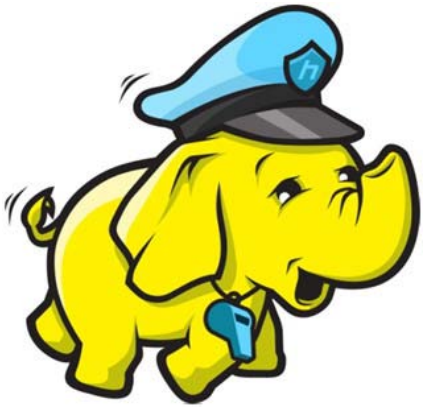
UC优视

UC Inside, World in Hand!



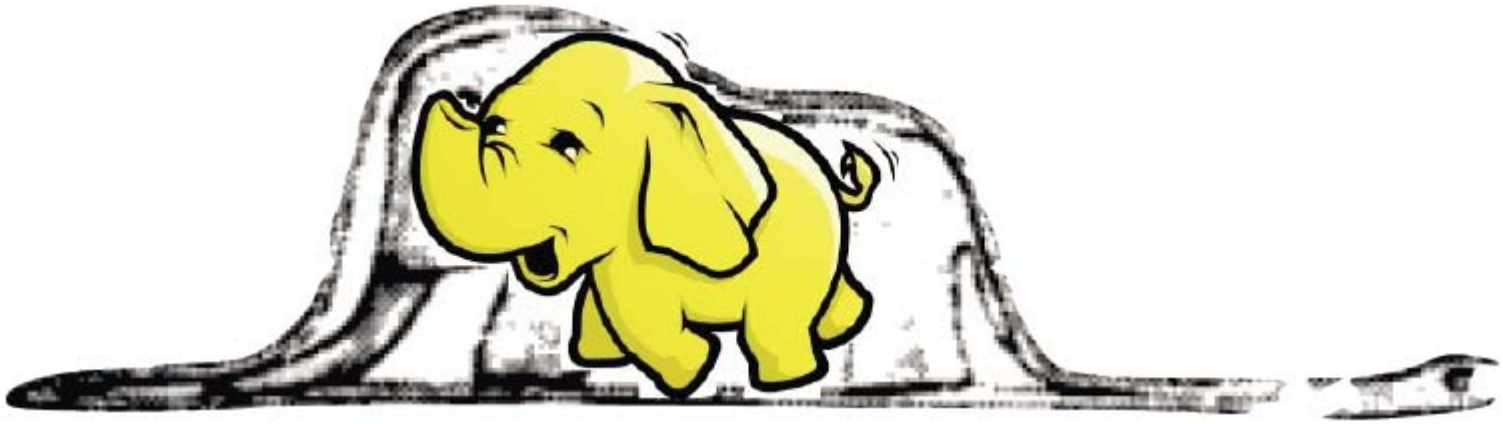
Hadoop原理介绍

基础应用组：郑海洪

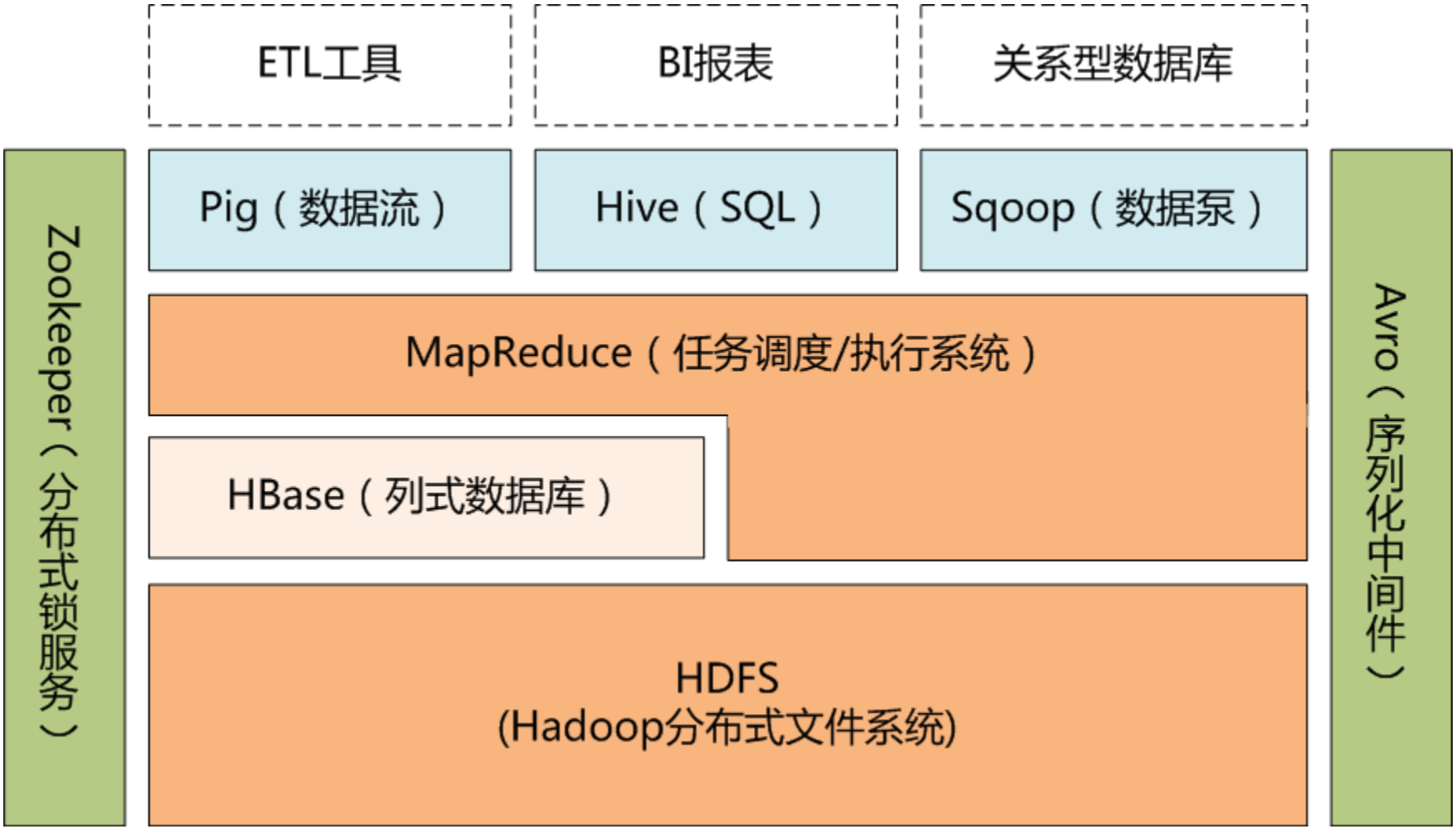
2011-03-10

揭开面纱

- Hadoop是一个开源的、可靠的、可扩展的分布式并行计算框架
- 主要组成：分布式文件系统HDFS和MapReduce算法执行
- 作者：Doug Cutting
- 语言：Java，支持多种编程语言，如：Python、C++
- 名称起源： Doug Cutting儿子的黄色大象玩具的名字



Hadoop生态圈



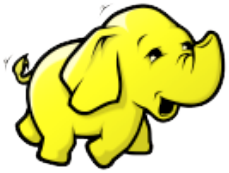
注：Hadoop的子项目包括HDFS和MapReduce，其他的项目属于Hadoop的相关项目，已经独立出去成为Apache的顶级项目。

假设与目标

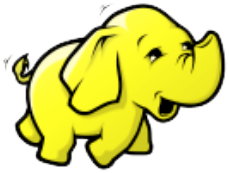
- 硬件错误是常态：自我修复
- 大数据集：文件大小GB、TB以上，上百个节点，上千万个文件
- 简单一致性模型：一次性写，多次读
- 移动计算环境比移动数据划算：计算尽可能接近数据
- 流式数据访问：高吞吐量、非低反应时间
- 跨硬件和软件平台的移动

演出开始

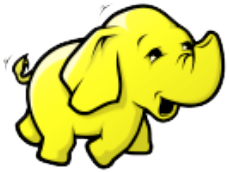
主 唱



NameNode (元数据服务器)

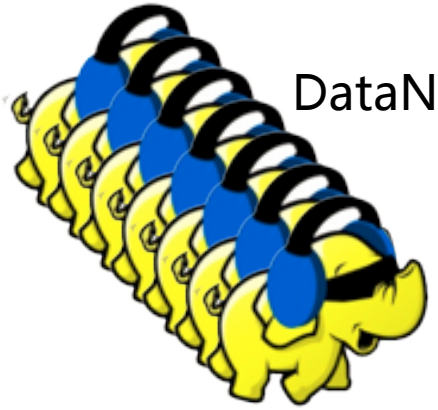


Secondary NameNode (辅助元数据服务器)



JobTracker (任务调度员)

合 唱



DataNodes(块存储)

TaskTrackers(任务执行)

部署模式

单机模式：只有一个JVM进程，没有分布式，不使用HDFS，通常用于调试。

伪分布式模式：只有一台机器，每个Hadoop守护进程都是一个独立的JVM进程，通常用于调试。

完全分布式模式：运行于多台机器上，真实环境。

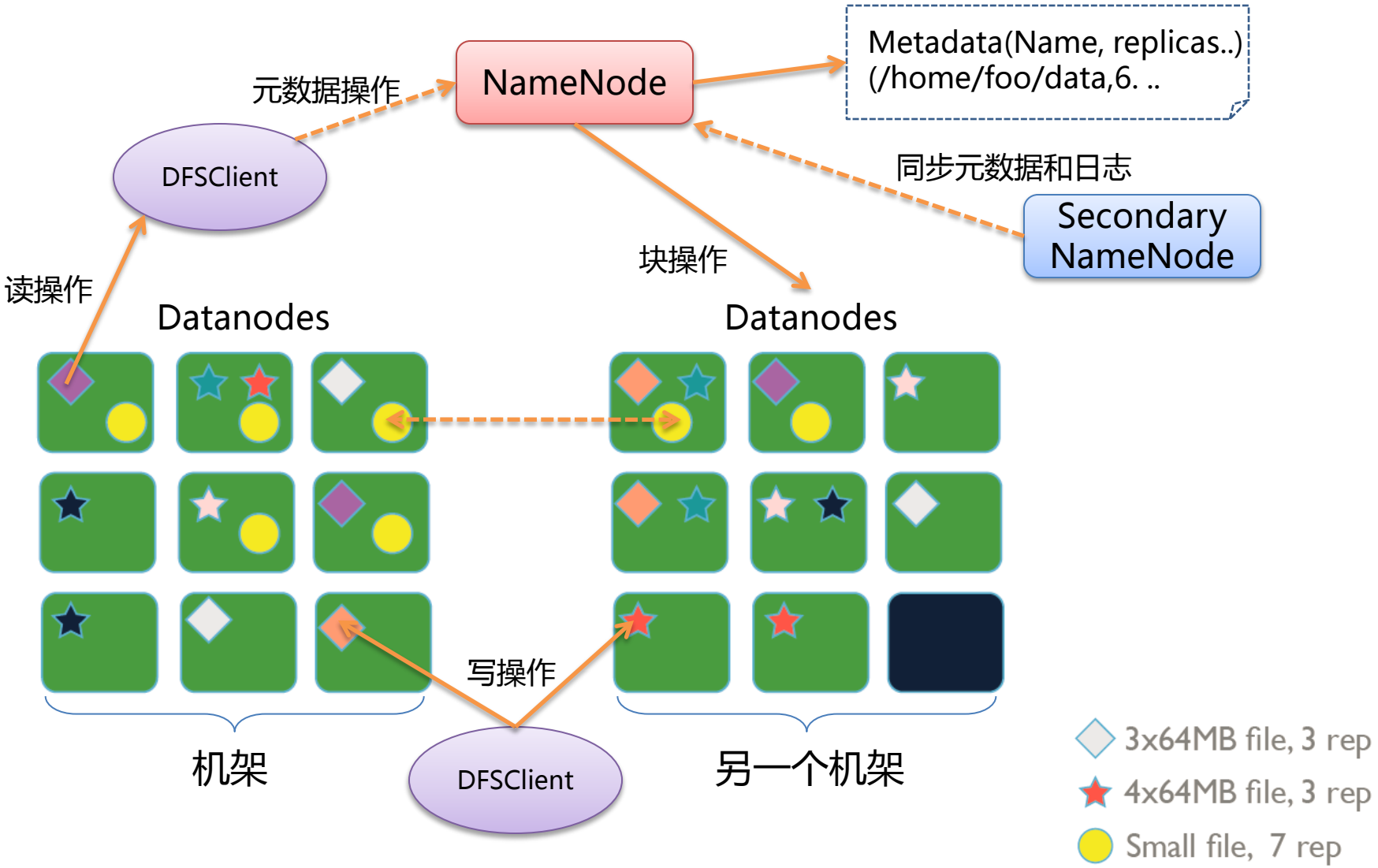
典型部署

- 5-4000 台服务器 (8-core, 8-24GB RAM, 4-12 TB, gig-E)
- 两层网络架构
- 每个机柜20-40节点

HDFS术语

HDFS	GFS	MooseFS	说明
NameNode	Master	Master	整个文件系统的大脑，它提供整个文件系统的目录信息，各个文件的分块信息，数据块的位置信息，并且管理各个数据服务器。
DataNode	Chunk Server	Chunk Server	分布式文件系统中的每一个文件，都被切分成若干个数据块，每一个数据块都被存储在不同的服务器上，此服务器称之为数据服务器。
Block	Chunk	Chunk	每个文件都会被切分成若干个块（默认64MB）每一块都有连续的一段文件内容，是存储的基本单位。
Packet	无	无	客户端写文件的时候，不是一个字节一个字节写入文件系统的，而是累计到一定数量后，往文件系统中写入一次，每发送一次的数据，都称为一个数据包。
Chunk	无	Block(64KB)	在每一个数据包中，都会将数据切成更小的块（512字节），每一个块配上一个奇偶校验码（CRC），这样的块，就是传输块。
Secondary NameNode	无	Metalogger	备用的主控服务器，在身后默默的拉取着主控服务器的日志，等待主控服务器牺牲后被扶正。

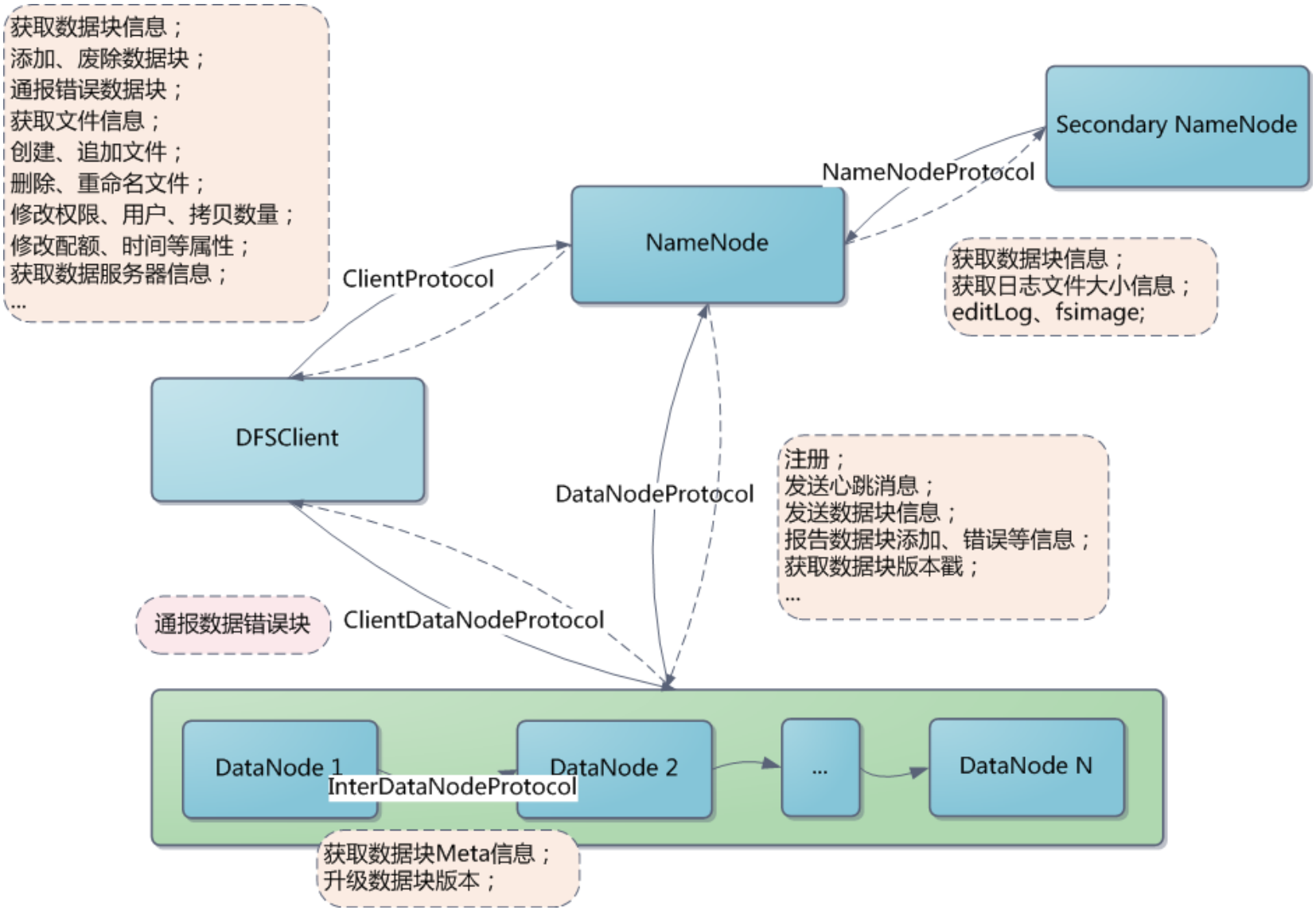
HDFS系统架构图



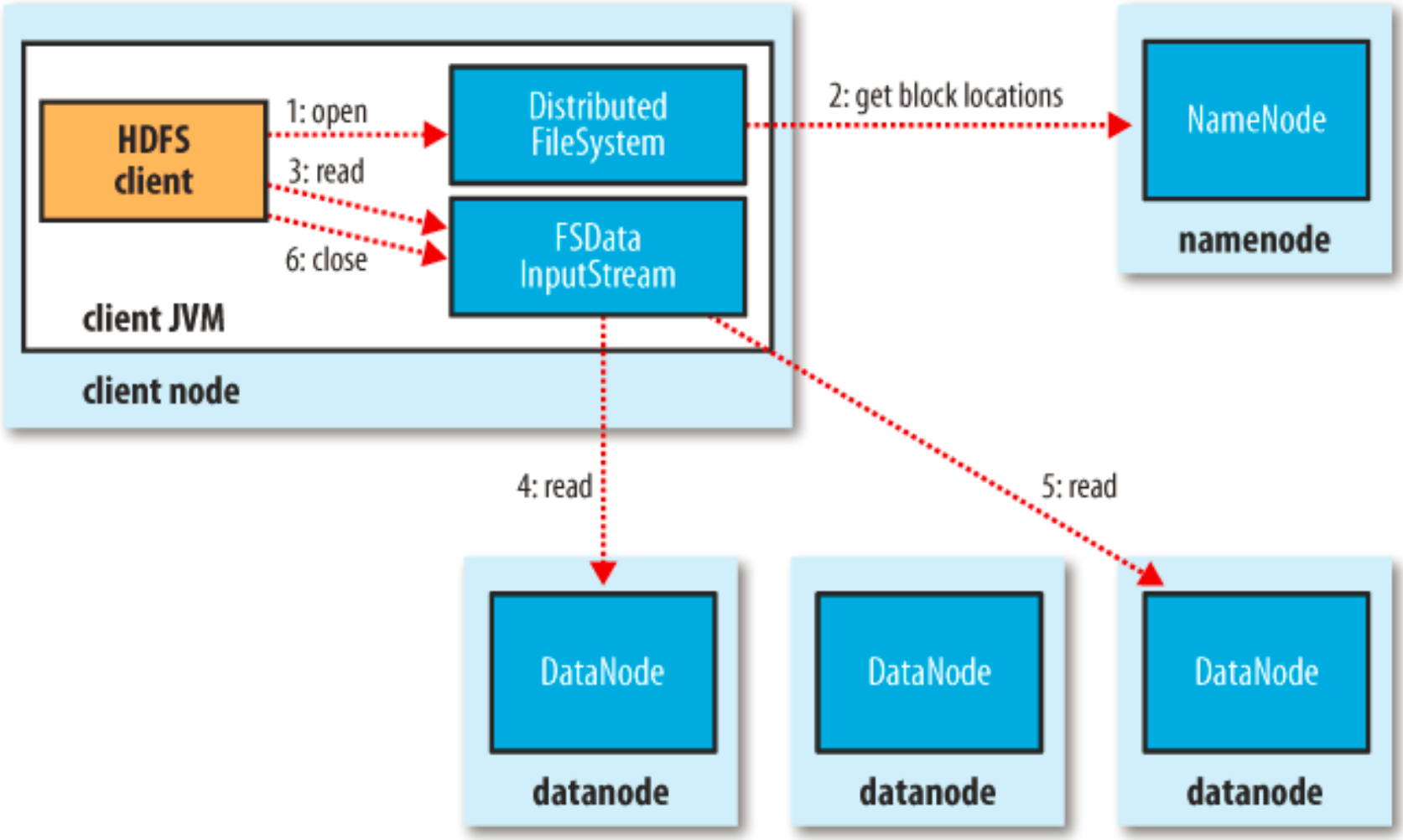
HDFS核心功能

主功能	说明
Namespace	HDFS支持传统的层次型文件组织，与大多数其他文件系统类似，用户可以创建目录，并在其间创建、删除、移动和重命名文件。
Shell命令	Hadoop包括一系列的类shell的命令，可直接和HDFS以及其他Hadoop支持的文件系统进行交互。
数据复制	每个文件的block大小和replication因子都是可配置的。Replication因子可以在文件创建的时候配置，以后也可以改变。HDFS中的文件是write-one，并且严格要求在任何时候只有一个writer。
机架感知	在大多数情况下，replication因子是3，HDFS的存放策略是将一个副本存放在本地机架上的节点，一个副本放在同一机架上的另一个节点，最后一个副本放在不同机架上的一个节点。机架的错误远远比节点的错误少，这个策略不会影响到数据的可靠性和有效性。
Editlog	FSEditLog类是整个日志体系的核心，提供了一大堆方便的日志写入API，以及日志的恢复存储等功能。
集群均衡	如果某个DataNode节点上的空闲空间低于特定的临界点，那么就会启动一个计划自动地将数据从一个DataNode搬移到空闲的DataNode。
空间的回收	删除文件并没有立刻从HDFS中删除，HDFS将这个文件重命名，并转移到/trash目录，用于恢复，/trash可设置保存时间。

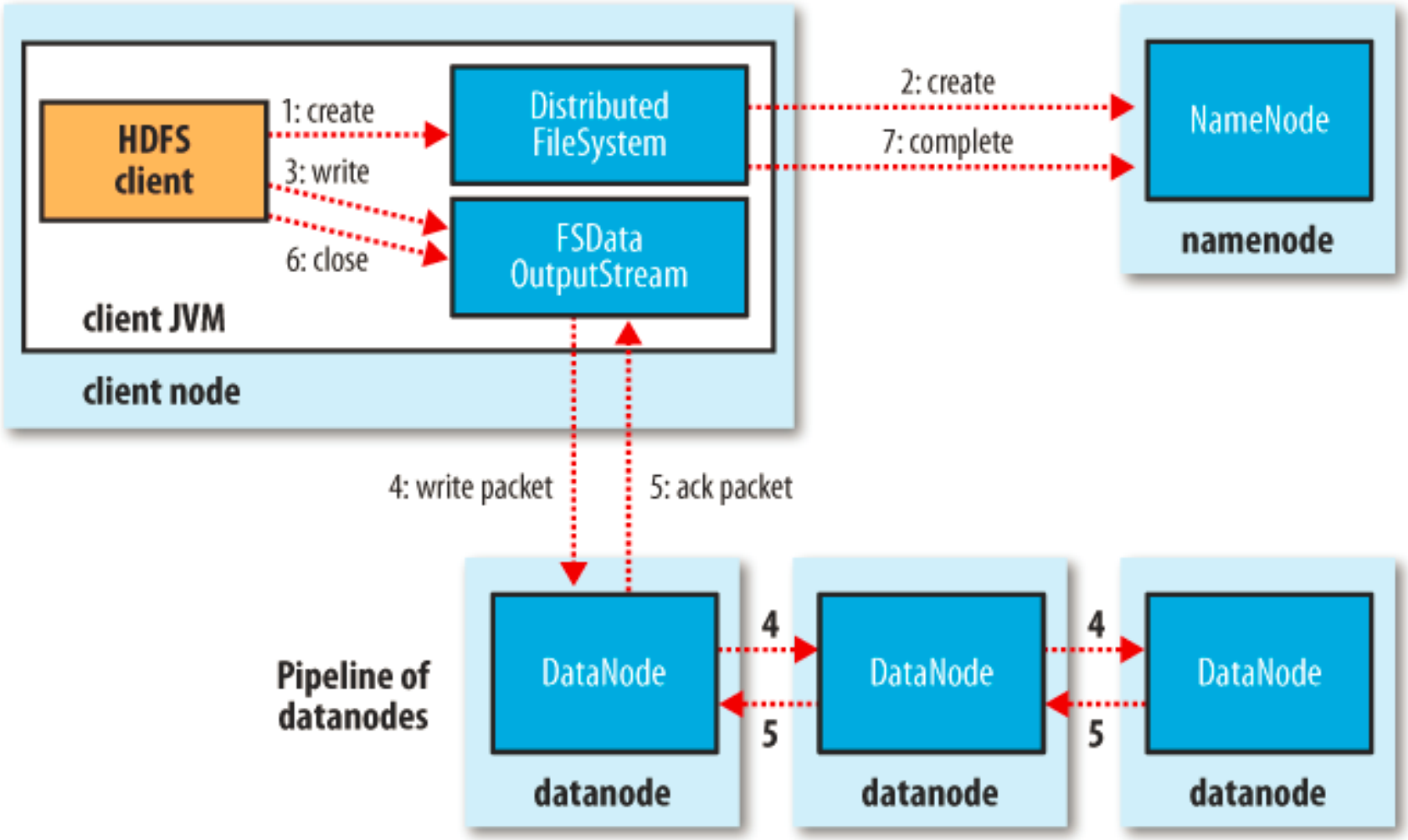
HDFS通讯图



读取文件

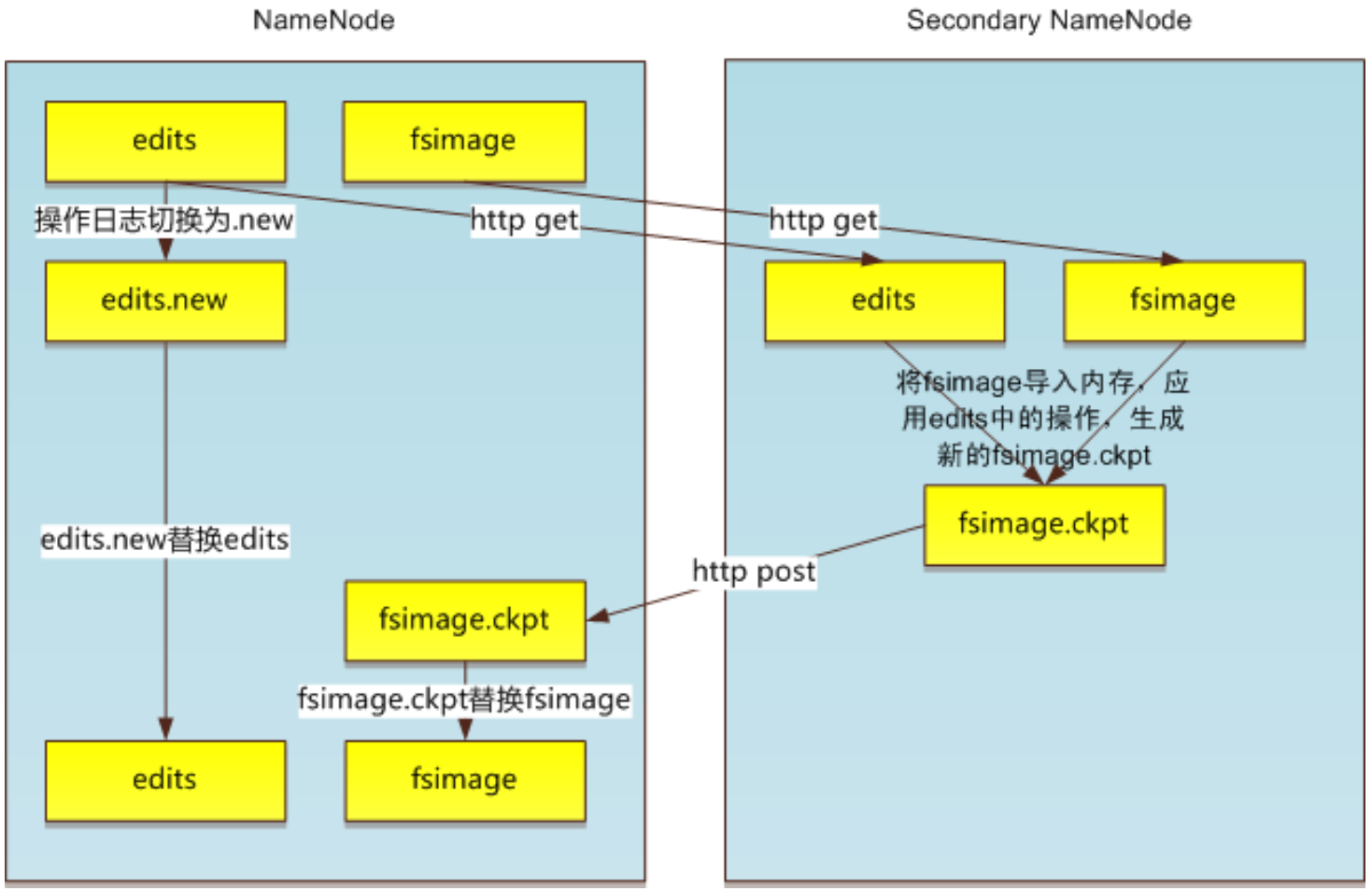


写入文件



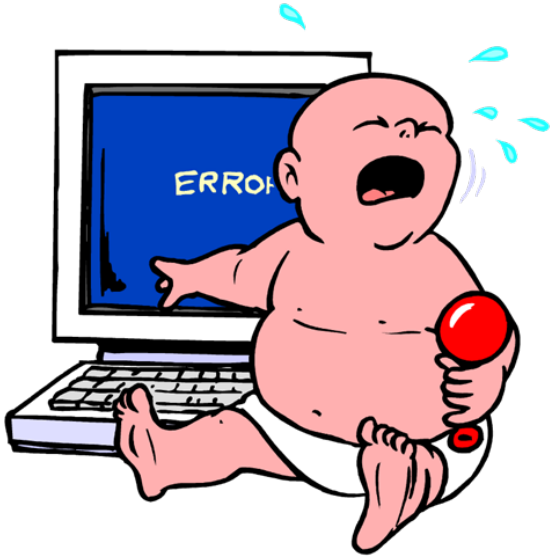
fsimage

fsimage包含文件的目录结构和分块信息，而数据块的位置信息则是通过动态汇总过来的，仅仅存活在内存数据结构中，每一个TaskTracker启动后，都会向NameNode发送注册消息，将其上数据块的状况都告知于NameNode。



HDFS故障恢复

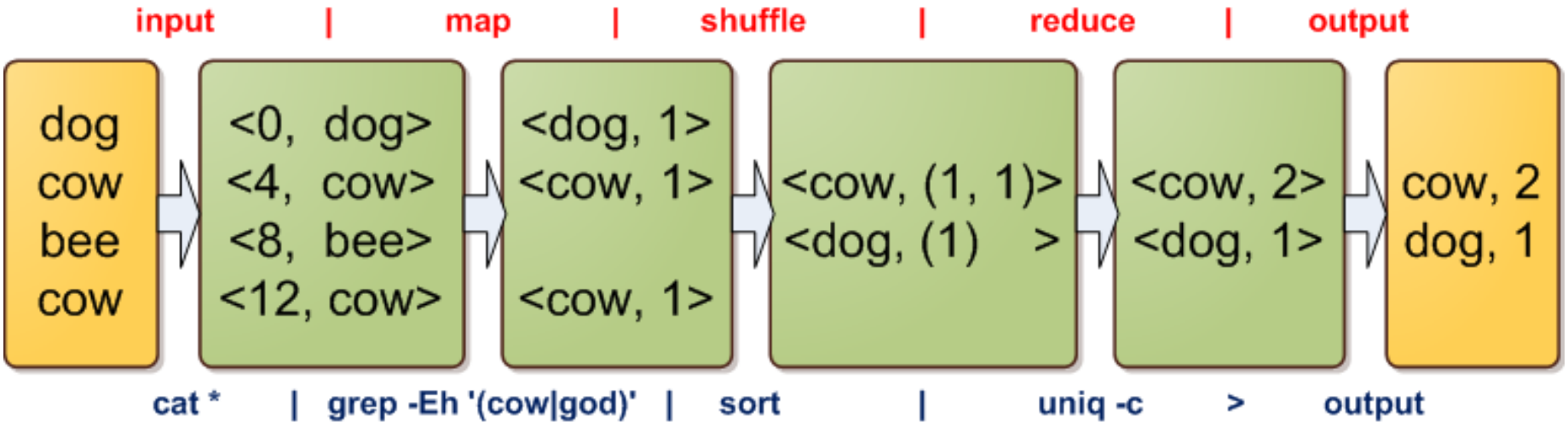
- DFSClient 崩溃？
  - 租约：当客户端需要占用某个文件时，与NameNode签订的一个短期合同
  - 超过期限没有续约，则终止租约，避免资源被长期霸占
- DataNode 崩溃?
  - 客户端读取另外一个副本
  - 后台负责副本的均衡和复制
- NameNode 崩溃?
  - 哎呀！需要人工干预
  - 宕机期间整个集群都没反应！
  - Secondary NameNode可以代替NameNode
  - 但可能会导致部分Editlog的丢失，无法100%恢复



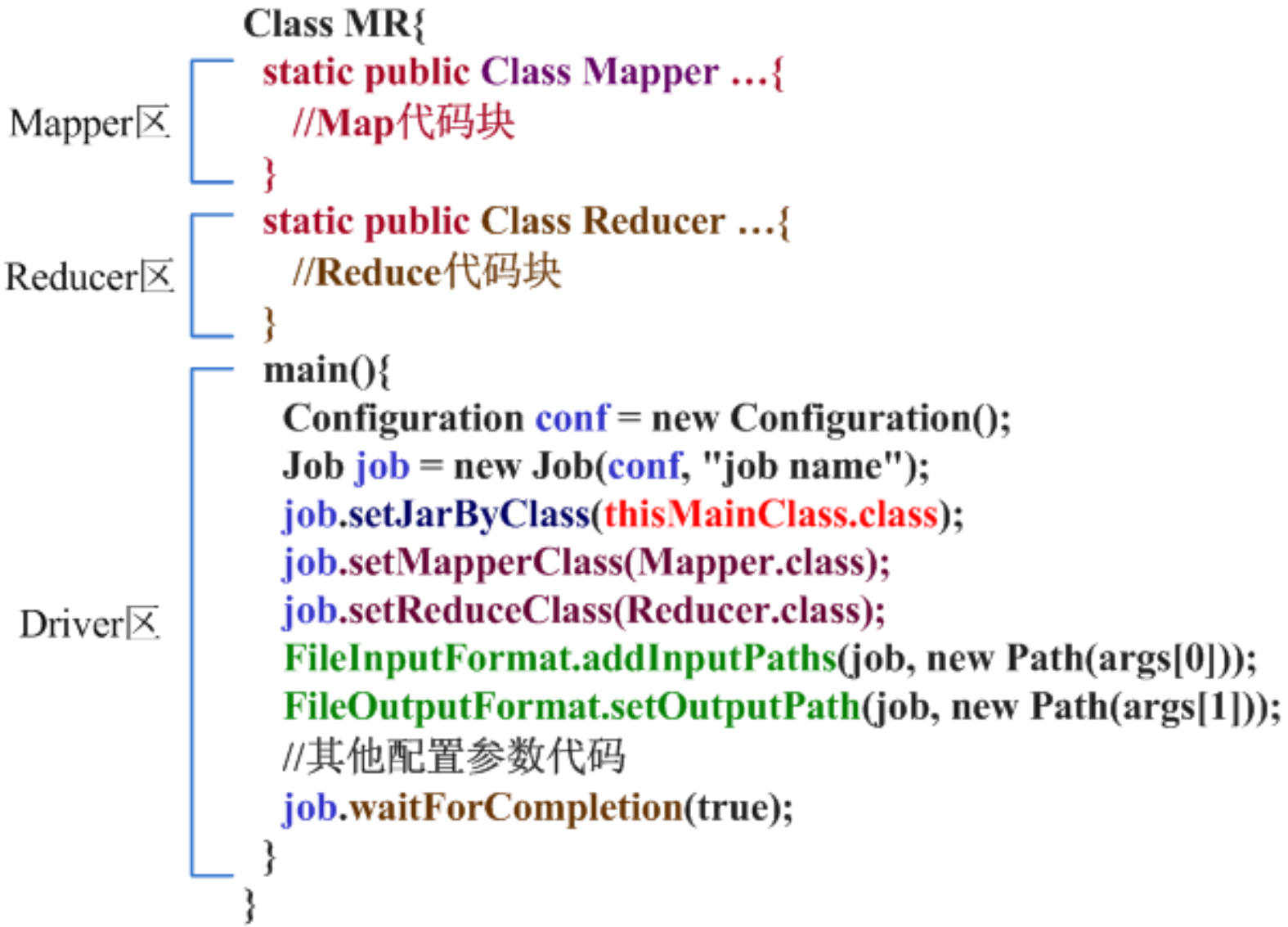
什么是MapReduce编程模型

- Map(映射)：对一些独立元素组成的列表的每一个元素进行指定的操作，可以高度并行。
- Reduce(化简)：对一个列表的元素进行合并。
- 一个简单的MapReduce程序只需要指定map()、reduce()、输入和输出，剩下的事由框架帮你搞定。

MapReduce举例



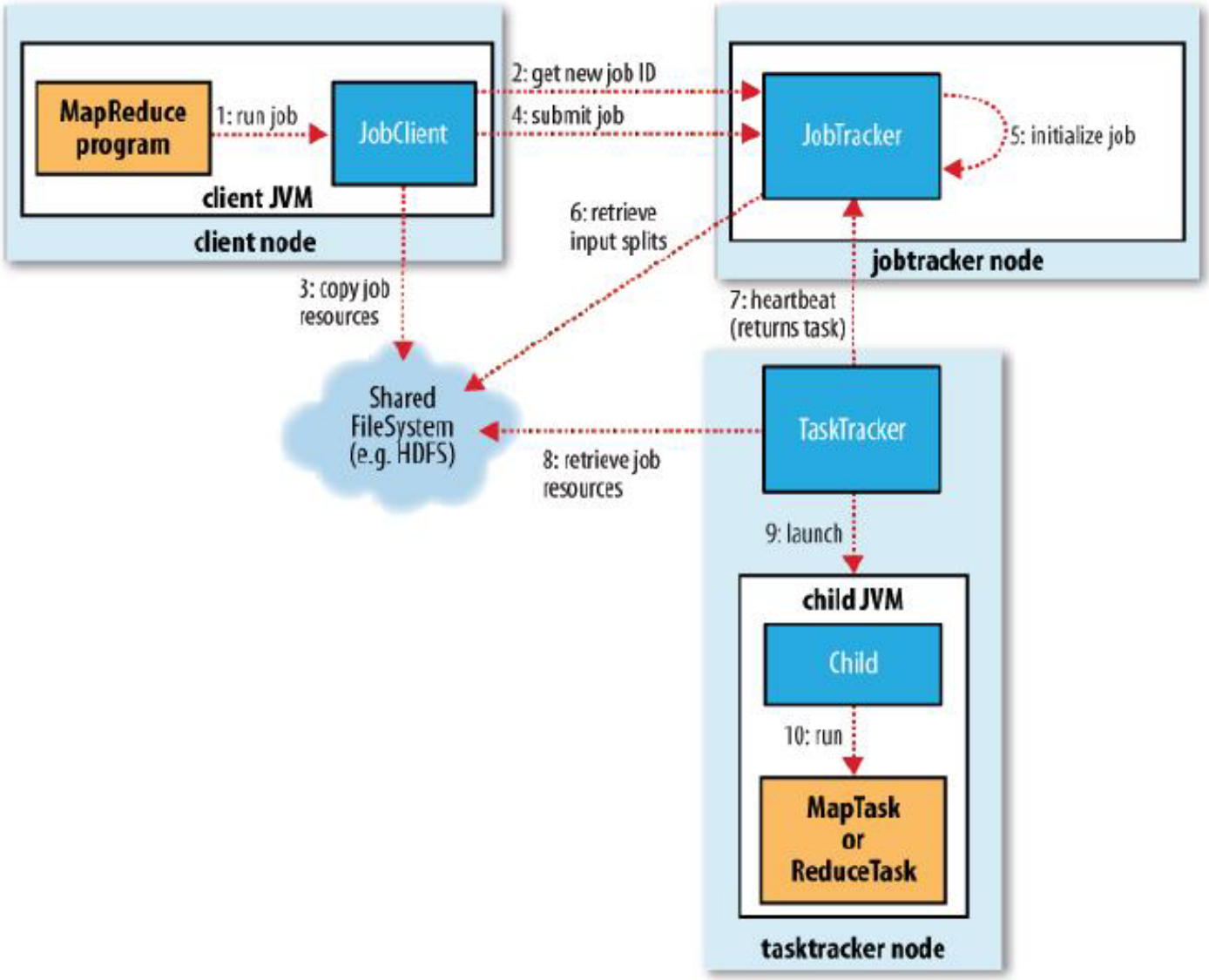
代码举例



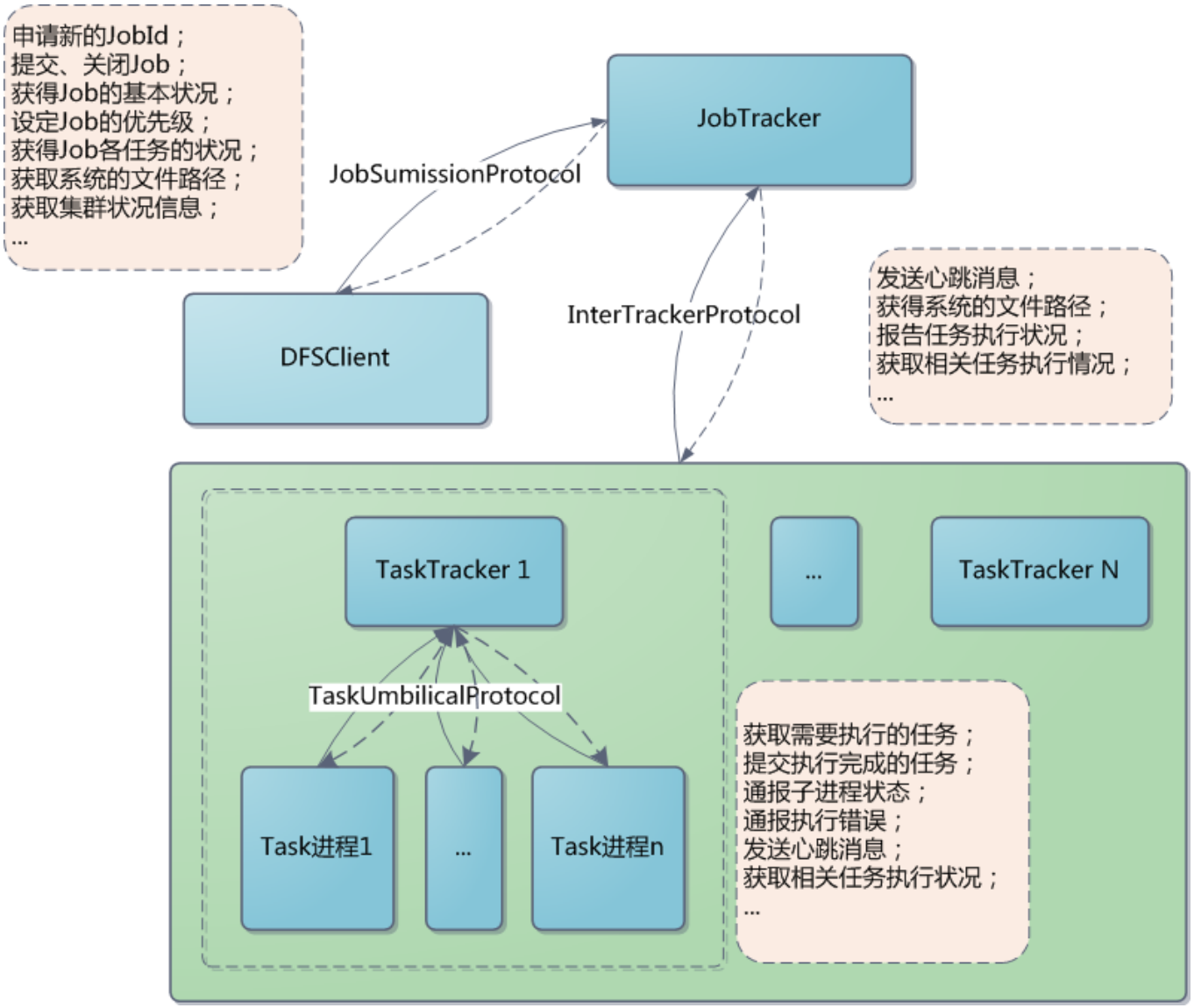
MapReduce术语

Hadoop术语	Google术语	说明
Job	Job	用户的每一个计算请求，称为一个作业。
JobTracker	Master	用户提交作业的服务器，同时，它还负责各个作业任务的分配，管理所有的任务服务器。
Task	Task	每一个作业，都需要拆分开了，交由多个服务器来完成，拆分出来的执行单位，就称为任务。
TaskTracker	Worker	任劳任怨的工蜂，负责执行具体的任务。
Speculative Task	Backup Task	每一个任务，都有可能执行失败或者缓慢，为了降低为此付出的代价，系统会未雨绸缪的实现在另外的任务服务器上执行同样一个任务，这就是备份任务，或叫推测性任务。

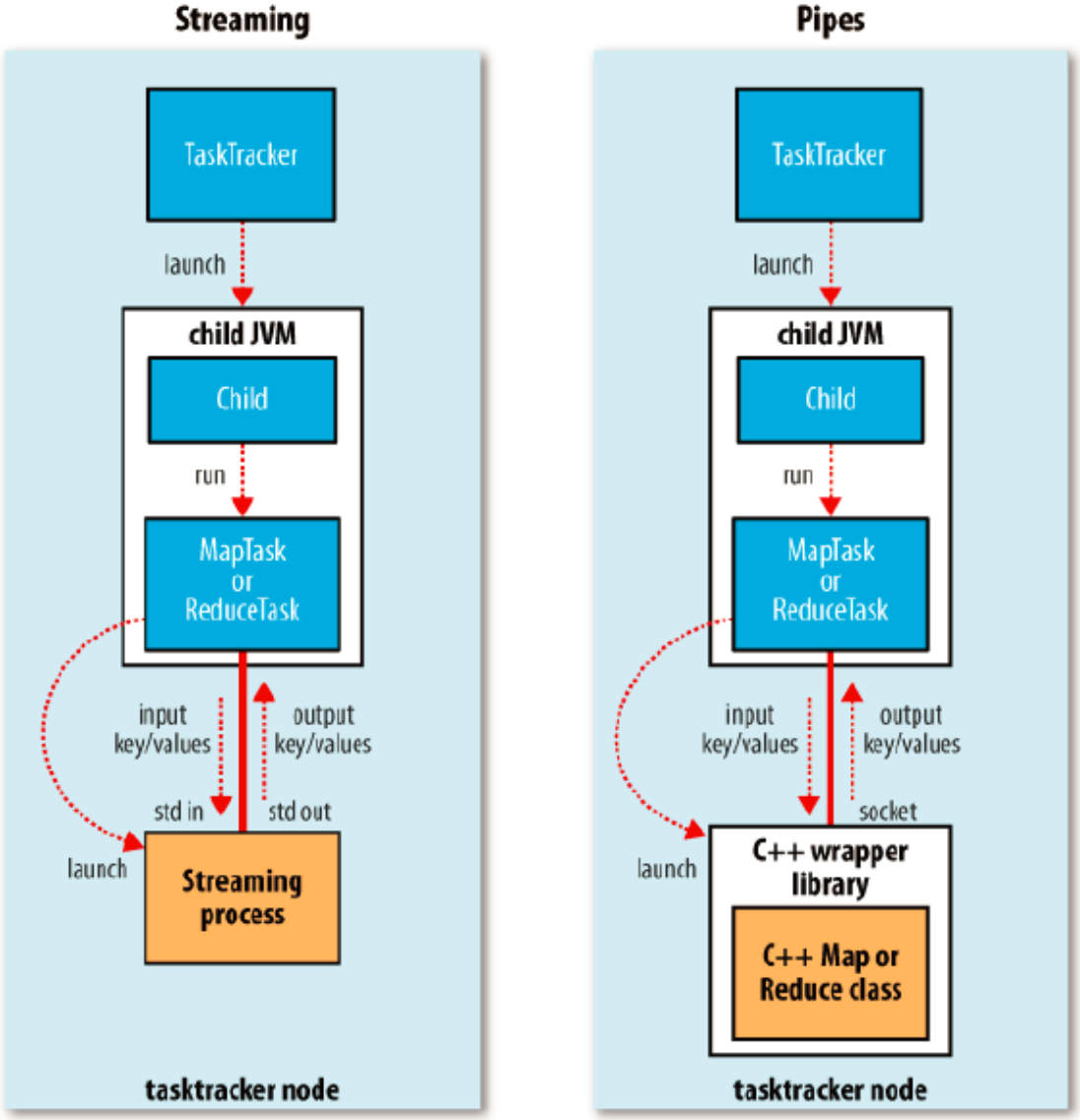
MapReduce流程图



MapReduce通讯图

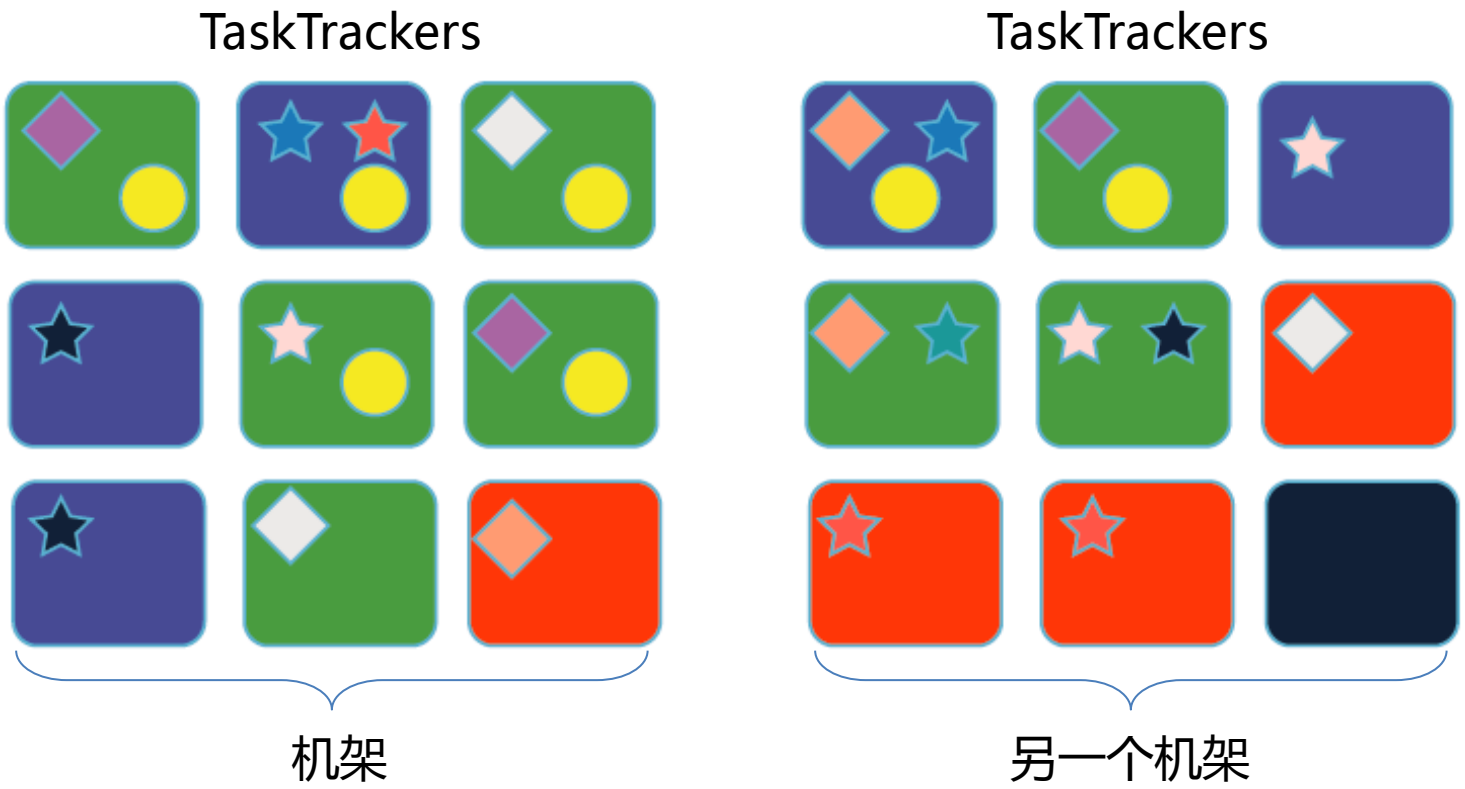


流和管道

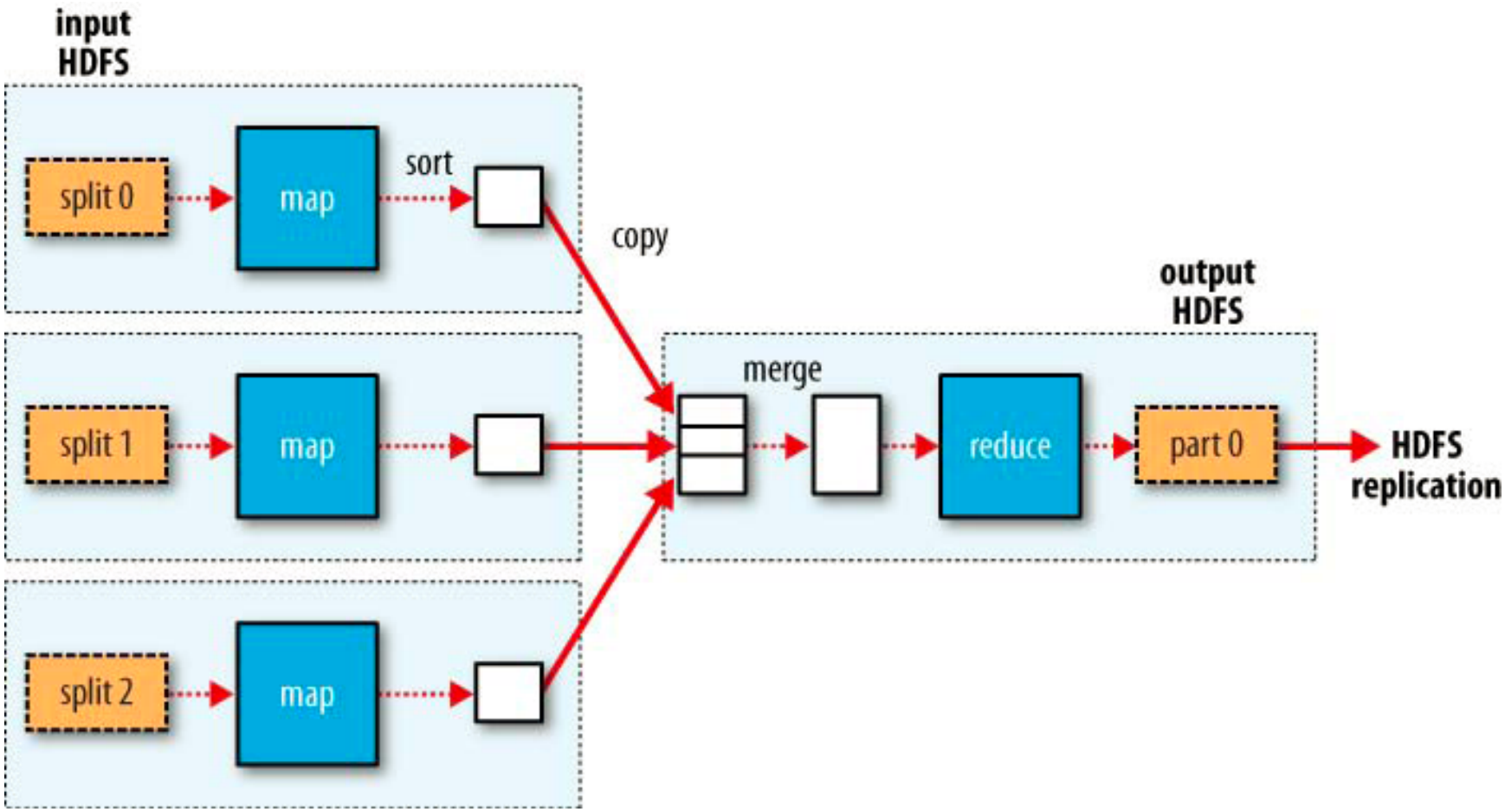


Task分配原则

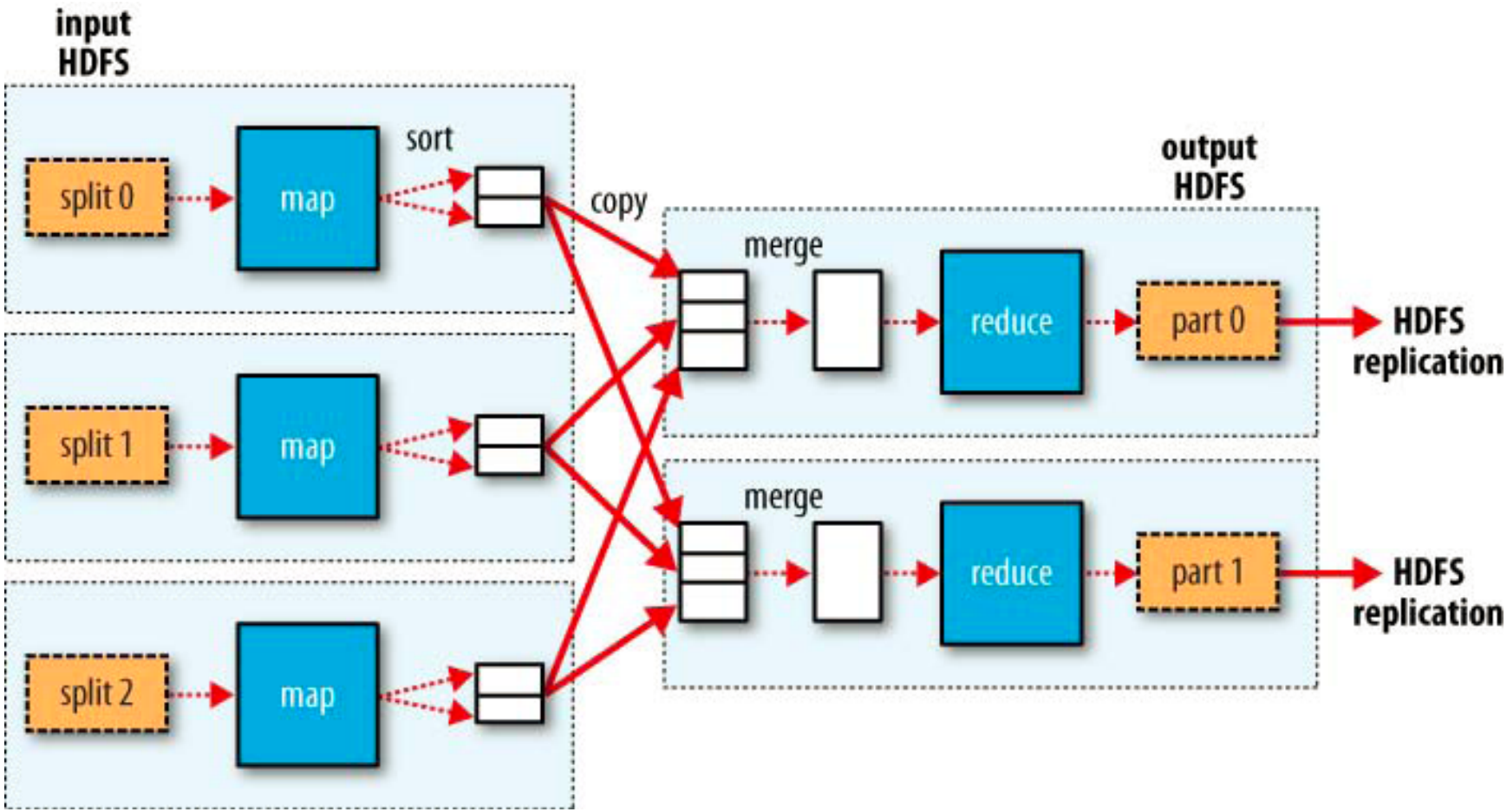
- TaskTrackers和DataNodes的部署是一致的
- 计算尽可能接近数据



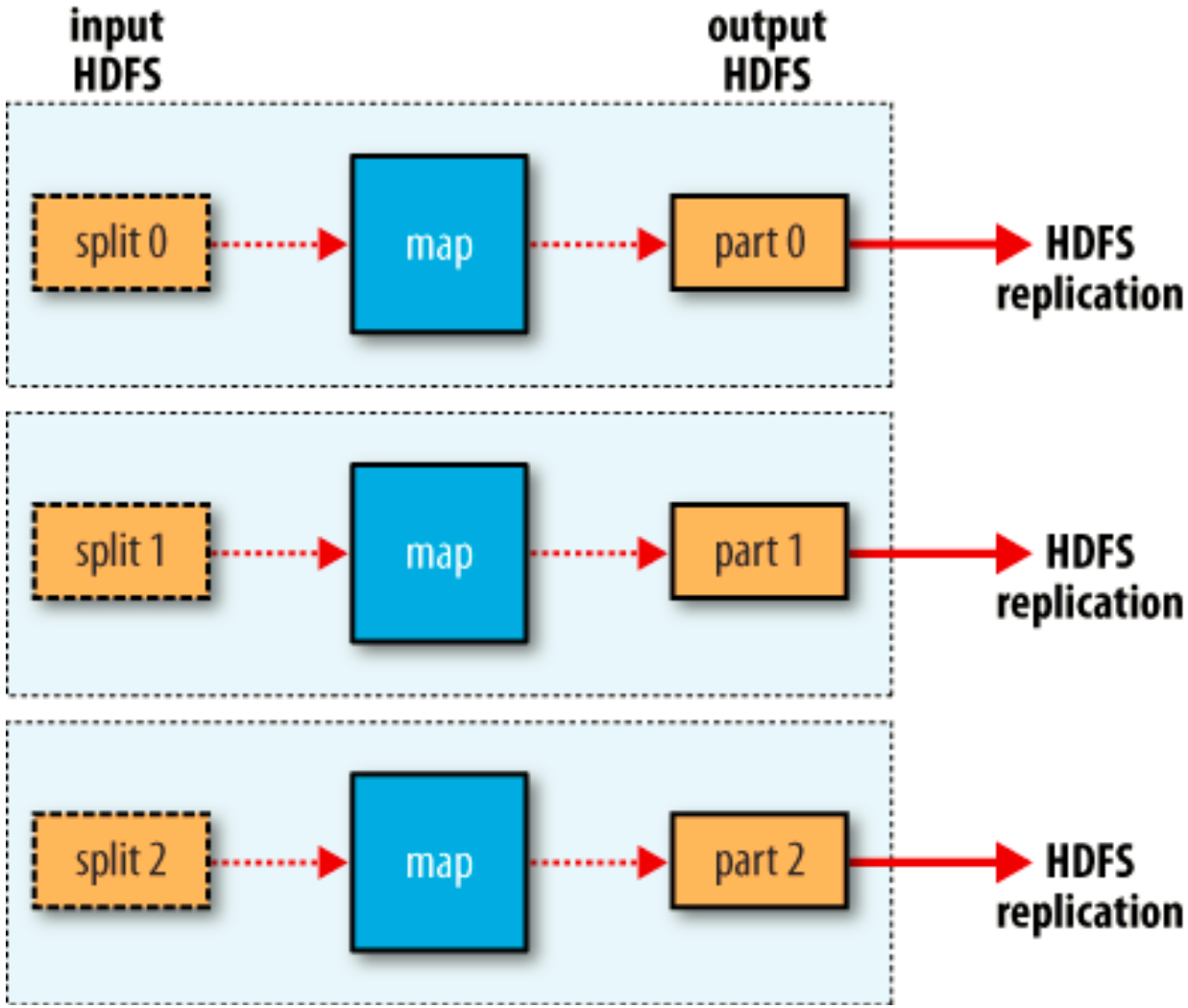
只有一个reduce task的数据流图



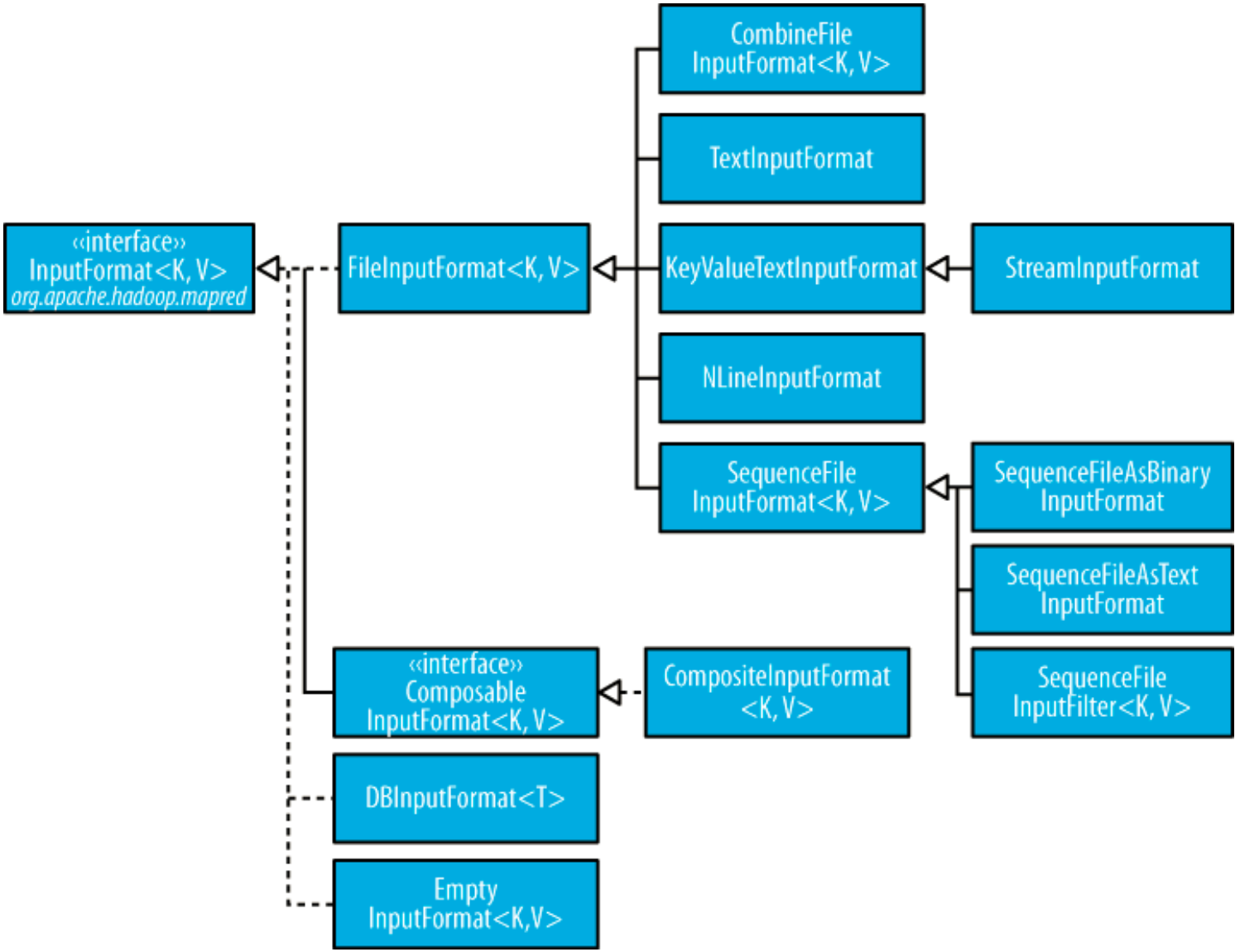
有N个reduce task的数据流图



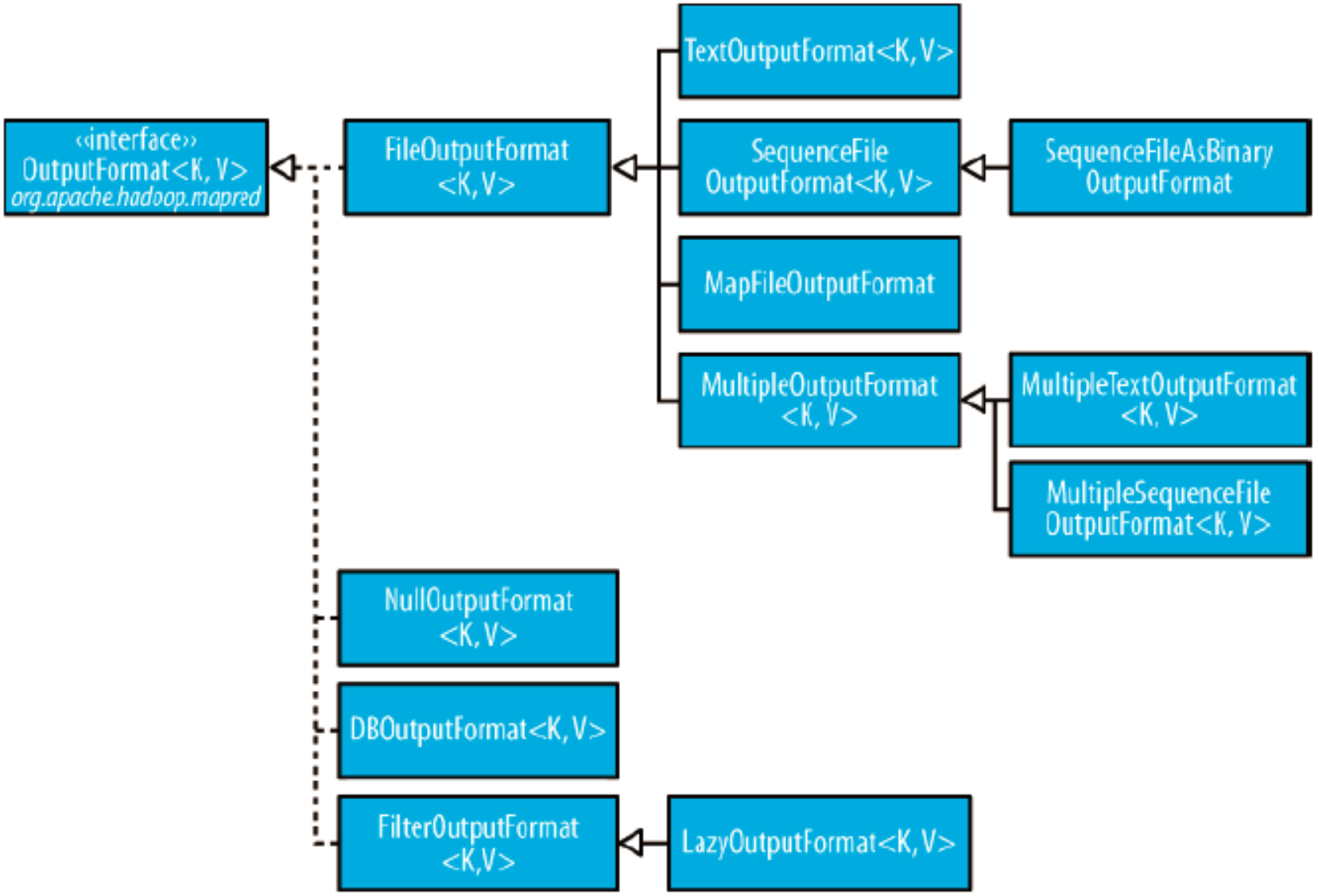
可以只有map task，没有reduce task



InputFormat

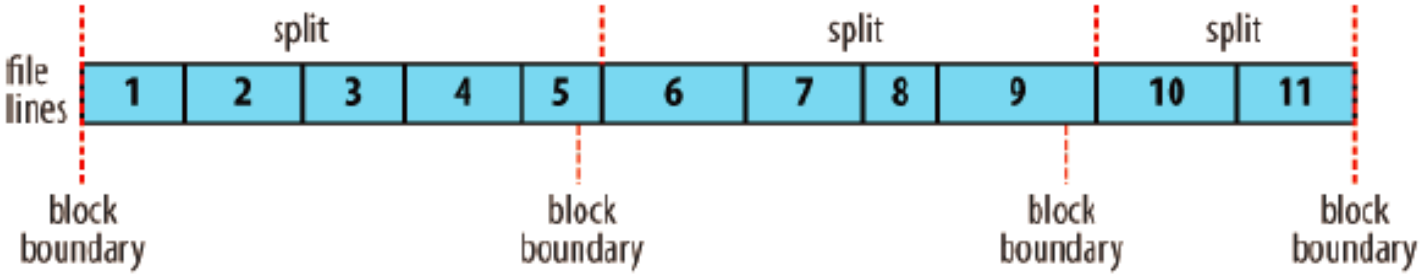


OutputFormat



split & block

- split是对record（即key-value pair）在逻辑上的切分，而HDFS中的block是对输入数据的物理切分。当两者一致时，很高效，但是实际中往往不一致。record可能会跨越block的边界。

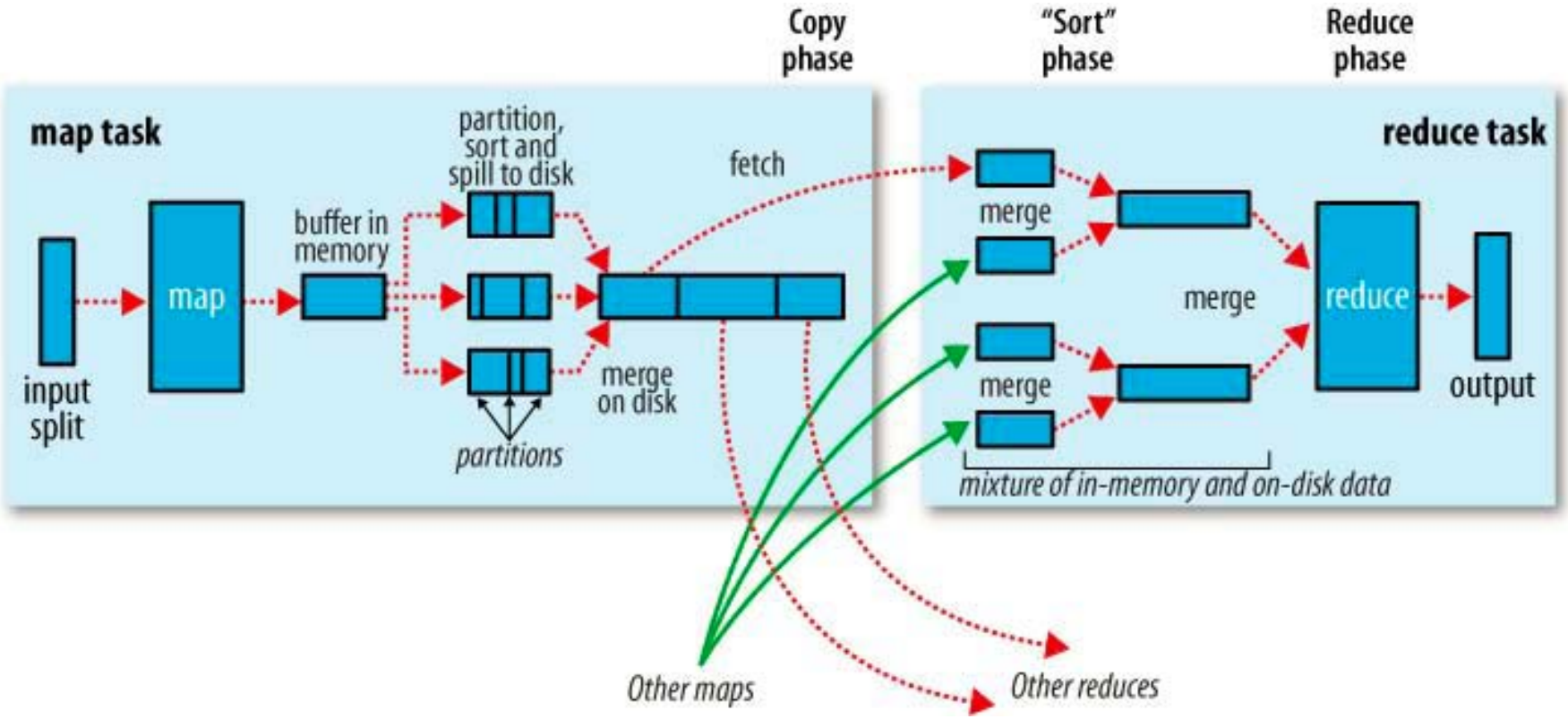


- split的大小选择
  - split不该太大（失去parallel性）
  - 也不该太小（额外的开销占比过大）
  - 与HDFS中的一个block的大小相同较为合适（block默认为64MB）

MapReduce核心功能

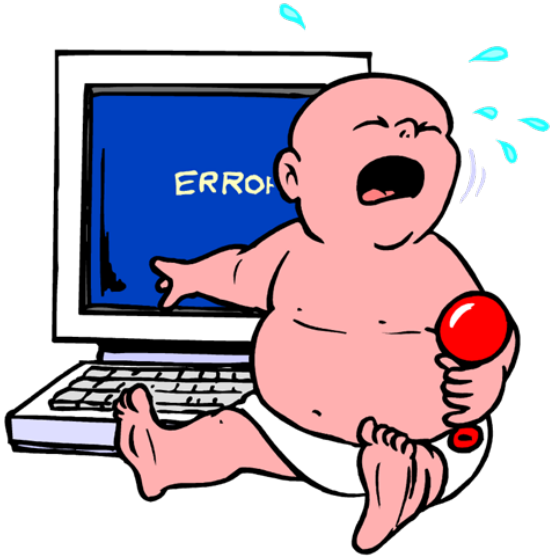
主功能	子功能	说明
Mapper		将输入键值对(key/value pair)映射到一组中间格式的键值对集合。
Combiner		对单个map的输出进行归约，减少map和reduce之间的数据传输，是一个优化的方法。
Partitioner		Partitioner负责控制map输出结果key的分割。通常使用的是Hash函数。分区的数目与一个作业的reduce任务的数目是一样的。因此，它控制将中间过程的key（也就是这条记录）应该发送给m个reduce任务中的哪一个来进行reduce操作。
Reducer		将与一个key关联的一组中间数值集归约（reduce）为一个更小的数值集。
	Shuffle	Reducer的输入就是Mapper已经排好序的输出。在这个阶段，框架通过HTTP为每个Reducer获得所有Mapper输出中与之相关的分块。
	Sort	这个阶段，框架将按照key的值对Reducer的输入进行分组 （因为不同mapper的输出中可能会有相同的key）。
	Secondary Sort	如果需要中间过程对key的分组规则和reduce前对key的分组规则不同，可进行二次排序。
	Reduce	为已分组的输入数据中的每个<key, (list of values)>对调用一次reduce()
Reporter		用于Map/Reduce应用程序报告进度，设定应用级别的状态消息，更新Counters（计数器）的机制。

shuffle & sort



MapReduce故障恢复

- Client崩溃？
  - 挂了就挂了，无伤大雅
- Task失败
  - 本机重试
  - 向JobTracker报告错误
  - 在别的TaskTracker中重试
- TaskTracker 崩溃?
  - 由JobTracker将其任务移交给别的TaskTracker
- JobTracker 崩溃?
  - 哎呀！需要人工干预
  - 正在运行的Job需要重新执行



谁在用Hadoop?

UC优视

82+PB, 25k+machines(2009)

12+PB , 10000+ cores,
15TB new data per day

~1TB per day, ~80 nodes

9+PB, 1100+nodes

20+PB, 2000+nodes,
10TB new data per day

Thank You!

uc.cn