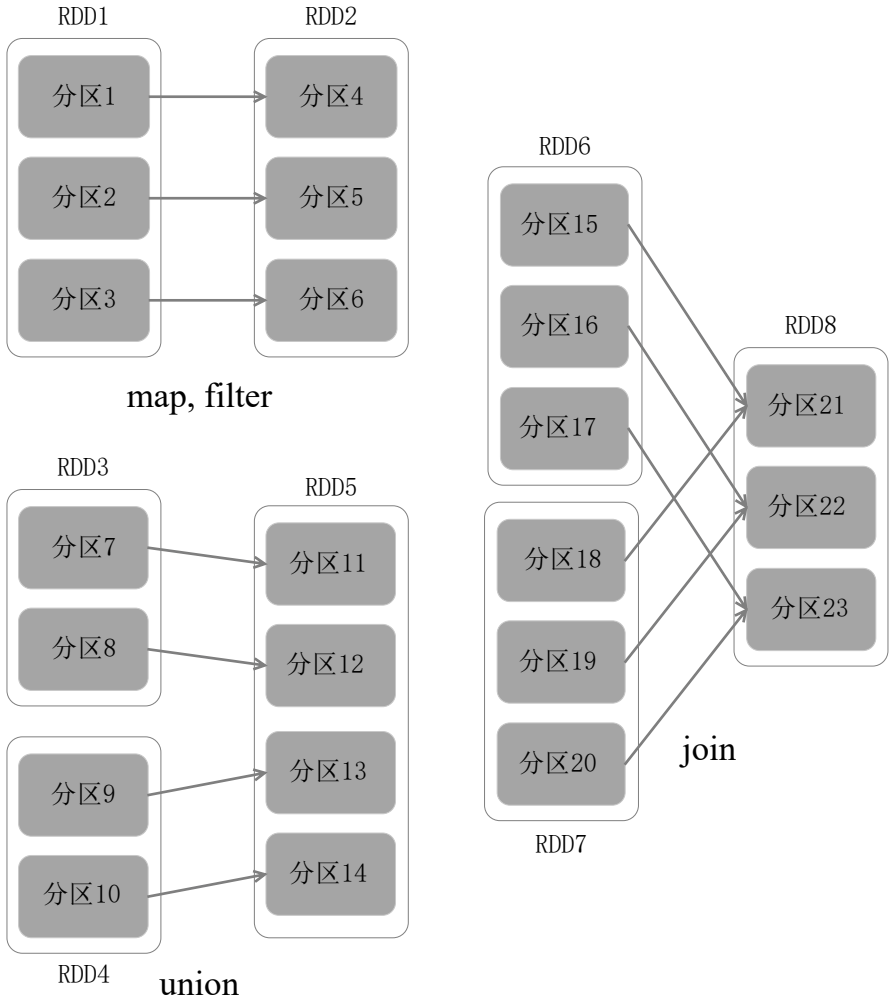


RDD执行过程的一个实例

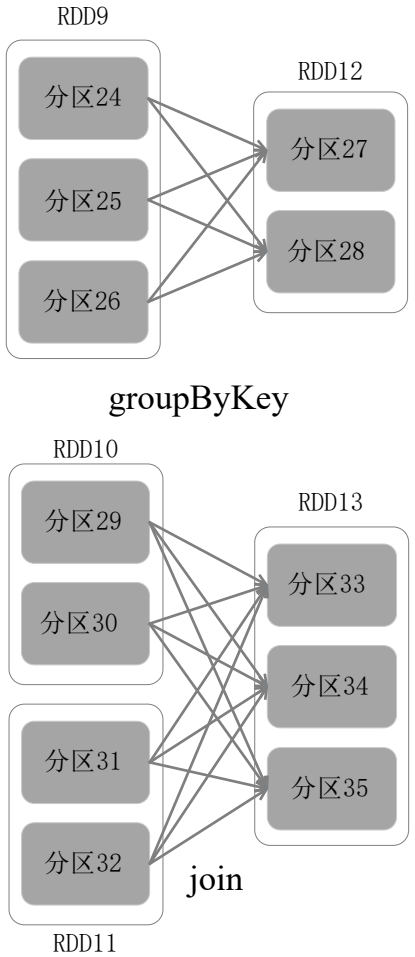
Reference: https://dblab.xmu.edu.cn/post/5633/

RDD的运行原理

RDD之间的依赖关系



(a)窄依赖



(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

Reference: https://dblab.xmu.edu.cn/post/5633/

RDD的运行原理

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：

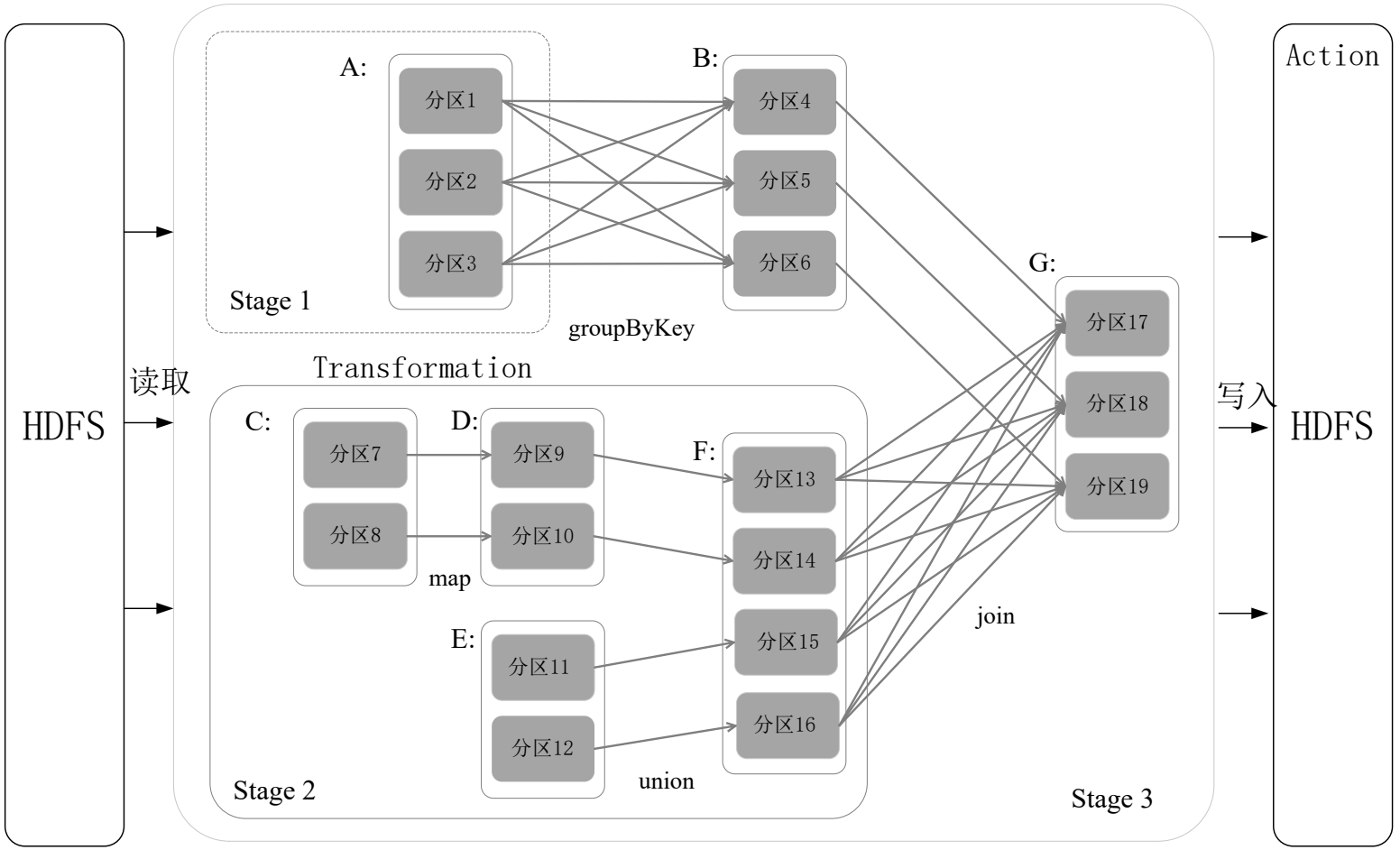
- 在DAG中进行反向解析，遇到宽依赖就断开
- 遇到窄依赖就把当前的RDD加入到Stage中
- 将窄依赖尽量划分在同一个Stage中，可以实现流水线计算

Reference: https://dblab.xmu.edu.cn/post/5633/

RDD的运行原理

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



根据RDD分区的依赖关系划分Stage

流水线操作实例

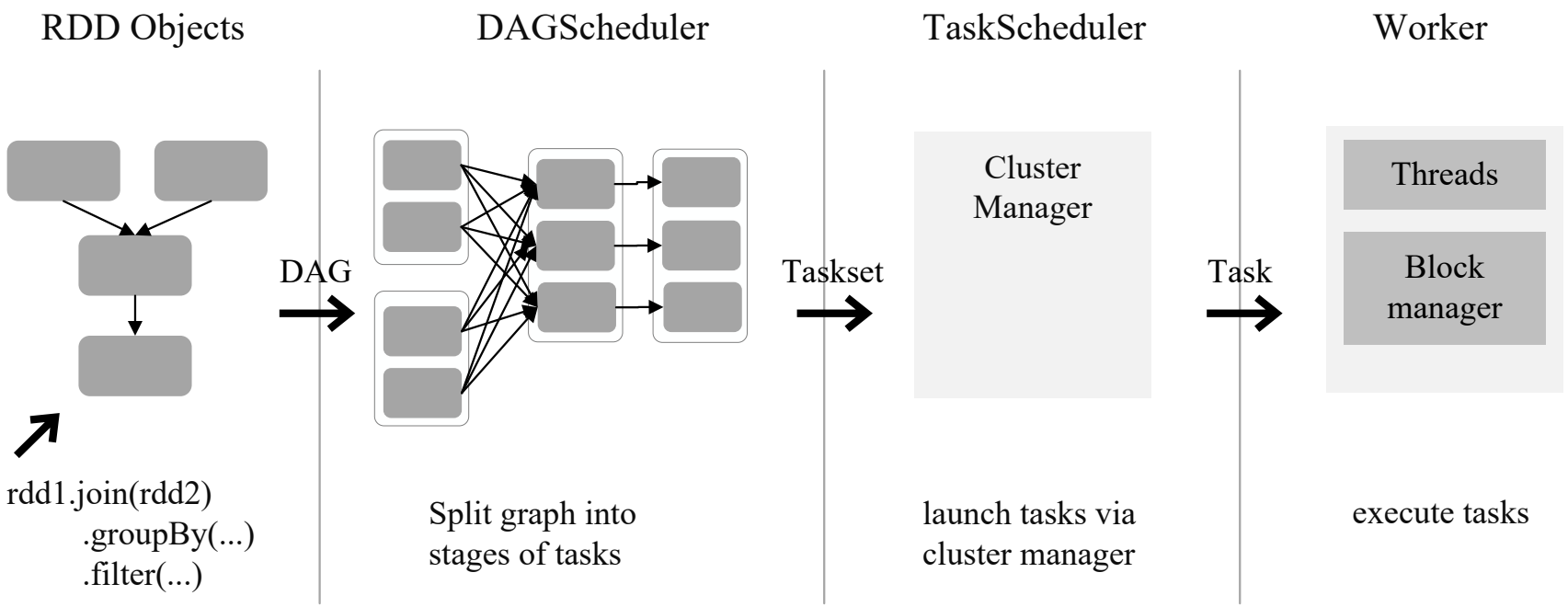
分区7通过map操作生成的分区9，可以不用等待分区8到分区10这个map操作的计算结束，而是继续进行union操作，得到分区13，这样流水线执行大大提高了计算的效率

Reference: https://dblab.xmu.edu.cn/post/5633/

RDD的运行原理

RDD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。

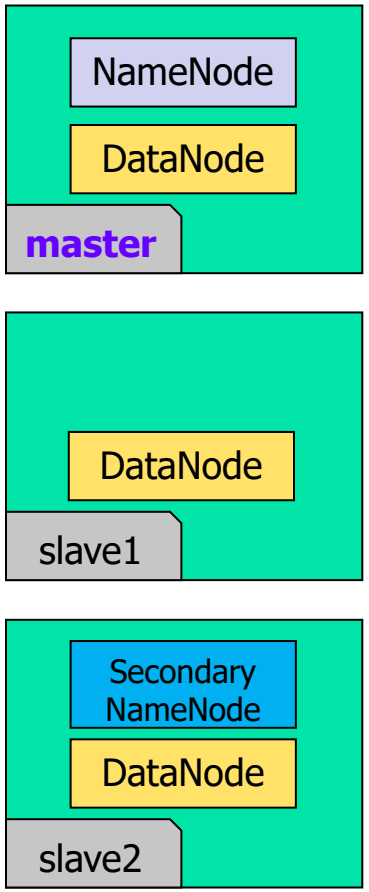


RDD在Spark中的运行过程

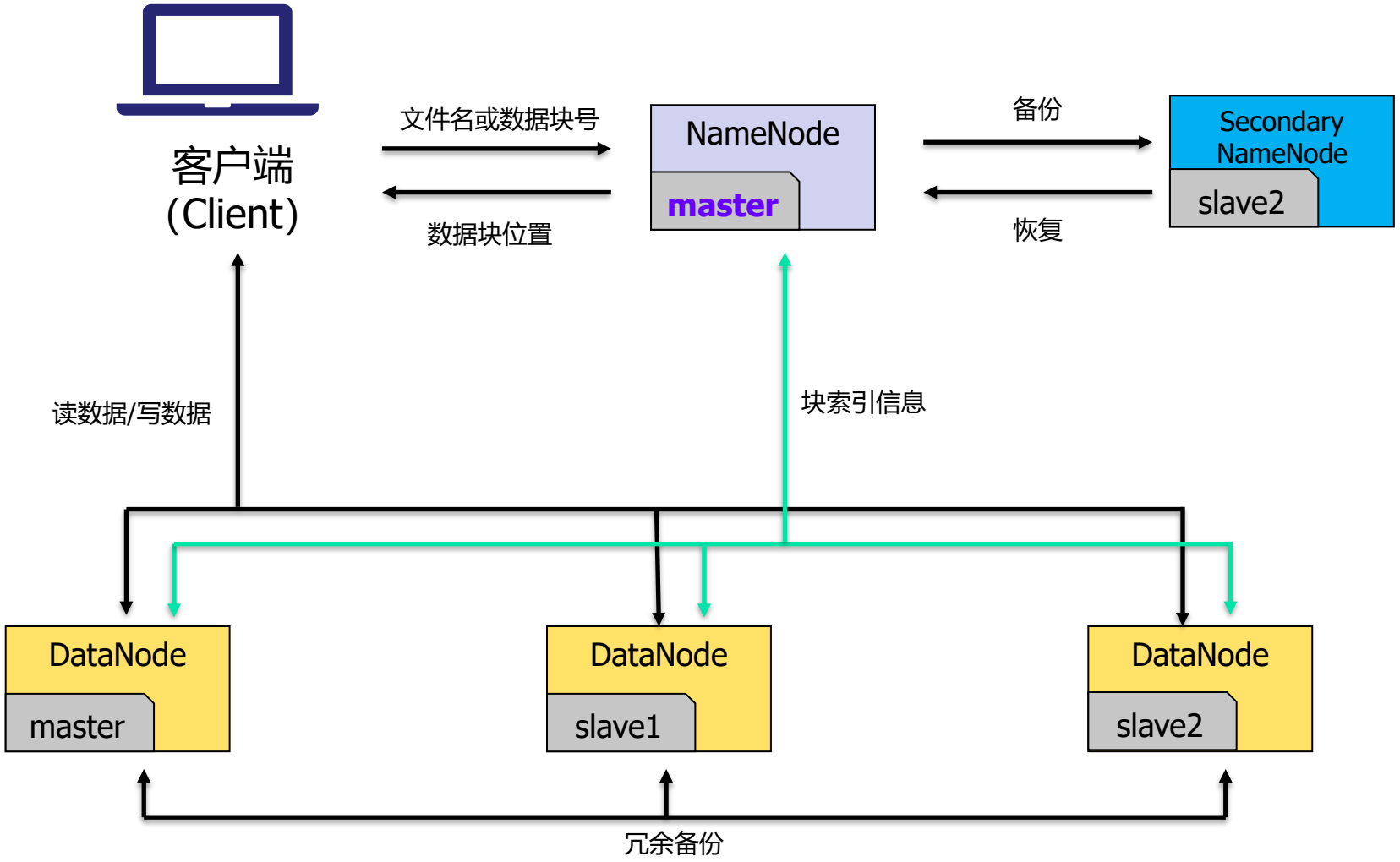
Reference: https://dblab.xmu.edu.cn/post/5633/

Hadoop平台搭建示例

HDFS环境示例



物理节点规划



集群架构

Browse Directory

/ Go!

Browse Directory

/exp2/douban Go!

Show 25 entries Search:

Permission	Owner	Name
-rw-r--r--	ices	comment_split.txt
-rw-r--r--	ices	comments.txt
-rw-r--r--	ices	movie_comment.json

Showing 1 to 3 of 3 entries Previous 1 Next

Hadoop, 2021.

Showing 1 to 10 of 10 entries Previous 1 Next

Hadoop, 2021.

File information - comment_split.txt

Download Head the file (first 32K) Tail the file (last 32K)

Block information -- Block 0

Block ID: 1073741856

Block Pool ID: BP-424571681-10.249.182.54-1633752692273

Generation Stamp: 1032

Size: 134217728

Availability:

- ices-master
- ices-slave1
- ices-slave2

Close

HDFS命令行操作（1）

查看所有命令

```
$ hadoop fs
```

```
[-appendToFile <localsrc> ... <dst>]
        [-cat [-ignoreCrc] <src> ...]
        [-chgrp [-R] GROUP PATH...]
        [-chmod [-R] <MODE[,MODE]... | OCTALMODE> PATH...]
        [-chown [-R] [OWNER][:[GROUP]] PATH...]
        [-copyFromLocal [-f] [-p] <localsrc> ... <dst>]
        [-copyToLocal [-p] [-ignoreCrc] [-crc] <src> ... <localdst>]
        [-count [-q] <path> ...]
        [-cp [-f] [-p] <src> ... <dst>]
        [-df [-h] [<path> ...]]
        [-du [-s] [-h] <path> ...]
        [-get [-p] [-ignoreCrc] [-crc] <src> ... <localdst>]
        [-getmerge [-nl] <src> <localdst>]
        [-help [cmd ...]]
        [-ls [-d] [-h] [-R] [<path> ...]]
        [-mkdir [-p] <path> ...]
        [-moveFromLocal <localsrc> ... <dst>]
```

HDFS命令行操作（2）

创建目录 /bigdata

```
$ hadoop fs -mkdir /bigdata
```

移动根目录文件 test.txt **到** /bigdata

```
$ hadoop fs -mv /test.txt /bigdata
```

显示 /bigdata **目录信息**

```
$ hadoop fs -ls /bigdata
```

```
(base) ices@ices-master:~$ hadoop fs -mkdir /bigdata
(base) ices@ices-master:~$ hadoop fs -mv /test.txt /bigdata
(base) ices@ices-master:~$ hadoop fs -ls /bigdata
Found 1 items
-rw-r--r--   3 ices supergroup          8 2021-11-02 15:42 /bigdata/test.txt
```

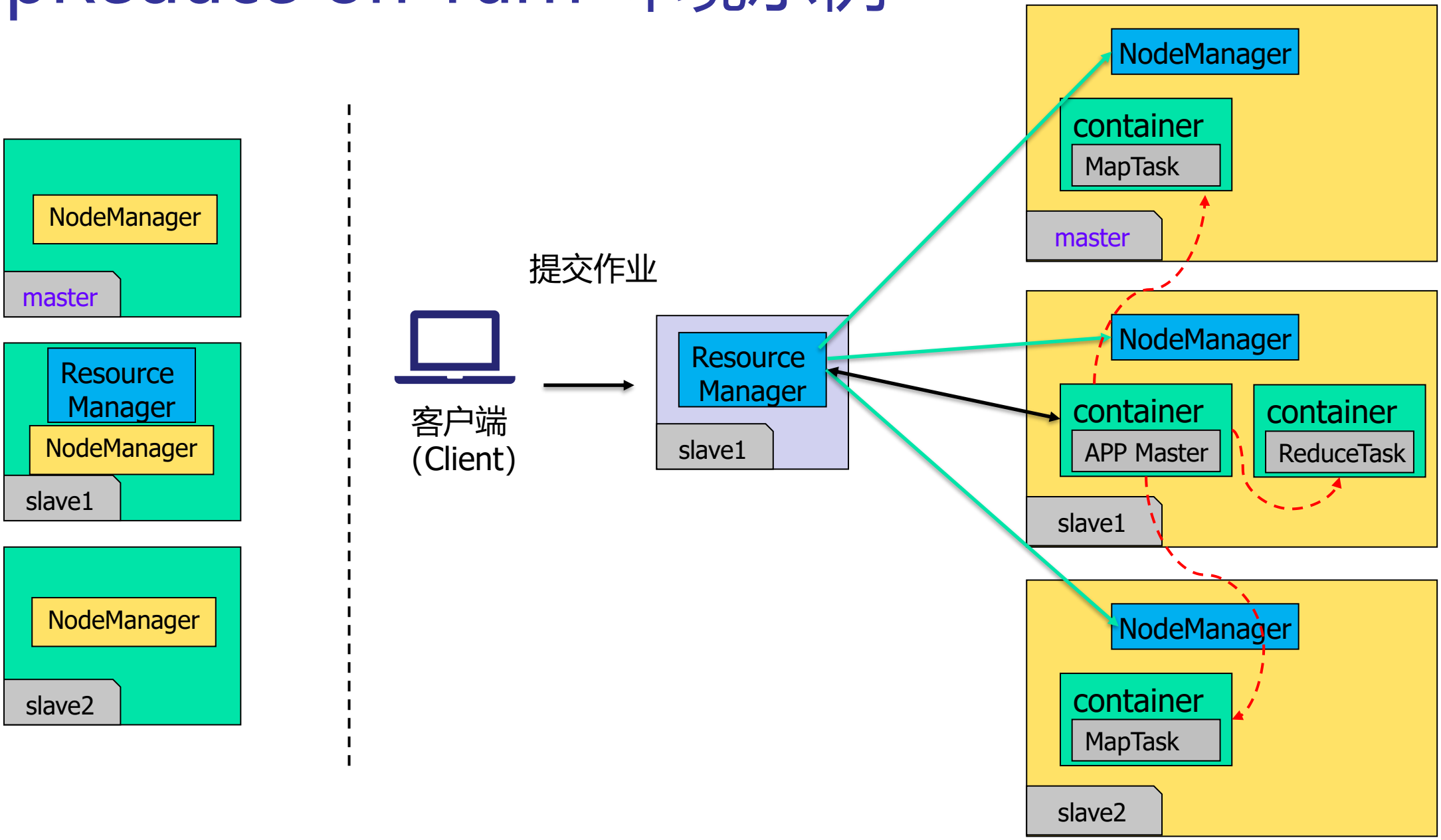
HDFS存取管理的Java API示例

```
public class HdfsClient {
    @Test
    public void testMkdirs() throws IOException, URISyntaxException, InterruptedException {
        // 1 获取文件系统
        Configuration configuration = new Configuration();
        FileSystem fs = FileSystem.get(new URI("hdfs://ices-master:8020"),
                                                configuration, "ices");
        // 2 创建目录
        fs.mkdirs(new Path("/hitsz/bigdata/"));

       // 3 列出目录下文件夹和文件名称
       FileStatus[] statuses = fs.listStatus(new Path("/hitsz"));
        for (FileStatus file : statuses){
            String isDir = file.isDirectory() ? "Folder " : "File";
            String path = file.getPath().getName();
            System.out.println(isDir+ "\t" +path);
        }

        fs.close();
    }
```

MapReduce on Yarn 环境示例



MapReduce作业运行示例

启动MapReduce作业：

Hadoop jar XXX.jar(jar包) XXX(类名) /input /output

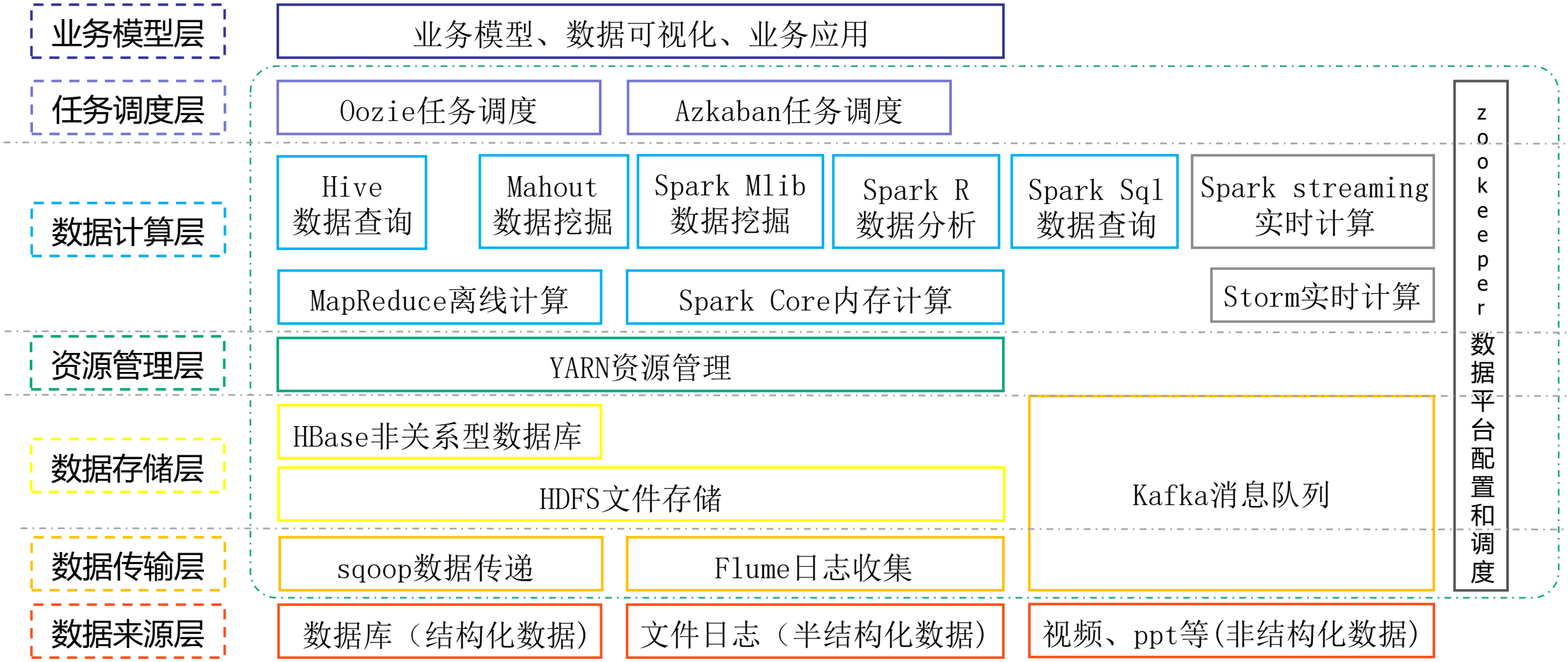
```
hadoop@ubuntu:/usr/local/hadoop$ ./bin/hadoop jar join.jar Reducejoin /input /ou
tput
20/05/13 20:23:01 WARN util.NativeCodeLoader: Unable to load native-hadoop libra
ry for your platform... using builtin-java classes where applicable
20/05/13 20:23:09 INFO Configuration.deprecation: session.id is deprecated. Inst
ead, use dfs.metrics.session-id
20/05/13 20:23:09 INFO jvm.JvmMetrics: Initializing JVM Metrics with processName
=JobTracker, sessionId=
20/05/13 20:23:09 WARN mapreduce.JobResourceUploader: Hadoop command-line option
 parsing not performed. Implement the Tool interface and execute your applicatio
n with ToolRunner to remedy this.
20/05/13 20:23:16 INFO input.FileInputFormat: Total input paths to process : 2
20/05/13 20:23:16 INFO mapreduce.JobSubmitter: number of splits:2
20/05/13 20:23:17 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_lo
cal281746740_0001
20/05/13 20:23:17 INFO mapreduce.Job: The url to track the job: http://localhost
:8080/
20/05/13 20:23:17 INFO mapreduce.Job: Running job: job_local281746740_0001
20/05/13 20:23:17 INFO mapred.LocalJobRunner: OutputCommitter set in config null
20/05/13 20:23:17 INFO output.FileOutputCommitter: File Output Committer Algorit
hm version is 1
20/05/13 20:23:17 INFO mapred.LocalJobRunner: OutputCommitter is org.apache.hado
op.mapreduce.lib.output.FileOutputCommitter
20/05/13 20:23:17 INFO mapred.LocalJobRunner: Waiting for map tasks
20/05/13 20:23:17 INFO mapred.LocalJobRunner: Starting task: attempt_local281746
740_0001_m_000000_0
20/05/13 20:23:17 INFO output.FileOutputCommitter: File Output Committer Algorit
hm version is 1
```

实践任务：一个简化的分布式大数据存储与处理示例

- 从点评网站上下载5万个网页并保存到Hadoop分布式文件存储系统中
- 解析各个网页，用Hadoop分布式处理框架统计出top 20的关键词

知识拓展：Hadoop生态及组件

Hadoop生态

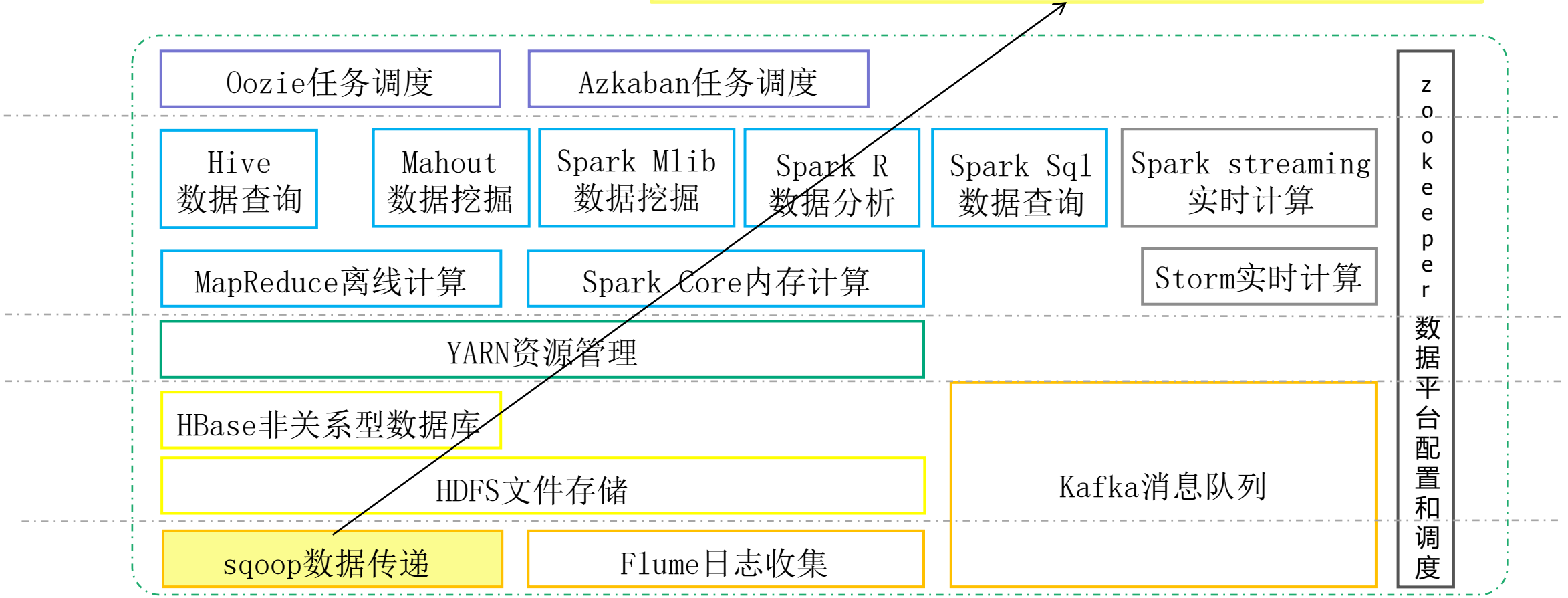

业务模型层
业务模型、数据可视化、业务应用
任务调度层
Oozie任务调度
Azkaban任务调度
数据计算层
Hive
数据查询
Mahout
数据挖掘
Spark Mlib
数据挖掘
Spark R
数据分析
Spark Sql
数据查询
Spark streaming
实时计算
MapReduce离线计算
Spark Core内存计算
Storm实时计算
资源管理层
YARN资源管理
数据存储层
HBase非关系型数据库
HDFS文件存储
Kafka消息队列
数据传输层
sqoop数据传递
Flume日志收集
数据来源层
数据库（结构化数据）
文件日志（半结构化数据）
视频、ppt等(非结构化数据)
zookeeper
数据平台配置和调度

Hadoop组件功能

组件	功能
HDFS	分布式文件系统
MapReduce	分布式编程模型
YARN	资源管理框架
Zookeeper	可扩展协调系统
AVRO	序列化框架
Hbase	分布式数据库
Hive	数据仓库
Pig	脚本语言
Sqoop	同步处理工具

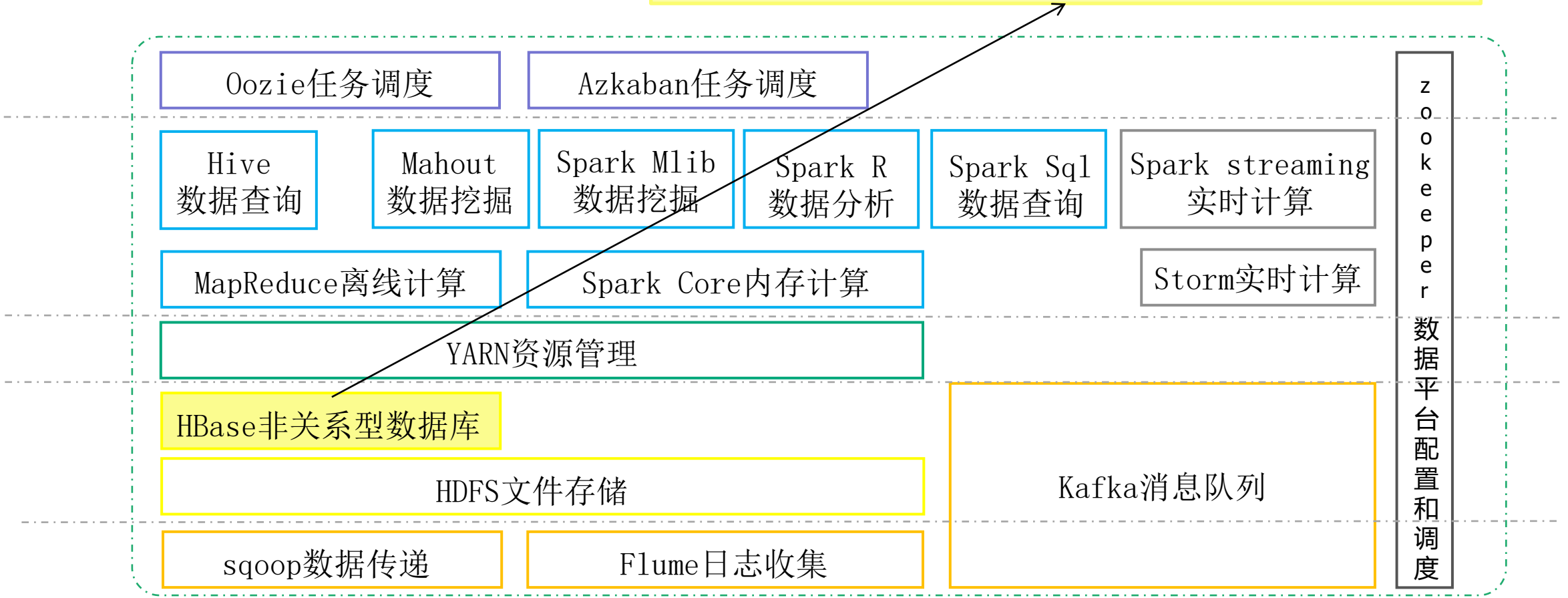
Hadoop组件功能

用于在Hadoop与传统的数据库间进行数据的传递，可以将一个关系型数据库（如： MySQL ,Oracle等）中的数据导进到Hadoop的HDFS中，也可以将HDFS的数据导进到关系型数据库



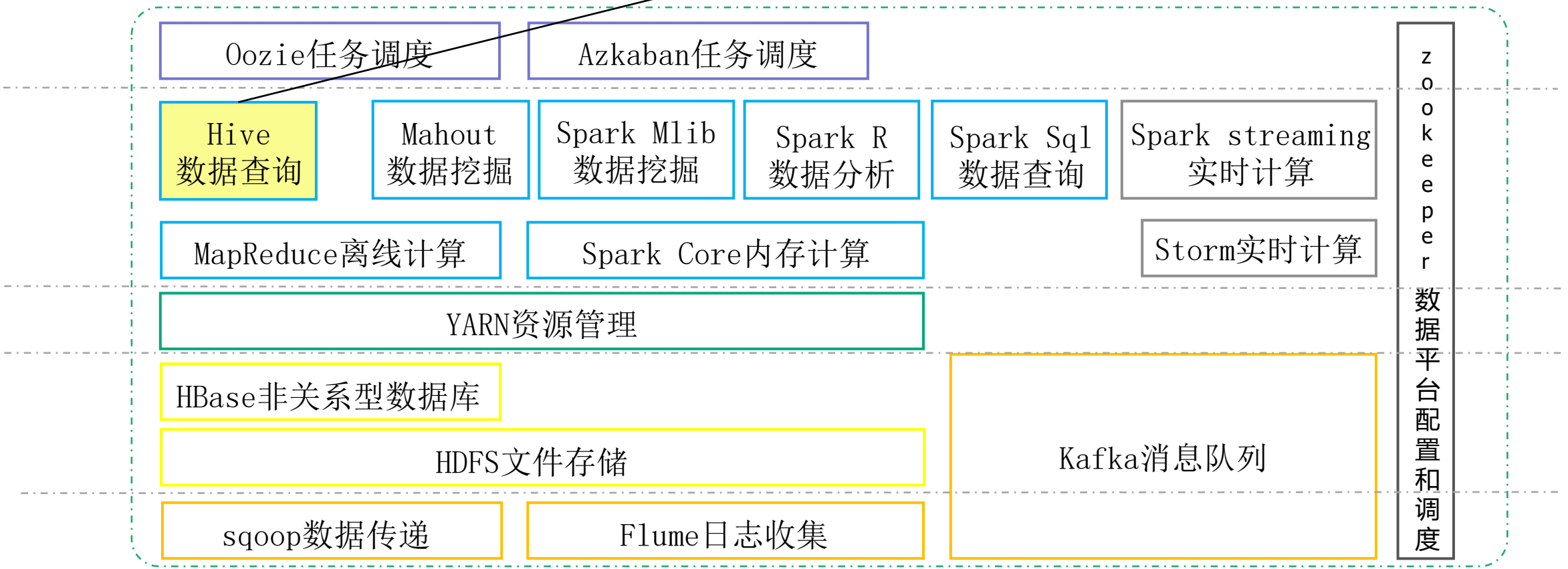
Hadoop组件功能

提供海量数据存储功能，是一种构建在HDFS之上的分布式、面向列的存储系统
对于半结构化或非结构化的数据很适合使用Hbase，因为Hbase支持动态添加列



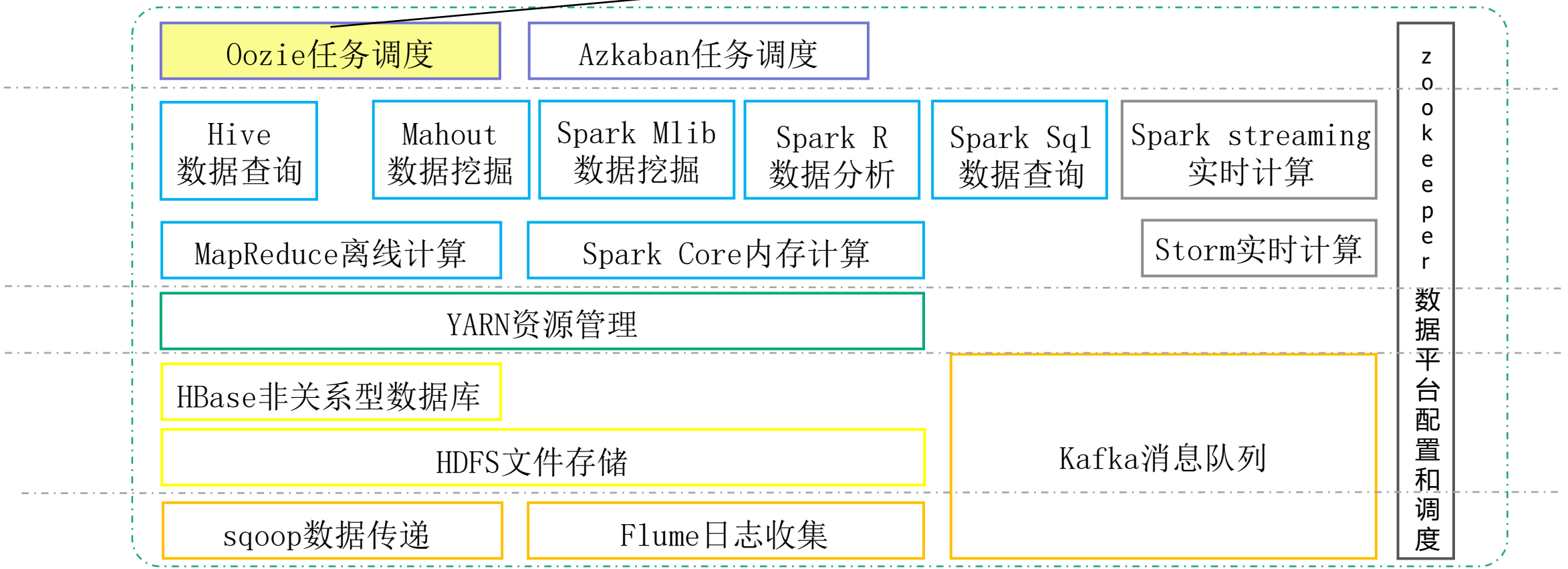
Hadoop组件功能

是基于Hadoop的一个数据仓库工具，可以将结构化的数据文件映射为一张数据库表并提供类sql查询功能



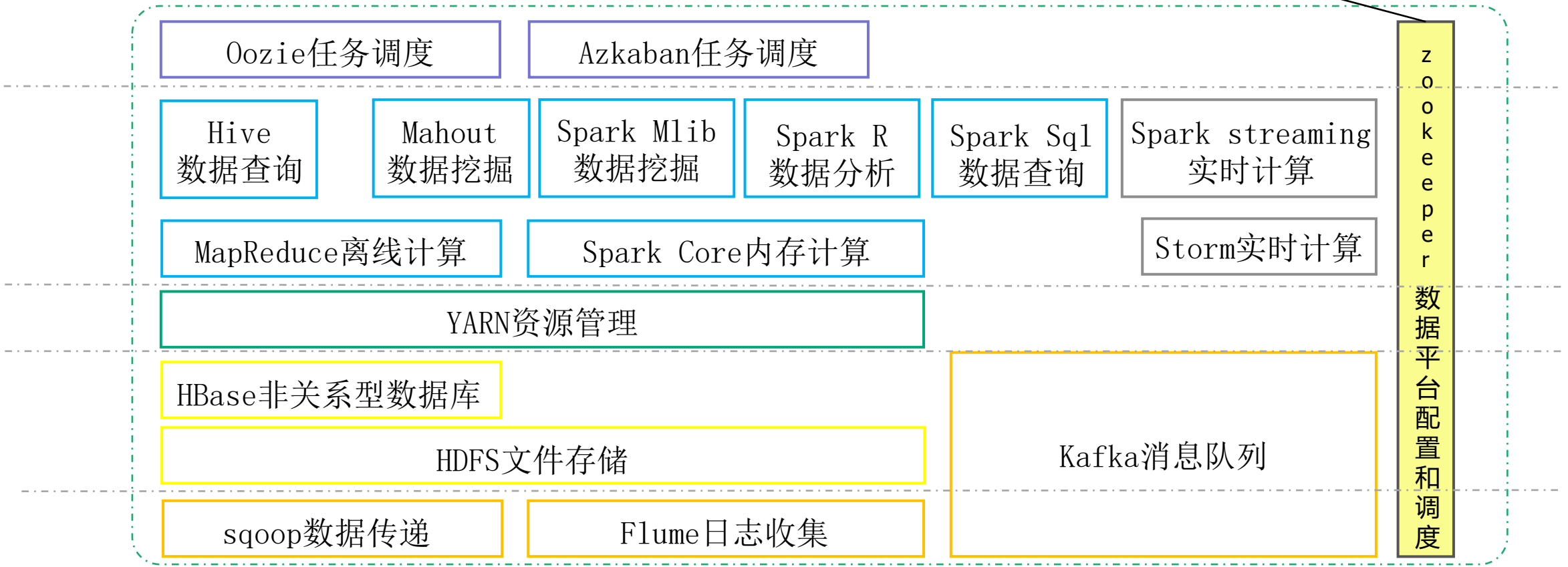
Hadoop组件功能

Oozie是一个用于管理Apache Hadoop作业的工作流调度程序系统，支持多种类型的Hadoop作业（例如Java map-reduce，Streaming map-reduce，Hive，Sqoop和Spark）以及系统特定的工作（例如Java程序和shell脚本）。



Hadoop组件功能

分布应用程序协调系统，为分布式应用提供一致性服务，如配置服务、名字服务等，有助于系统避免单点故障



Hadoop的典型应用

- 百度
  - 百度在2012年其总的集群规模达到近十个，单集群超过2800台机器节点，Hadoop机器总数有上万台机器，总的存储容量超过100PB，已经使用的超过74PB，每天提交的作业数目有数千个之多，每天的输入数据量已经超过7500TB，输出超过1700TB。
  - 百度的Hadoop集群主要应用包括
    - 数据挖掘与分析、日志分析平台、数据仓库系统、推荐引擎系统、用户行为分析系统

Hadoop的典型应用

- 阿里巴巴
  - 阿里巴巴的Hadoop集群截至2012年大约有3200台服务器，大约300000物理CPU核心，总内存100TB，总的存储容量超过60PB，每天的作业数目超过150000个，每天hive query查询大于6000个，每天扫描数据量约为7.5PB，每天扫描文件数约为4亿，存储利用率大约为80%，CPU利用率平均为65%，峰值可以达到80%。阿里巴巴的Hadoop集群拥有150个用户组、4500个集群用户，为淘宝、天猫、一淘、聚划算、CBU、支付宝提供底层的基础计算和存储服务
  - 阿里巴巴的Hadoop集群主要应用包括
    - 搜索支撑、广告系统、数据平台系统、量子统计、淘数据、推荐引擎系统、搜索排行榜

Hadoop的典型应用

- 腾讯
  - 腾讯也是使用Hadoop最早的中国互联网公司之一，截至2012年年底，腾讯的Hadoop集群机器总量超过5000台，最大单集群约为2000个节点，并利用Hadoop-Hive构建了自己的数据仓库系统TDW，同时还开发了自己的TDW-IDE基础开发环境。腾讯的Hadoop为腾讯各个产品线提供基础云计算和云存储服务
  - 腾讯的Hadoop服务于其下各种产品
    - 腾讯社交广告平台、QQ、QQ音乐等

致谢

- 一小部分图表、文字参考了教材、互联网上的开放资料等，本文件仅供公益性的学习参考，在此表示感谢！如有版权要求请联系：yym@hit.edu.cn，谢谢！