HZ BOOKS
华章IT

Apress®

资深Spark专家撰写，涵盖高效使用Spark需要的所有技术

既全面讲解Spark核心概念和基本原理，又通过丰富的代码示例阐释Spark的实践应用

大数据技术丛书

Big Data Analytics with Spark

A Practitioner's Guide to Using Spark for Large Scale Data Analysis

Spark大数据分析

核心概念、技术及实践

[美] 穆罕默德·古勒（Mohammed Guller） 著

赵斌 马景 陈冠诚 译

机械工业出版社
China Machine Press

大数据技术丛书

Big Data Analytics with Spark

A Practitioner's Guide to Using Spark for Large Scale Data Analysis

Spark大数据分析

核心概念、技术及实践

[美] 穆罕默德·古勒（Mohammed Guller） 著

赵斌 马景 陈冠诚 译

机械工业出版社
China Machine Press

图书在版编目（CIP）数据

Spark 大数据分析：核心概念、技术及实践 /（美）穆罕默德·古勒（Mohammed Guller）著；赵斌，马景，陈冠诚译 . —北京：机械工业出版社，2017.5
（大数据技术丛书）
书名原文: Big Data Analytics with Spark: A Practitioner's Guide to Using Spark for Large Scale Data Analysis

ISBN 978-7-111-56561-1

I. S… II. ① 穆… ② 赵… ③ 马… ④ 陈… III. 数据处理软件 IV. TP274

中国版本图书馆 CIP 数据核字（2017）第 074151 号

本书版权登记号：图字：01-2016-8844

Spark 大数据分析：核心概念、技术及实践

出版发行：机械工业出版社（北京市西城区百万庄大街 22 号　邮政编码：100037）
责任编辑：谢晓芳　　责任校对：殷　虹
印　　刷：三河市宏图印务有限公司　　版　　次：2017 年 5 月第 1 版第 1 次印刷
开　　本：186mm × 240mm　1/16　　印　　张：16.5
书　　号：ISBN 978-7-111-56561-1　　定　　价：69.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换
客服热线：（010）88379426　88361066　　投稿热线：（010）88379604
购书热线：（010）68326294　88379649　68995259　　读者信箱：hzit@hzbook.com

华章IT
HZBOOKS | Information Technology

The Translator's Words 译 者 序

近几年来，大数据处理技术越来越受到大家的关注，各项新技术相继出现，Spark 就是其中的佼佼者。Spark 立足于内存计算，其基于 RDD 的计算模型可以兼顾 MapReduce 和迭代型计算。此外，它还可以拓展至流式计算、机器学习、图计算等领域。正是由于 Spark 通用、快速的特点，越来越多的公司采用 Spark 来搭建大数据计算平台，并在其上进行相应的开发。

本书是一本为 Spark 初学者准备的入门书。虽说是入门，但涵盖了 Spark 日常使用的方方面面。本书从基本的 Scala 语法讲起，进而介绍作为基石的 Spark Core。在此基础之上，再对 Spark 的各大组件 Streaming、SQL、MLlib、GraphX 进行了详细的介绍。最后，以 Spark 集群管理作为结尾。书中不仅给出了示例代码，还对 Spark 的核心概念和基本原理进行了较为全面的介绍，让读者不仅知其然且知其所以然。通过本书，读者可以快速上手 Spark，并且把 Spark 应用到实践中。

本书得以完成离不开各方的支持，感谢机械工业出版社各位编辑的大力支持和有益的建议。在翻译的过程中，来自家人和朋友的鼓励与支持也让我深受感动。

Spark 本身也在不断发展中，在本书翻译期间官方发布了 Spark 2.0，在提升性能的同时引入了若干重大更新。本人已尽量以注解的形式保证本书内容对 Spark 2.0 的可用性。然而，由于本人学识有限，Spark 涉及的知识面广，难免会有疏漏错误之处，恳请读者指正批评。

赵斌

前　言 *Preface*

本书是大数据和 Spark 方面的一本简明易懂的手册。它将助你学习如何用 Spark 来完成很多大数据分析任务。它覆盖了高效利用 Spark 所需要知道的一切内容。

购买本书的好处之一就是：帮你高效学习 Spark，节省你大量时间。本书所覆盖的主题在互联网上都可以找到，网上有很多关于 Spark 的博客、PPT 和视频。事实上，Spark 的资料浩如烟海，你可能需要在网络上不同地方花费数月来阅读关于 Spark 的点滴和碎片知识。本书提供了一个更好的选择：内容组织精妙，并以易懂的形式表现出来。

本书的内容和材料的组织基于我在不同的大数据相关会议上所组织的 Spark 研讨会。与会者对于内容和流程方面的积极反馈激励我写了这本书。

书和研讨会的区别之一在于后者具有交互性。然而，组织过几次 Spark 研讨会后，我了解到了人们普遍存在的问题，我把这些内容也收录在本书中。如果阅读本书时有问题，我鼓励你们通过 LinkedIn 或 Twitter 联系我。任何问题都可以问，不存在什么“愚蠢的问题”。

本书没有覆盖 Spark 的每一个细节，而是包含了高效使用 Spark 所需要知道的重要主题。我的目标是帮你建立起坚实的基础。一旦基础牢固，就可以轻松学习一项新技术的所有细节。另外，我希望保持本书尽可能简单。如果读完本书后发现 Spark 看起来也挺简单的，那我的目的也就达到了。

本书中的任何主题都不要求有先验知识。本书会一步步介绍关键概念，每一节建立在前一节的基础上。同样，每一章都是下一章的基石。如果当下不需要，你可以略过后面一些章节中讲解的不同的 Spark 库。不过我还是鼓励你阅读所有章节。即使可能和你当前的项目不相关，那些部分也可能会给你新的灵感。

通过本书你会学到很多 Spark 及其相关技术的知识。然而，要充分利用本书，建议亲自运行书中所展示的例子：用代码示例做实验。当你写代码并执行时，很多事情就变得更加清晰。如果你一边阅读一边练习并用示例来实验，当读完本书时，你将成为一名基础扎实的 Spark 开发者。

在我开发 Spark 应用时，我发现了一个有用的资源——Spark 官方 API 文档，其访问地址为 http://spark.apache.org/docs/latest/api/scala。初学者可能觉得它难以理解，不过一旦你学习了基本概念后，会发现它很有用。

另一个有用的资源是 Spark 邮件列表。Spark 社区很活跃、有用。不仅 Spark 开发者会回答问题，有经验的 Spark 用户也会志愿帮助新人。无论你遇到什么问题，很有可能 Spark 邮件列表中有人已经解决过这个问题了。

而且，也可以联系我，我很乐意倾听，欢迎反馈、建议和提问。

——Mohammed Guller

LinkedIn: www.linkedin.com/in/mohammedguller

Twitter: @MohammedGuller

致　谢 *Acknowledgements*

许多人都直接地或间接地为本书作出了贡献。如果没有他们的支持、鼓励与帮助，我是无法完成本书的编写的。我想借此机会向他们表示感谢。

首先，也是最重要的，我想要感谢我的妻子 Tarannum 和我的三个可爱的孩子 Sarah、Soha、Sohail。写书是一项艰巨的任务。在从事全职工作的同时写书意味着我无法花费太多的时间在我的家人身上。上班时间我忙于工作，晚上和周末我则全身投入到本书的写作上。我对我家人给予的全方位的支持和鼓励表示感谢。有时候，Soha 和 Sohail 会提出一些有意思的想法让我陪他们一起玩，但是在大部分时候，他们还是让我在本应该陪他们玩耍的时候专注于写书。

接下来，感谢 Matei Zaharia、Reynold Xin、Michael Armbrust、Tathagata Das、Patrick Wendell、Joseph Bradley、Xiangrui Meng、Joseph Gonzalez、Ankur Dave 以及其他 Spark 开发者。他们不仅创造出了一项卓越的技术，还持续快速改进它。没有他们的发明，本书将不会存在。

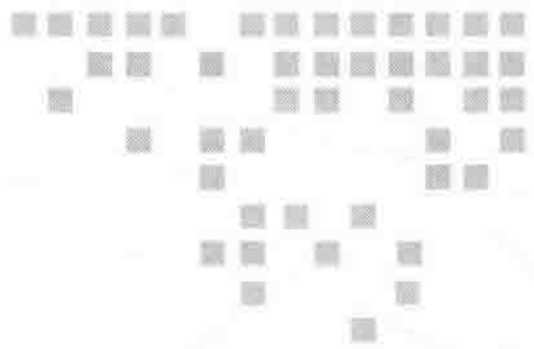
当我在 Glassbeam 公司提议使用 Spark 来解决当时困扰我们的一些问题时，Spark 还是一项新技术且少有人了解。我想要感谢工程副总裁 Ashok Agarwal 和首席执行官 Puneet Pandit 允许我使用 Spark。如果没有来自将 Spark 内置于产品中和日常使用的一手经验，要写出一本有关 Spark 的书是相当困难的。

接下来，我想感谢技术审校者 Sundar Rajan Raman 和 Heping Liu。他们认真检查了本书内容的准确性并运行了书中的例子以确保它们能正常运行，还提出了不少有帮助的建议。

最后，我想感谢 Apress 参与本书出版的工作人员 Chris Nelson、Jill Balzano、Kim Burton-Weisman、Celestin John Suresh、Nikhil Chinnari、Dhaneesh Kumar 等。Jill Balzano 协调了与本书出版相关的所有工作。作为一个编辑，Chris Nelson 为本书作出了卓越的贡献。我十分感谢他的建议与编辑，有了他的参与，本书变得更完美了。文字编辑 Kim Burton-Weisman 认真阅读了本书的每一句话以保证书写正确，同时也改正了不少书写错误。很荣幸能与 Apress 团队一起工作。

——Mohammed Guller

Contents 目　录



第 1 章 Chapter 1

大数据技术一览

我们正处在大数据时代。数据不仅是任何组织的命脉，而且在指数级增长。今天所产生的数据比过去几年所产生的数据大好几个数量级。挑战在于如何从数据中获取商业价值。这就是大数据相关技术想要解决的问题。因此，大数据已成为过去几年最热门的技术趋势之一。一些非常活跃的开源项目都与大数据有关，而且这类项目的数量在迅速增长。聚焦在大数据方向的创业公司在近年来呈爆发式增长。很多知名公司在大数据技术方面投入了大笔资金。

尽管“大数据”这个词很火，但是它的定义是比较模糊的。人们从不同方面来定义“大数据”。一种定义与数据容量相关，另一种则与数据的丰富度有关。有些人把大数据定义为传统标准下“过于大”的数据，而另一些人则把大数据定义为捕捉了所描绘实体更多细节的数据。前者的例子之一就是超过数拍字节（PB）或太字节（TB）大小的数据集，如果这样的数据存储在传统的关系数据库（RDBMS）表中，将会有数十亿行。后者的一个例子是有极宽行的数据集，这样的数据存储在 RDBMS 中，将会有数千列。另一种流行的大数据定义是由 3 个 V（volume、velocity 和 variety，即容量、速度和多样性）所表征的数据。我刚才讨论了容量。速度指的是数据以极快的速率产生，多样性则指的是数据可以是非结构化、半结构化或多结构的。

标准的关系数据库无法轻易处理大数据。这些数据库的核心技术在数十年前所设计，当时极少有组织拥有拍字节级甚至太字节级的数据。现在对一些组织来说，每天产生数太字节的数据也很正常。数据的容量和产生速度都呈爆发式增长。因此，迫切需要新的技术：能快速处理和分析大规模数据。

其他推动大数据技术的因素包括：可扩展性、高可用性和低成本下的容错性。长期以

来，处理和分析大数据集的技术被广泛研究并以专有商业产品的形式被使用。例如，MPP（大规模并行处理）数据库已经诞生有段时间了。MPP 数据库使用一种“无共享”架构，数据在集群的各个节点进行存储和处理。每一个节点有自己的 CPU、内存和硬盘，节点之间通过网络互联来通信。数据分割在集群的各个节点，而节点之间不存在竞争，所以每个节点可以并行处理数据。这种数据库的例子包括 Teradata、Netezza、Greenplum、ParAccel 和 Vertica。Teradata 发明于 20 世纪 70 年代末，在 20 世纪 90 年代前，它就能够处理太字节级别的数据了。但是，专有的 MPP 数据库非常昂贵，不是所有人能负担得起的。

本章介绍一些开源的大数据相关技术。本章涉及的技术看起来好像随意挑选的，实际上它们由共同的主题而连接：它们和 Spark 一起使用，或者 Spark 可以取代其中一些技术。当你开始使用 Spark 时，你可能会涉及这些技术。而且，熟悉这些技术会帮你更好地理解 Spark（这将在第 3 章介绍）。

1.1 Hadoop

Hadoop 是最早流行的开源大数据技术之一。这是一个可扩展、可容错的系统，用来处理跨越集群（包含多台商用服务器）的大数据集。它利用跨集群的可用资源，为大规模数据处理提供了一个简单的编程框架。Hadoop 受启发于 Google 发明的一个系统（用来给它的搜索产品创建反向索引）。Jeffrey Dean 和 Sanjay Ghemawat 在 2004 年发表的论文中描述了这个他们为 Google 而创造的系统。第一篇的标题为“MapReduce：大集群上简化的数据处理”，参见 research.google.com/archive/mapreduce.html；第二篇的标题为“Google 文件系统”，参见 research.google.com/archive/gfs.html。受启发于这些论文，Doug Cutting 和 Mike Cafarella 开发了一个开源的实现，就是后来的 Hadoop。

很多组织都用 Hadoop 替换掉昂贵的商业产品来处理大数据集。一个原因就是成本。Hadoop 是开源的，并可以运行在商用硬件的集群上。可以通过增加廉价的服务器来轻松地扩展。Hadoop 提供了高可用性和容错性，所以你不需要购买昂贵的硬件。另外，它对于特定类型的数据处理任务非常合适，比如对于大规模数据的批处理和 ETL（Extract、transform、load，提取、转换、加载）。

Hadoop 基于几个重要的概念。第一，使用商用服务器集群来同时存储和处理大量数据比使用高端的强劲服务器更便宜。换句话说，Hadoop 使用横向扩展（scale-out）架构，而不是纵向扩展（scale-up）架构。

第二，以软件形式来实现容错比通过硬件实现更便宜。容错服务器很贵，而 Hadoop 不依赖于容错服务器，它假设服务器会出错，并透明地处理服务器错误。应用开发者不需要操心处理硬件错误，那些繁杂的细节可以交给 Hadoop 来处理。

第三，通过网络把代码从一台计算机转到另一台比通过相同的网络移动大数据集更有效、更快速。举个例子，假设你有一个 100 台计算机组成的集群，每台计算机上有 1TB 的

数据。要处理这些数据，一个选择是：把数据转移到一台能够处理 100TB 数据的超级计算机。然而，转移 100TB 的数据将花费极长时间，即使是在高速网络上。另外，通过这种方式处理数据将需要非常昂贵的硬件。另一个选择是：把处理数据的代码转移到具有 100 个节点的集群中的每台计算机。这比第一种选择更快、更高效。而且，你不需要高端、昂贵的服务器。

第四，把核心数据处理逻辑和分布式计算逻辑分开的方式，使得编写一个分布式应用更加简单。开发一个利用计算机集群中资源的应用比开发一个运行在单台计算机上的应用更加困难。能写出运行在单台机器上的应用的开发者数量比能写分布式应用的开发者多好几个数量级。Hadoop 提供了一个框架，隐藏了编写分布式应用的复杂性，使得各个组织有更多可用的应用开发者。

尽管人们以一个单一产品来讨论 Hadoop，但是实际上它并不是一个单一产品。它由三个关键组件组成：集群管理器、分布式计算引擎和分布式文件系统（见图 1-1）。

2.0 版本以前，Hadoop 的架构一直是单一整体的，所有组件紧密耦合并绑定在一起。从 2.0 版本开始，Hadoop 应用了一个模块化的架构，可以混合 Hadoop 组件和非 Hadoop 技术。

图 1-1 中所示的三个概念组件具体实现为：HDFS、MapReduce 和 YARN（见图 1-2）。

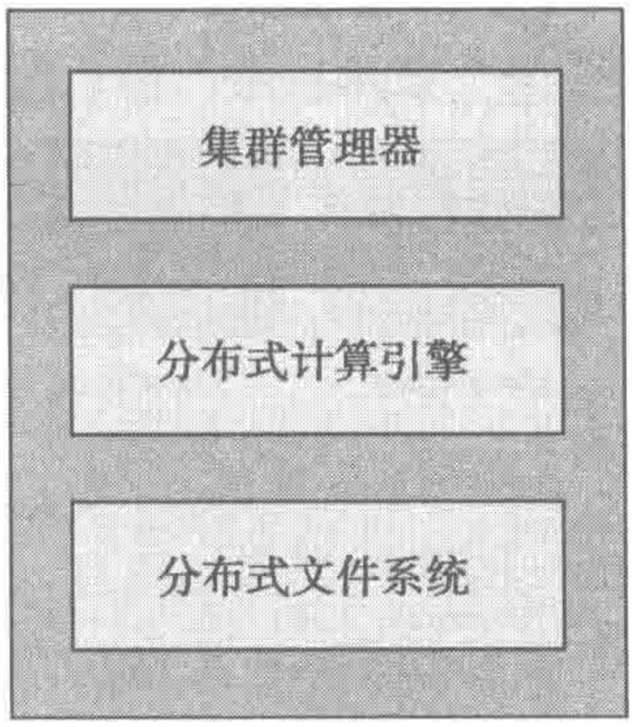


图 1-1　Hadoop 关键概念组件

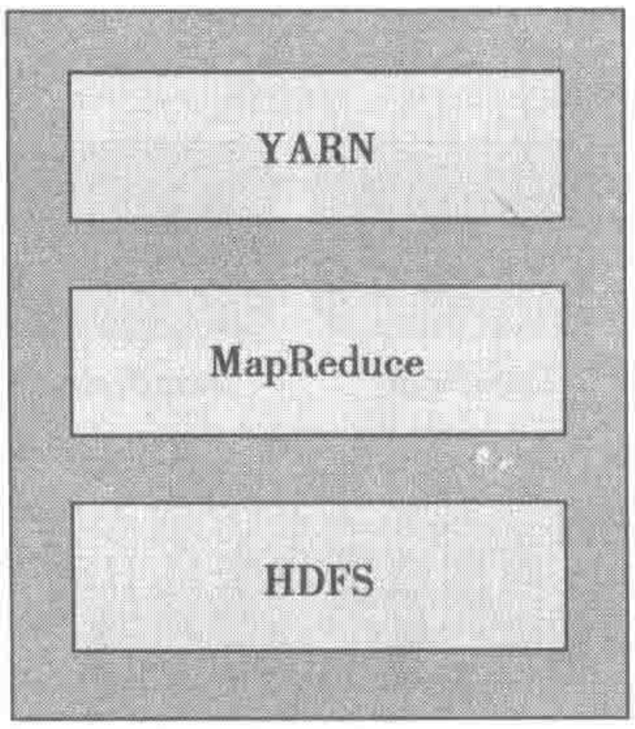


图 1-2　关键 Hadoop 组件

HDFS 和 MapReduce 在本章讨论，YARN 将在第 11 章介绍。

1.1.1　HDFS

正如其名，HDFS（Hadoop Distributed File System）是一个分布式文件系统，它在商用服务器集群中存储文件，用来存储和快速访问大文件与大数据集。这是一个可扩展、可容错的系统。

HDFS 是一个块结构的文件系统。正像 Linux 文件系统那样，HDFS 把文件分成固定大小的块，通常叫作分块或分片。默认的块大小为 128MB，但是可以配置。从这个块的大小

可清楚地看到，HDFS 不是用来存储小文件的。如果可能，HDFS 会把一个文件的各个块分布在不同机器上。因此，应用可以并行文件级别的读和写操作，使得读写跨越不同计算机、分布在大量硬盘中的大 HDFS 文件比读写存储在单一硬盘上的大文件更迅速。

把一个文件分布到多台机器上会增加集群中某台机器宕机时文件不可用的风险。HDFS 通过复制每个文件块到多台机器来降低这个风险。默认的复制因子是 3。这样一来，即使一两台机器宕机，文件也照样可读。HDFS 基于通常机器可能宕机这个假设而设计，所以可以处理集群中一台或多台机器的宕机问题。

一个 HDFS 集群包含两种类型的节点：NameNode 和 DataNode（见图 1-3）。NameNode 管理文件系统的命名空间，存储一个文件的所有元数据。比如，它追踪文件名、权限和文件块位置。为了更快地访问元数据，NameNode 把所有元数据都存储在内存中。一个 DataNode 以文件块的形式存储实际的文件内容。

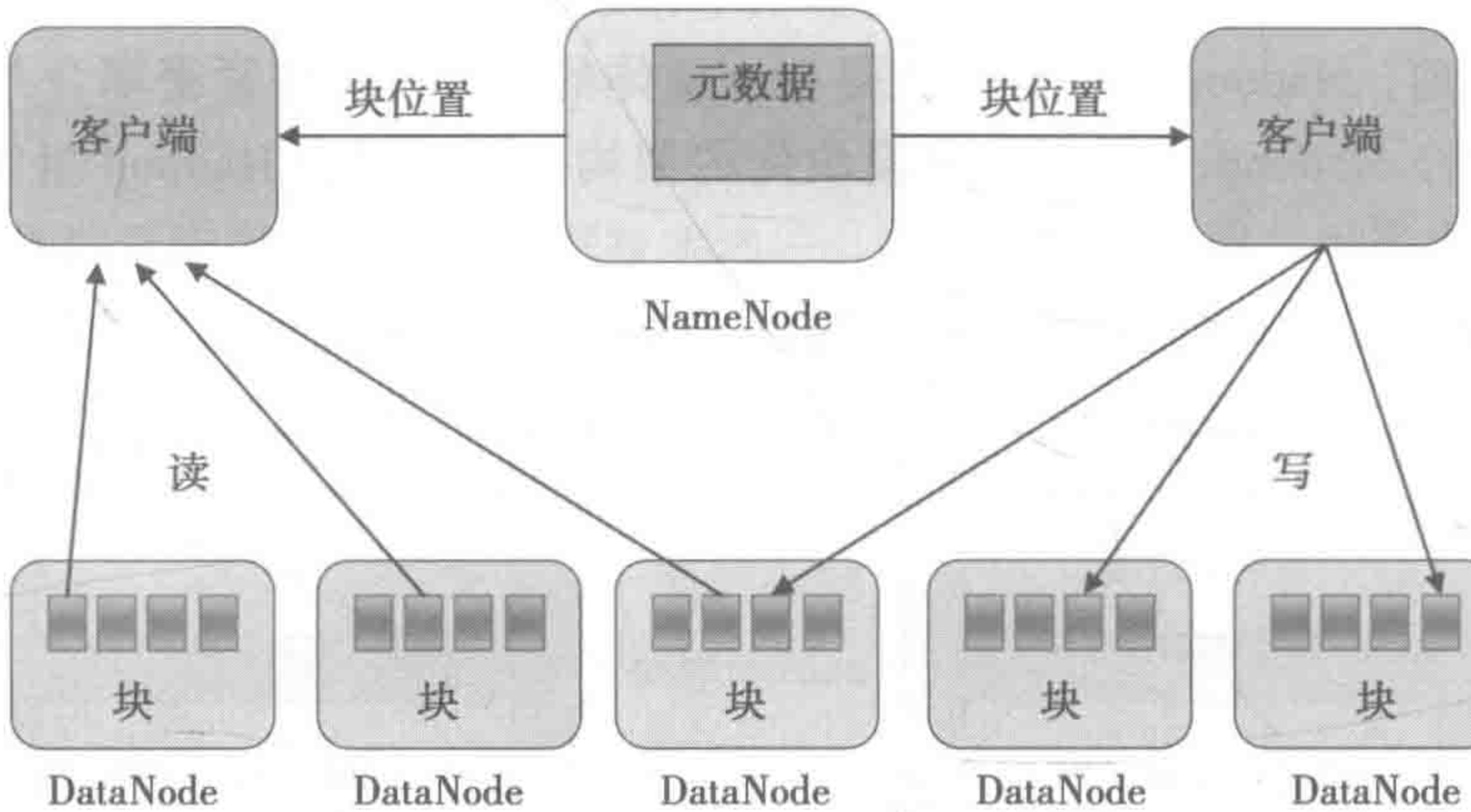


图 1-3　HDFS 架构

NameNode 周期性接收来自 HDFS 集群中 DataNode 的两种类型的消息，分别叫作心跳消息和块报告消息。DataNode 发送一个心跳消息来告知 NameNode 工作正常。块报告消息包含一个 DataNode 上所有数据块的列表。

当一个客户端应用想要读取一个文件时，它首先应该访问 NameNode。NameNode 以组成文件的所有文件块的位置来响应。块的位置标识了持有对应文件块数据的 DataNode。客户端紧接着直接向 DataNode 发送读请求，以获取每个文件块。NameNode 不参与从 DataNode 到客户端的实际数据传输过程。

同样地，当客户端应用想要写数据到 HDFS 文件时，它首先访问 NameNode 并要求它在 HDFS 命名空间中创建一个新的条目。NameNode 会检查同名文件是否已存在以及客户端是否有权限来创建新文件。接下来，客户端应用请求 NameNode 为文件的第一个块选择 DataNode。它会在所有持有块的复制节点之间创建一个管道，并把数据块发送到管道中的第一个 DataNode。第一个 DataNode 在本地存储数据块，然后把它转发给第二个 Data-