北京交通大学

基于大数据的智能推荐系统设计与实现

Hadoop

张迪
dizhang@bjtu.edu.cn

Security

Apache Ranger

Meta Data Management

Apache Atlas

Data Format

Parquet, Avro, ORC, Arrow

Coordinate & Management

Zookeeper

Ambari

In-Memory Processing

Apache Ignite

Apache Spark

Stream Processing

Flink

Storm

kafka

SQL Over Hadoop

HIVE

APACHE DRILL

Impala

NoSQL Database

APACHE HBASE

Search Engine

Apache Solr

Data Piping

nifi

FLUME

sqoop

Machine Learning

MADLib

mahout

Apache Spark MLlib

Scripting

Scheduler

OOZIE

Airflow

Resource Management

hadoop YARN

MESOS

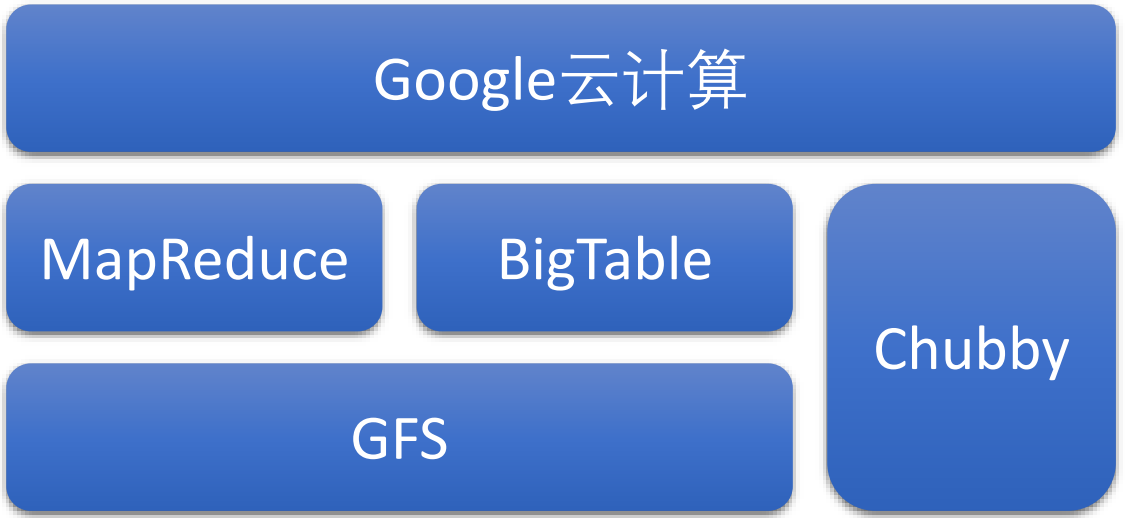
Storage

hadoop HDFS

ALLUXIO

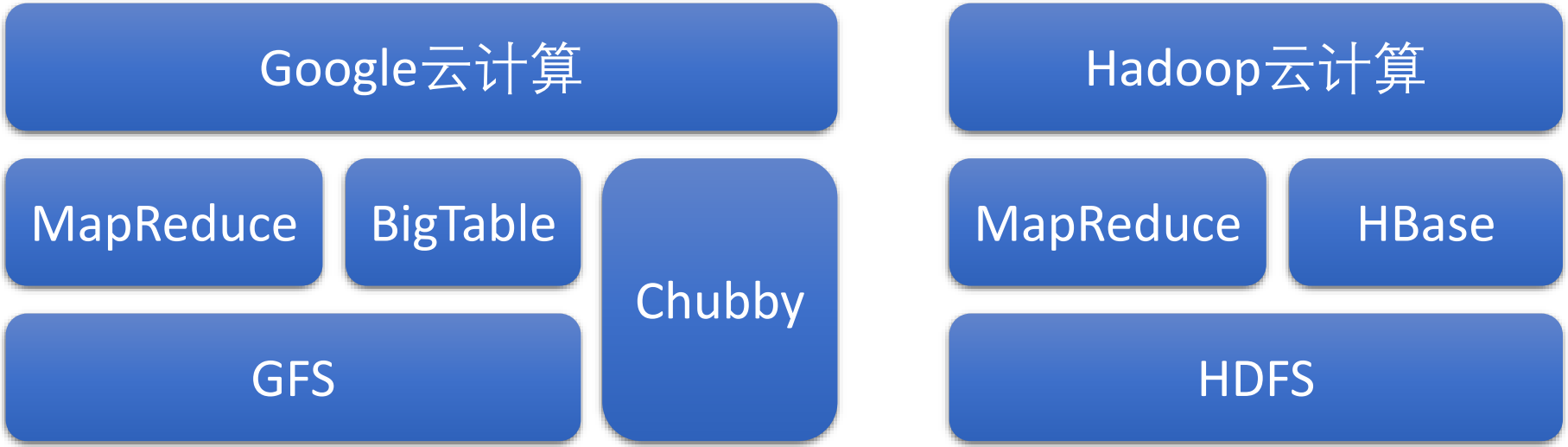
Google的云计算平台

- Google的三驾马车
  - GFS (SOSP'03)：分布式文件系统
  - MapReduce (OSDI'04)：分布式并行计算编程模型
  - Bigtable (OSDI'06)：分布式结构化数据存储系统



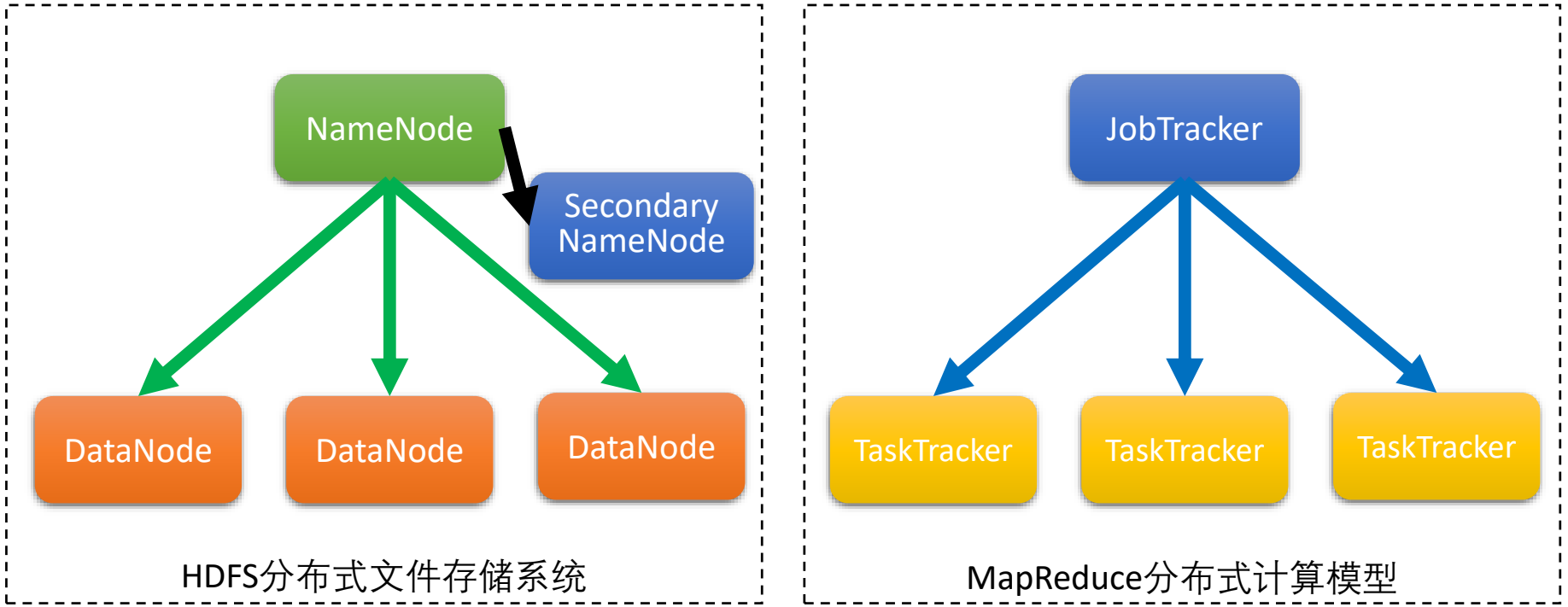
Hadoop

- 由Lucene和Nutch的作者Doug Cutting在2005年发起
  - 分布式文件系统：HDFS→GFS
  - 并行计算模型：MapReduce→MapReduce
  - 结构化数据库：HBase→BigTable



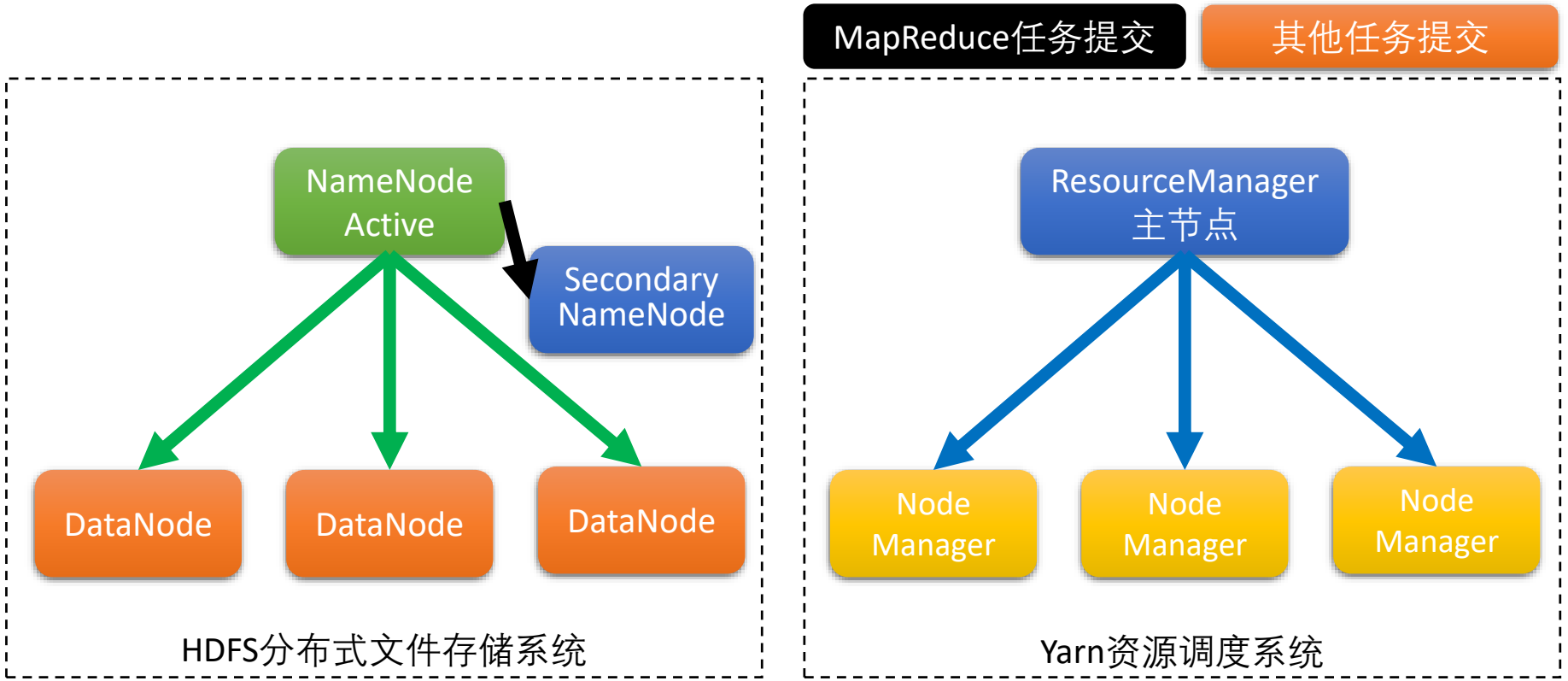
Hadoop架构模型

- 1.x版本



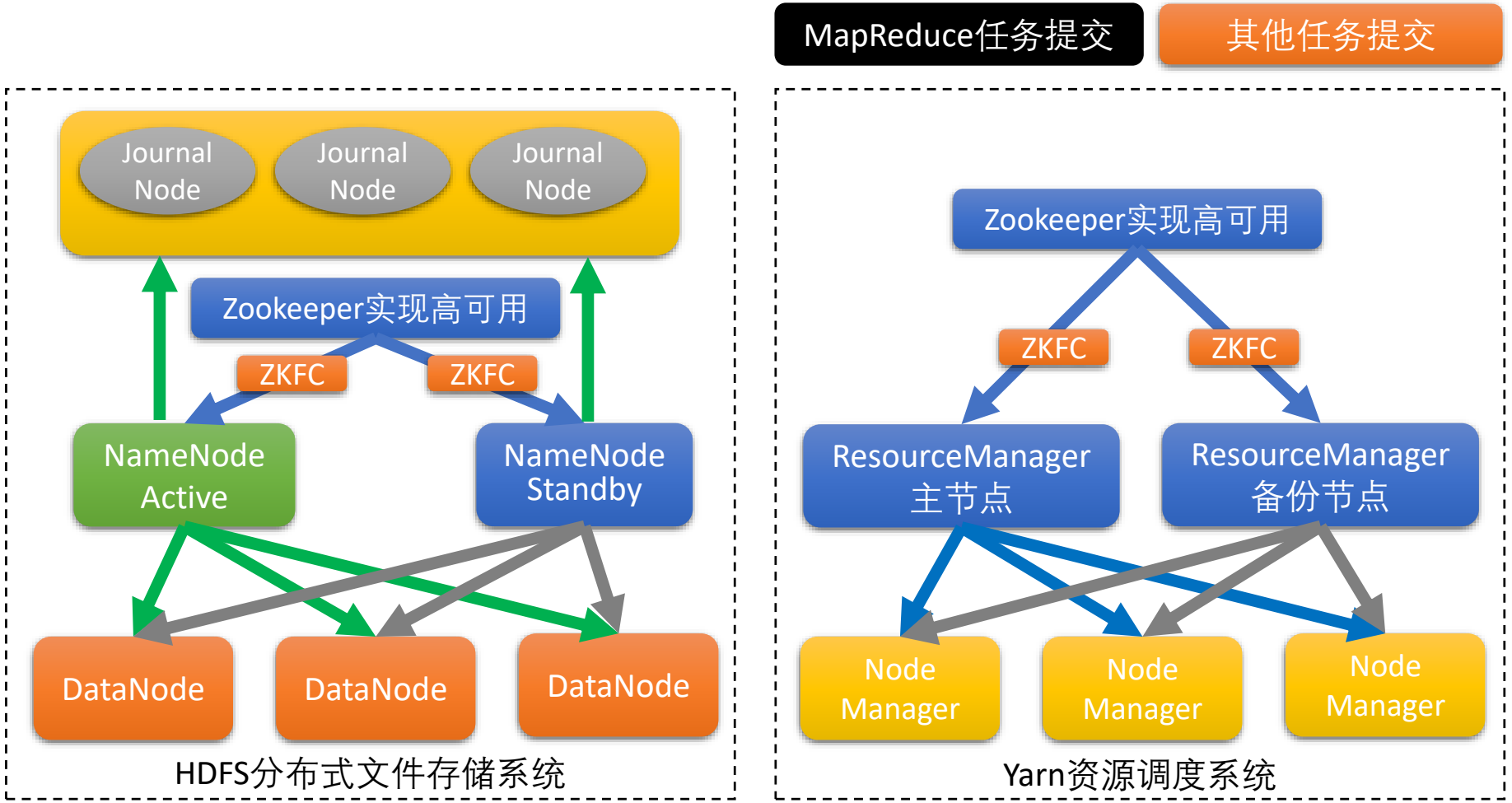
Hadoop架构模型

- 2.x版本

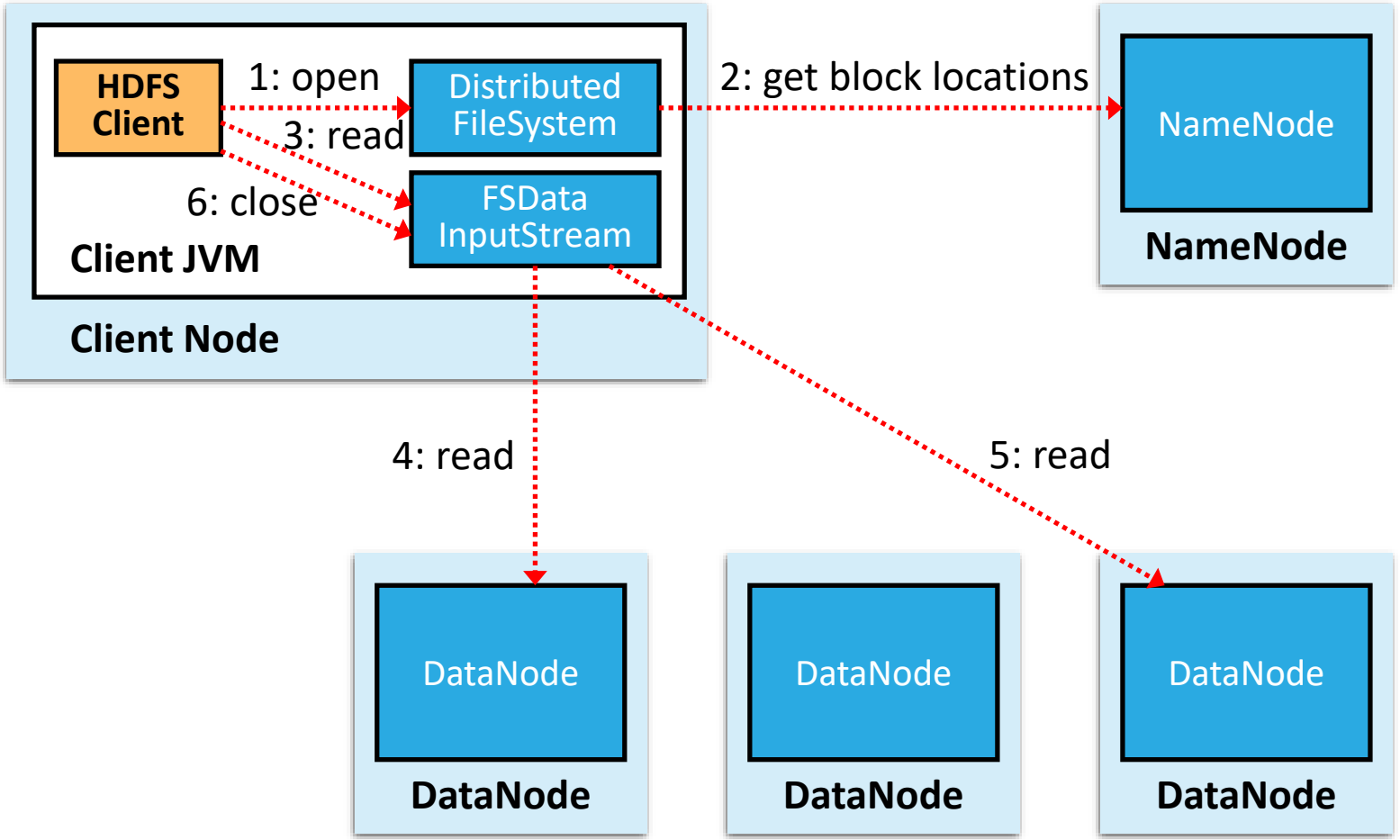


Hadoop架构模型

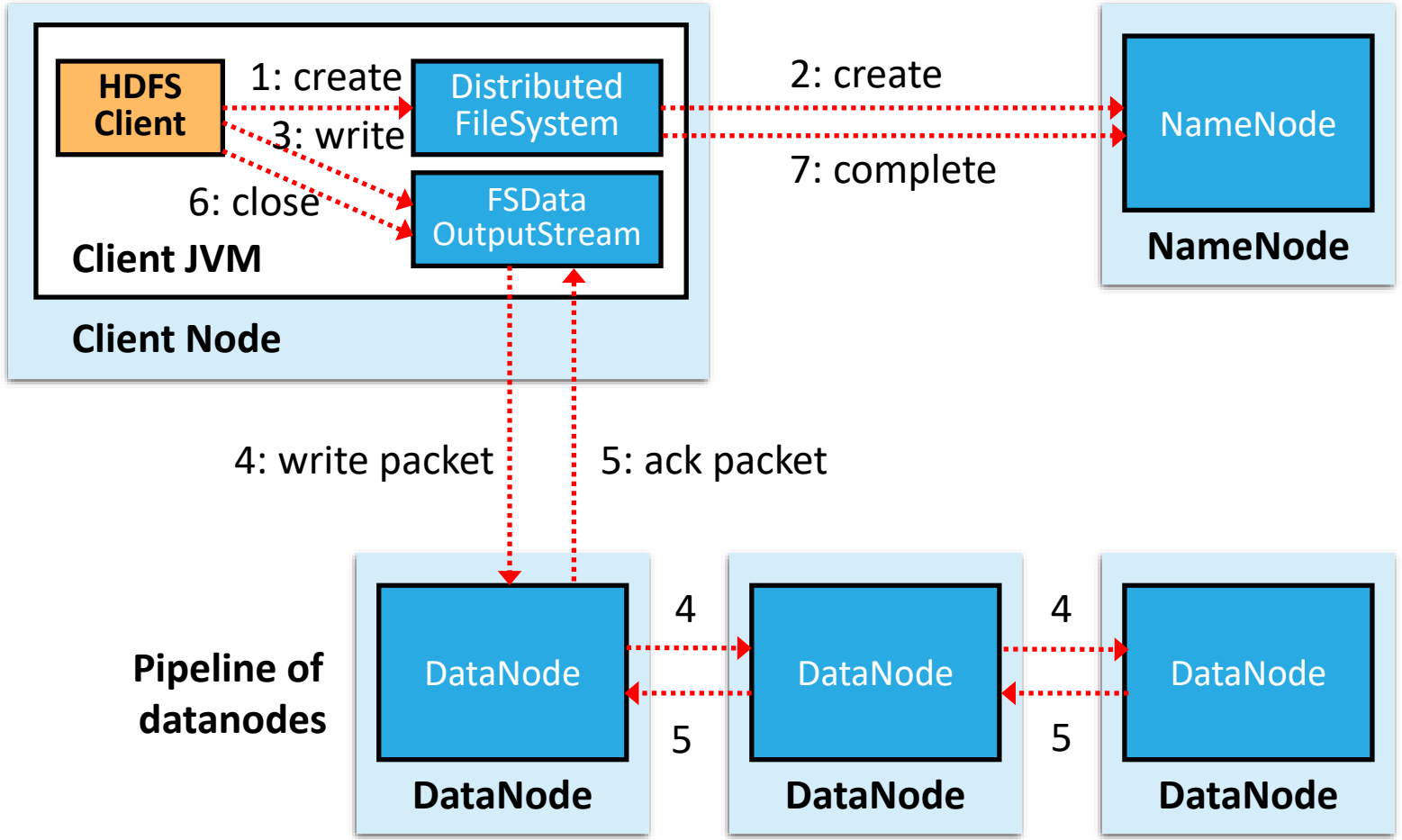
- 2.x版本，高可用



HDFS如何读取文件


HDFS Client
1: open
Distributed FileSystem
2: get block locations
NameNode
3: read
6: close
FSData InputStream
Client JVM
Client Node
NameNode
4: read
5: read
DataNode
DataNode
DataNode
DataNode
DataNode
DataNode

HDFS如何写入文件


HDFS Client
1: create
3: write
6: close
Distributed FileSystem
FSData OutputStream
Client JVM
Client Node
2: create
7: complete
NameNode
NameNode
4: write packet
5: ack packet
Pipeline of datanodes
DataNode
4
5
DataNode
4
5
DataNode
DataNode
DataNode
DataNode

什么是MapReduce？

- MapReduce是一个分布式的并行计算编程模型
  - 极大地简化了分布式并行开发，隐藏了复杂的分布式开发细节，将运行于大规模集群的并行计算抽象为两个函数，也即Map（映射）和Reduce（规约）
- MapReduce的编程思想
  - 分而治之

MapReduce可解决哪些问题

- MapReduce待处理的数据集可以分解成许多小的数据集，而且每个小数据可以完全并行的处理
- 可以解决的基本算法
  - 分布式排序
  - 词频统计
  - 文档倒排索引等
- 可解决的复杂算法
  - Web搜索引擎：爬取、网页排序和搜索算法
  - 数据/文本统计分析
  - 图算法、机器学习和数据挖掘等

MapReduce的设计思想

- 函数式编程语言Lisp
  - Lisp是一种应用于人工智能处理的函数式语言
  - 定义了map和reduce操作
- MapReduce定义了Map和Reduce两个抽象的操作
  - Map：对一组数据元素进行某种重复式的处理
  - Reduce：对Map的中间结果进行某种进一步的结果整理

MapReduce之Map

- map: (k1, v1) →list(k2, v2)
- 输入
  - 键值对(k1, v1)表示的数据
- 处理
  - 文档数据记录将以“键值对”形式传入map函数，map函数根据需求处理这一键值对，并以另一种键值对形式输出一组键值对中间结果list(k2, v2)
- 输出：键值对(k2, v2)表示的一组中间数据，也即list(k2, v2)

MapReduce之Reduce

- reduce: (k2, list(v2)) → list(k3, v3)
- 输入
  - 由map输出的一组键值对list(k2, v2)将被进行合并处理，同样主键k2下的不同数据合并到一个list(v2)中，所以reduce的输入为(k2, list(v2))
- 处理
  - 对传入的中间结果列表数据进行某种处理，并产生最终的某种形式的结果输出list(k3, v3)
- 输出： list(k3, v3)

基于Map和Reduce的并行计算

- 各个map函数对所划分的数据并行处理，从不同的输入数据产生不同的中间结果输出
- 各个reduce并行计算，各自负责处理不同的中间结果数据集合
- 进行reduce之前，须等待所有的map函数完成，并且进入reduce之前会对map的中间结果进行整理，保证map结果发送到reduce

函数	输入	输出
Map	(k1, v1)	List(k2, v2)
Reduce	(k2, List(v2))	List(k3, v3)

MapReduce小例子

- 现在有一组包含用户名和密码的数据，要求统计出使用次数大于1的密码。
- 数据格式：“用户名，密码”
  - zhang, 123456
  - wang, qazxsw
  - liu, 123456
  - meng, xxx123
  - hunan, qazxsw
  - chin, qazxsw
  - feifei, 1008xyz

期望得到的统计结果：

12345=2

qazxsw=3

MapReduce小例子之Map

map: (k1, v1) →list(k2, v2)

- Map的输入(k1, v1)默认是(文件行号, 文件行)
  - (0, ”zhang, 123456”), (1, “wang, qazxsw”), (2, “liu, 123456”), …
- 然后，需要在输入的value参数v1中，提取出密码为key k2，value是1（次数）的输出，即list(k2, v2)为：
  - (123456, 1), (qazxsw, 1), (123456, 1), (xxx123, 1), (qazxsw, 1), (qazxsw, 1), (1008xyz, 1)
- 输出被系统自动归并为(k2, list(v2))，也即：
  - (123456, (1, 1)), (qazxsw, (1, 1, 1)), (xxx123, (1)), (1008xyz, (1))
  - 这个(k2, list(v2))接着作为reduce的输入

MapReduce小例子之Reduce

reduce: (k2, list(v2)) → list(k3, v3)

- Reduce函数的输入是(k2, list(v2))
  - (123456, (1, 1)), (qazxsw, (1, 1, 1)), (xxx123, (1)), (1008xyz, (1))
- 我们可以把每个k2下的list(v2)元素相加，得到：
  - (123456, 2), (qazxsw, 3), (xxx123, 1), (1008xyz, 1)
- 然后只保留次数大于1的，得到最后的reduce结果list(k3, v3)：
  - (123456, 2), (qazxsw, 3)
  - 这里k3与k2类型一致，v3与v2类型一致

Thank you!

Q & A