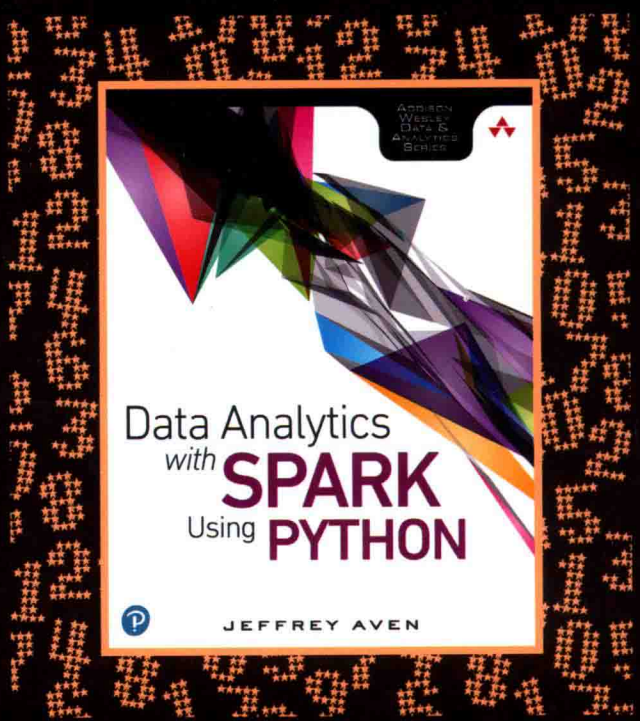
HZ BOOKS 华章IT
Pearson
数据科学与工程技术丛书

Spark数据分析

基于Python语言

[澳] 杰夫瑞·艾文（Jeffrey Aven） 著
王道远 译



DATA ANALYTICS WITH SPARK USING PYTHON

机械工业出版社
China Machine Press

数据科学与工程技术丛书

DATA ANALYTICS WITH SPARK USING PYTHON

Spark数据分析

基于Python语言

[澳] 杰夫瑞·艾文（Jeffrey Aven）著
王道远 译

机械工业出版社
China Machine Press

图书在版编目（CIP）数据

Spark 数据分析：基于 Python 语言 /（澳）杰夫瑞·艾文（Jeffrey Aven）著；王道远译．
—北京：机械工业出版社，2019.3
（数据科学与工程技术丛书）
书名原文：Data Analytics with Spark Using Python

ISBN 978-7-111-62272-7

I. S… II. ①杰… ②王… III. 数据处理软件 IV. TP274

中国版本图书馆 CIP 数据核字（2019）第 050791 号

本书版权登记号：图字 01-2018-8487

本书重点关注 Spark 项目的基本知识，从 Spark 核心开始，然后拓展到各种 Spark 扩展、Spark 相关项目、Spark 子项目，以及 Spark 所处的丰富的生态系统里各种别的开源技术，比如 Hadoop、Kafka、Cassandra 等。针对 Python 用户，介绍了如何使用 Spark 进行数据分析，涵盖了 RDD 编程、SQL 编程、流式数据处理、机器学习等内容，是一本非常好的入门指南，而作者对 Spark 的架构和大数据理解颇深，让资深用户也能梳理知识点并从中获益。

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：张梦玲　　责任校对：李秋荣
印　　刷：北京诚信伟业印刷有限公司　　版　　次：2019 年 4 月第 1 版第 1 次印刷
开　　本：185mm×260mm　1/16　　印　　张：15.5
书　　号：ISBN 978-7-111-62272-7　　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）88378991　88379833　　投稿热线：（010）88379604
购书热线：（010）68326294　　读者信箱：hzjsj@hzbook.com

译 者 序

作为一门备受欢迎的语言，Python 让编程不再是程序员的专属。Python 丰富的第三方类库和胶水语言的特性，使其能够满足各种领域的需求；同时，它简洁易读的语法，让各行各业的人都能轻松上手。上至人工智能，下至幼儿编程，Python 都是风头无两。数据分析领域中也当然少不了 Python 的身影，而随着大数据的崛起，使用 Python 处理大规模数据的需求就自然而然地出现了。尽管 Python 的性能有限，似乎与大规模数据分析绝缘，但 Spark 所提供的 Python API，借助底层充分优化，使得 Python 真正有了处理大规模数据的能力。而 Spark 的 Python 接口也扩展了其用户群体。这本书针对 Python 用户，介绍了如何使用 Spark 进行数据分析，涵盖了 RDD 编程、SQL 编程、流式数据处理、机器学习等内容，是一本非常好的入门指南，而作者对 Spark 的架构和大数据理解颇深，让资深用户也能梳理知识点并从中获益。

近几年来，大数据这个词的热度似乎已经被人工智能盖过。但是实际上，大数据的真正应用在这几年才开始走进各行各业。而 Spark 的接口非常简单易用，对于广大 Python 爱好者来说，Spark 是投身大数据革命的最佳选择。如今，Spark 的用户越来越多，其中使用 PySpark 的人也不在少数，越来越多的组织通过大数据技术创造出属于自己的价值。希望本书读者也能成为大数据的弄潮儿。

尽管译者已经结合自身多年的 Spark 开发经验对本书内容做了一定修正，但由于自身水平和所投入的时间精力有限，本书难免会有疏漏之处。欢迎读者朋友们雅正，我的电子邮箱是 me@daoyuan.wang。

感谢我的家人在翻译期间对我的支持，让我在这几个月将不多的空闲时间用于翻译，并享受着无须做家务也能过得井井有条的生活。

王道远

2018 年冬

前　言

Spark 在这场由大数据与开源软件掀起的颠覆性革命中处于核心位置。不论是尝试 Spark 的意向还是实际用例的数量都在以几何级数增长，而且毫无衰退的迹象。本书将手把手引导你在大数据分析领域中收获事业上的成功。

本书重点

本书重点关注 Spark 项目的基本知识，从 Spark 核心技术开始，然后拓展到各种 Spark 扩展技术、Spark 相关项目及子项目，以及 Spark 所处的丰富的生态系统里各种别的开源技术，比如 Hadoop、Kafka、Cassandra 等。

本书所介绍的 Spark 基本概念（包括运行环境、集群架构、应用架构等）与编程语言无关且非常基础，而大多数示例程序和练习是用 Python 实现的。Spark 的 Python API（PySpark）为数据分析师、数据工程师、数据科学家等提供了易用的编程环境，让开发者能在获得 Python 语言的灵活性和可扩展性的同时，获得 Spark 的分布式处理能力和伸缩性。

本书所涉及的范围非常广泛，涵盖了从基本的 Spark 核心编程到 Spark SQL、Spark Streaming、机器学习等方方面面的内容。本书对于每个主题都给出了良好的介绍和概览，足以让你以 Spark 项目为基础构建出针对任何特定领域或学科的平台。

目标读者

本书是为有志进入大数据领域或已经入门想要进一步巩固大数据领域知识的数据分析师和工程师而写的。当前市场非常需要具备大数据技能、懂得大数据领域优秀处理框架 Spark 的工程师。本书的目标是针对这一不断增长的市场需求培训读者，使得读者获得雇主急需的技能。

对于阅读本书来说，有 Python 使用经验是有帮助的，没有的话也没关系，毕竟 Python 对于任何有编程经验的人来说都非常直观易懂。读者最好对数据分析和数据处理有一定了解。这本书尤其适合有兴趣进入大数据领域的数据仓库技术人员阅读。

如何使用本书

本书分为两大部分共 8 章。第一部分“Spark 基础”包括 4 章，会使读者深刻理解

Spark 是什么，如何部署 Spark，如何使用 Spark 进行基本的数据处理操作。

第 1 章概要介绍大数据生态圈，包括 Spark 项目的起源和演进过程。讨论 Spark 项目的关键属性，包括 Spark 是什么，用起来如何，以及 Spark 与 Hadoop 项目之间的关系。

第 2 章展示如何部署一个 Spark 集群，包括 Spark 集群的各种部署模式，以及调用 Spark 的各种方法。

第 3 章讨论 Spark 集群和应用是如何运作的，让读者深刻理解 Spark 是如何工作的。

第 4 章介绍使用弹性分布式数据集（RDD）进行 Spark 初级编程的基础知识。

第二部分“基础拓展”包括后 4 章的内容，扩展到 Spark 的 core 模块以外，包括 SQL 和 NoSQL 系统、流处理应用、数据科学与机器学习中 Spark 的使用。

第 5 章讲解用来扩展、加速和优化常规 Spark 例程的高级元件，包括各种共享变量和 RDD 存储，以及分区的概念及其实现。

第 6 章讨论 Spark 与 SQL 的整合，还有 Spark 与非关系型数据库的整合。

第 7 章介绍 Spark 的 Streaming 子项目，以及 Streaming 中最基本的 DStream 对象。该章还涵盖 Spark 对于 Apache Kafka 这样的常用消息系统的使用。

第 8 章介绍通过 R 语言使用 Spark 建立预测模型，以及 Spark 中用来实现机器学习的子项目 MLlib。

本书代码

本书中各个练习的示例数据和源代码可以从 http://sparkusingpython.com 下载。也可以从 https://github.com/sparktraining/spark_using_python 查看或者下载。

引　言

Spark 是优秀的大数据处理平台和编程接口，与大数据技术浪潮密不可分。在本书撰写时，Spark 是 Apache 软件基金会（ASF）框架下最活跃的开源项目，也是最活跃的开源大数据项目。

数据分析、数据处理以及数据科学社区都对 Spark 有着浓厚兴趣，因而理解 Spark 是什么，Spark 能干什么，Spark 的优势在哪里，以及如何利用 Spark 进行大数据分析都是很重要的。这本书涵盖了上述内容的方方面面。

与其他介绍 Spark 的出版物不同，多数书籍主要介绍 Scala API，而本书专注于 Spark 的 Python API，也就是 PySpark。选择 Python 作为本书基础语言是因为 Python 是一门直观易懂的解释型语言，广为人知而且新手也极易上手。更何况 Python 对于数据科学家而言是一门非常受欢迎的编程语言，而数据科学家在 Spark 社区也是相当大的一个群体。

本书会从零开始讲大数据和 Spark，因此无论你是从未接触 Spark 和 Hadoop，还是已经有所接触但正在寻求全面了解 Spark 的运作方式和如何充分利用 Spark 丰富的功能，这本书都是合适的。

本书还会介绍一些相近的或者相互协作的平台、项目以及技术，比如 Hadoop、HBase、Kafka 等，并介绍它们如何与 Spark 交互与集成。

过去的几年中，我的工作都集中在这个领域，包括讲授大数据分析课程，以及为客户提供咨询服务。我经历了 Spark 和大数据乃至整个开源运动的出现和成熟，并且参与到了开源软件融入企业使用的进程中。我尽量把个人的学习历程总结到了本书中。

希望这本书能助你成为大数据和 Spark 从业人员。

目　录

第一部分

Spark 基础

第 1 章

大数据、Hadoop、Spark 介绍

在古代，人们使用牛来拉重物，而当一头牛拉不动的时候，人们并不会尝试把牛养得更壮。我们也不应该尝试使用更强大的计算机，而应该尝试使用更多的计算机。

——美国计算机科学家，海军准将格蕾丝·穆雷·赫柏[⊖]

本章提要

- 大数据与 Apache Hadoop 项目简介
- Hadoop 核心组件（HDFS 和 YARN）概览
- Apache Spark 简介
- PySpark 编程所需的 Python 基础，包括函数式编程基础知识

Hadoop 和 Spark 项目都和大数据运动密不可分。从项目早期主要用于搜索引擎厂商和学术界，到现在用于从数据仓库到复杂事件处理（Complex Event Processing，CEP）再到机器学习的各种各样的应用中，Hadoop 和 Spark 已经在数据格局中做出了不可磨灭的贡献。

本章会介绍一些基本的分布式计算概念、Hadoop 项目和 Spark 项目、Python 函数式编程，为你后续的学习打下坚实的基础。

1.1 大数据、分布式计算、Hadoop 简介

在讨论 Spark 之前，有必要回顾并理解所谓“大数据”的历史。要想成为精通 Spark 的专家，你不仅需要理解 Hadoop 以及 Spark 对它的用法，还要理解 Hadoop 项目的一些核心概念，比如数据本地化、无共享和映射 – 归约（MapReduce），因为它们对于 Spark 而言都适用且不可或缺。

1.1.1 大数据与 Hadoop 简史

我们常说的“大数据”是一套数据存储和处理的方法论，它最早在本世纪初出现于搜索

⊖ 格蕾丝·穆雷·赫柏（Grace Murray Hopper），COBOL 语言主要设计者，本书原文为海军少将（Rear Admiral），美国 1983 年将原海军准将（Commodore）军衔改称为一星少将（Rear Admiral lower half，RDML），自此美国一星准将和二星少将（Rear Admiral，RADM）都被称为少将（Rear Admiral），而其他使用类似军衔体系的国家中少将仍仅指二星少将，格蕾丝属于一星少将，因此此处译文中仍使用“准将”一词。——译者注

引擎厂商，主要是谷歌和雅虎。搜索引擎厂商是第一批遇到互联网规模（Internet-scale）问题的，主要问题是如何处理与存储互联网世界里所有文件的索引。尽管现在互联网的体量比起当初早已翻了数倍，但上述问题在当时依然是一个巨大的挑战。

雅虎和谷歌分别独立入手开发解决这一挑战的方法。在 2003 年，谷歌发表了一篇题为《The Google File System》的白皮书。紧接着，在 2004 年，谷歌又发布了另一篇题为《MapReduce: Simplified Data Processing on Large Clusters》的白皮书。差不多在同一时间，Doug Cutting（公认的 Hadoop 项目创始人）和 Mike Cafarella 正在忙于一个名为 Nutch 的网络爬虫项目，该项目基于 Cutting 的开源项目 Lucene（现在是 Apache Lucene）。谷歌发布的白皮书启发了 Cutting，他将 Nutch 项目中做的一些工作与这些白皮书中列出的存储和处理原理进行了整合。整合的成果就是如今的 Hadoop。后来在 2006 年，雅虎决定接受 Hadoop，并雇佣 Doug Cutting 让他全职开展该项目的相关工作。Hadoop 在 2006 年成为 Apache 软件基金会的一员。

Apache 软件基金会

Apache 软件基金会（ASF）是 1999 年成立的非营利性组织，为开发者提供向开源项目做贡献的开源软件结构与框架。ASF 鼓励合作与社区参与，保护志愿人员免于相关诉讼。ASF 以精英管理的概念为前提，意味着项目受到绩效的支配。

贡献者（contributor）是对项目贡献了代码或者文档的开发人员。他们通常活跃于邮件组和答疑论坛，对项目缺点提出意见和建议，或是提出解决问题的代码补丁。

提交者（committer）是因专业绩效突出而获得一个项目主代码仓库的代码提交权限的开发人员。提交者需要签署贡献者许可协议（Contributor License Agreement，CLA），会拥有一个 apache.org 后缀的电子邮箱地址。提交者形成一个委员会来做项目相关的一些决策。

访问 http://apache.org/ 可以获取更多关于 Apache 软件基金会的信息。

差不多在 Hadoop 项目诞生的同时，还有一些其他的技术革新也在进行中，包括如下几项：

- 电子商务的疾速扩张
- 移动互联网的诞生与迅速成长
- 博客与用户驱动的网络内容
- 社交媒体

这些革新累积导致所生成的数据量指数级增长。数据的洪流加速了大数据运动的扩张，进而导致了其他相关项目的出现（如 Spark），还有开源消息系统（如 Kafka），NoSQL 平台（如 HBase 和 Cassandra），这些都会在本书后续内容中进行讨论。

但这一切都从 Hadoop 开始。

1.1.2　Hadoop 简介

Hadoop 是一个数据存储与数据处理平台，项目起源于数据本地化的核心概念。**数据本**

地化（data locality）指在数据存储的地方处理数据，让计算靠近数据，而不是像数据库管理系统那样向数据存储请求数据，发到远端数据处理系统或者主机进行计算。

由于网络应用快速发展导致大数据的出现，在计算时通过网络传输大量数据的传统方式不再高效、实用，有时甚至无法实现。

Hadoop 让大型数据集可以在各节点上通过**无共享**（shared nothing）的方式在本地处理，每个节点可以独立处理比整个数据集小得多的一个子集，而无须与其他节点通信。这种特性是通过分布式文件系统实现的。

Hadoop 在执行写操作时是没有结构信息的。这就是所谓的**读时模式**（schema-on-read）系统。这意味着 Hadoop 可以存储和处理各种各样的数据，无论是无结构的文本文档，还是半结构化的 JSON（JavaScript Object Notation）文档或 XML 文档，又或是从关系型数据库中提取的完全结构化的数据。

读时模式系统和我们所熟知的关系型数据库有着本质区别。关系型数据库通常被认为属于**写时模式**（schema-on-read），数据一般具有强结构性，表结构是预定义的，并且在 INSERT、UPDATE 和 UPSERT 等操作时要求结构完全匹配。

类似 HBase 或者 Cassandra 这样的 NoSQL 平台也属于读时模式系统。你会在第 6 章了解到更多 NoSQL 平台相关的内容。

由于在 Hadoop 的写操作时并没有明确的结构信息，所以写出的文件并没有索引、统计信息，或是数据库系统常用的其他一些用于优化查询操作、筛选或减少返回到客户端的数据量的数据结构。这进一步凸显了数据本地化的必要性。

Hadoop 的设计思路是通过对大型问题分而治之的方法，运用数据本地化与无共享的概念，将大型问题切分成一系列小规模的问题，实现“大海里捞针”。Spark 也使用了非常相似的概念。

1. Hadoop 核心组件

Hadoop 包含两个核心组件：HDFS（Hadoop 分布式文件系统）和 YARN（Yet Another Resource Negotiator，另一个资源协调器）。HDFS 是 Hadoop 的存储系统，而 YARN 可以当作 Hadoop 处理或者资源调度的子系统（如图 1.1 所示）。

这两个组件相互独立，可以各自运行在自己的集群上。当 HDFS 集群和 YARN 集群部署在一起时，我们把这两个系统的组合称为一个 Hadoop 集群。Hadoop 的这两个核心组件都可以被 Spark 利用起来，本章的后续部分会进一步讨论。

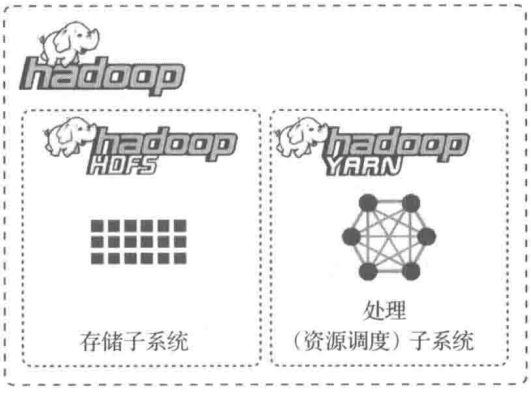


图 1.1　Hadoop 核心组件

集群术语

集群（cluster）指一组协同工作执行诸如计算或处理功能的系统。集群中的单个服务

器称为节点（node）。

集群可以有多种拓扑和通信模型。其中一种模型为主 - 从模型。主 - 从模型是由一个进程控制其他至少一个进程的通信方式。在某些系统中，直到运行时或处理任务时，主节点才从一组可用的进程中选出来的。但在其他情况下，比如 HDFS 集群或者 YARN 集群里，主进程和从进程的角色都是预先分配好的，在集群整个生命周期里不会发生变化。

任何以某种方式与 Hadoop 交互或者整合的项目都称为 Hadoop“生态圈”项目，比如 Flume、Sqoop 等数据接入项目，或者 Pig、Hive 等数据分析工具。Spark 可以当作一个 Hadoop 生态圈项目，不过这有一些争议，因为 Spark 无须 Hadoop 也能运行。

2. HDFS：文件、数据块、元数据

HDFS 是一种虚拟文件系统，其中的文件由分布在集群中至少一个节点上的**数据块**（block）组成。在把文件上传到文件系统的时候，文件会按照配置好的数据块大小进行分割，这样的过程称为数据接入（ingestion）。分割后得到的数据块分布在整个集群的节点上，每个数据块会重复出现在几个节点上，以此实现容错，并提高本地处理数据的概率（设计目的是“让计算靠近数据”）。HDFS 数据块由 HDFS 集群从节点的 DataNode 进程存储和管理。

DataNode 进程是 HDFS 从节点守护进程，运行在 HDFS 集群中至少一个节点上。DataNode 负责管理数据块存储和数据读写访问，还有数据块复制，这也是数据接入过程的一部分，如图 1.2 所示。

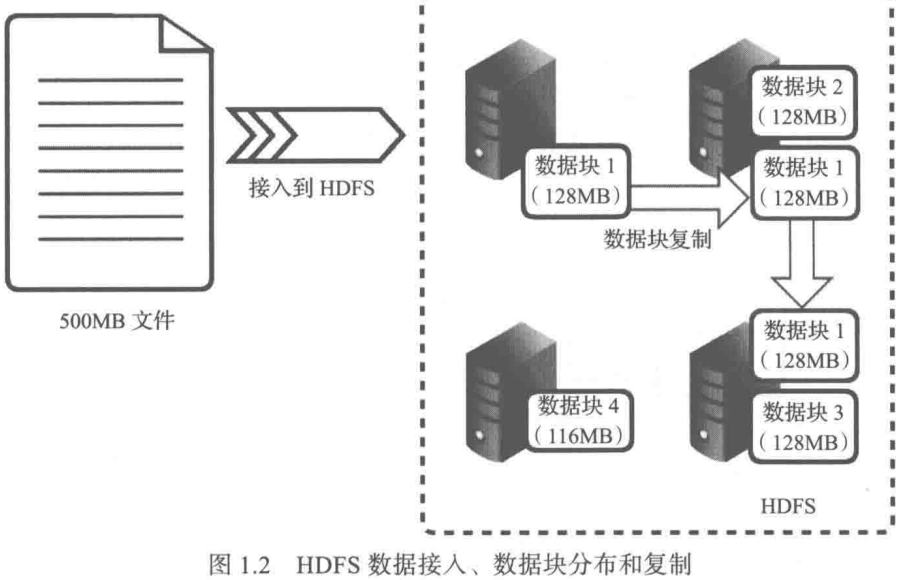


图 1.2　HDFS 数据接入、数据块分布和复制

在一个完整的分布式 Hadoop 集群中，通常有许多主机运行着 DataNode 进程。后面你会看到 DataNode 以数据分区的形式为部署在 Hadoop 集群上的 Spark 应用的分布式 Spark 工作者进程提供输入数据。

文件系统的元数据中存储着文件系统的信息，还有其中的目录、文件信息，以及组成文件的物理数据块信息。HDFS 的主节点进程称为 NameNode，文件系统元数据就存储在

NameNode 进程的常驻内存里。HDFS 集群的 NameNode 通过类似于关系型数据库事务日志的日志功能为元数据提供持久性。NameNode 负责为 HDFS 客户端提供读写数据块的具体位置，这样客户端就可以直接和 DataNode 通信并进行数据操作。图 1.3 呈现了 HDFS 读操作的示意图，而图 1.4 解析了 HDFS 写操作的过程。

3. 用 YARN 进行应用调度

YARN 管理并协调着 Hadoop 里的数据处理。在这种场景下，数据一般都以 HDFS 作为输入输出源。YARN 集群架构使用的是与 HDFS 类似的主从集群框架，主节点守护进程称为 ResouceManager，而从节点守护进程称为 NodeManager，至少有一个，运行在集群的从节点上。

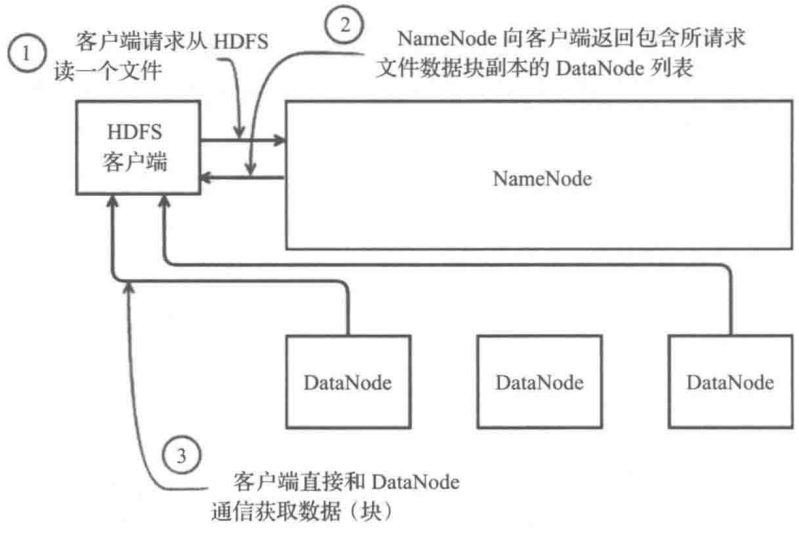


图 1.3　HDFS 读操作解析

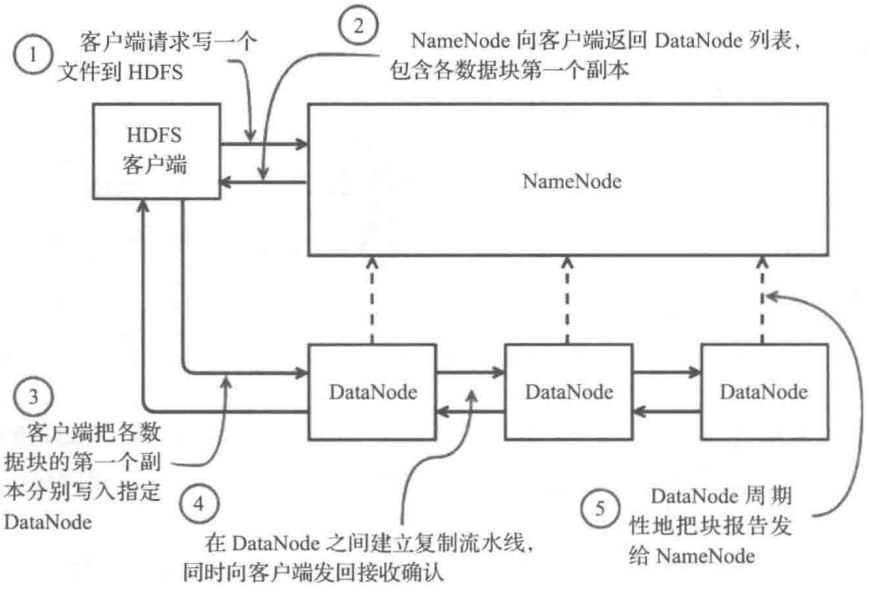


图 1.4　HDFS 写操作解析

ResourceManager 负责为集群上运行的应用分配集群计算资源。资源以容器作为单位分配，容器有预定义好的 CPU 核心数和内存限制。容器分配的最大最小阈值等都可以在集群中进行配置。使用容器可以保障进程间的资源隔离。

ResourceManager 也会在随应用退出并释放所占资源时维护集群剩余可用的资源量，同时还跟踪集群上当前运行的应用的状态。ResourceManager 默认会在所运行主机的 8088 端口上提供内嵌的网页版用户交互界面，这对于查看应用状态很有用，无论应用正在运行、运行完成或是运行失败，如图 1.5 所示。这个用户界面在管理 YARN 集群上运行的 Spark 应用状态时需要经常用到。

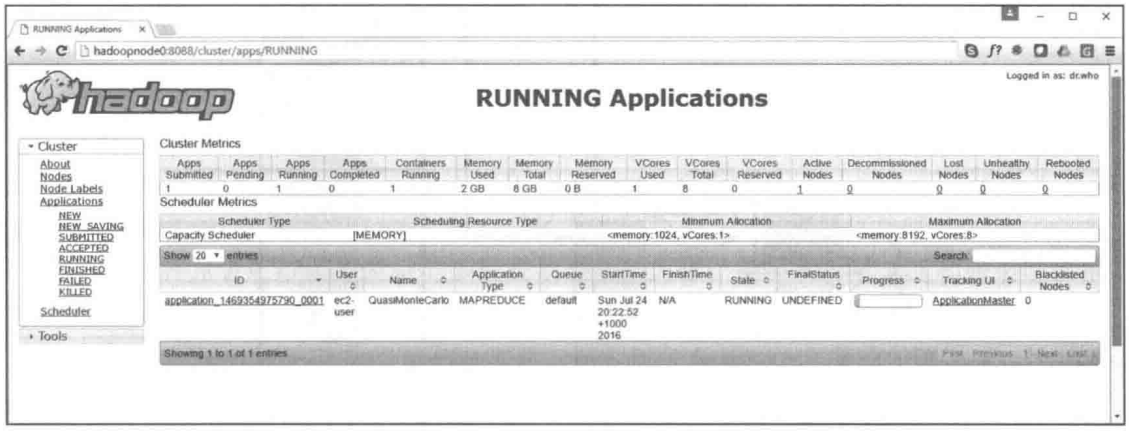


图 1.5　YARN 资源管理器 ResourceManager 用户界面

客户端把应用（比如一个 Spark 应用程序）提交到 ResourceManager，然后 ResourceManager 首先从集群中一个可用的 NodeManager 里分配出应用的第一个容器，作为应用的委托进程，这个进程就是 ApplicationMaster。然后 ApplicationMaster 继续为应用申请运行任务所需的其余容器。

NodeManager 是 YARN 从节点守护进程，管理从节点主机上运行的容器。容器用于执行应用里的任务。回想一下无共享的概念，Hadoop 解决大规模问题的思路是“分而治之”，大规模问题被分解为一堆小规模任务，很多任务可以并发执行。这些任务都运行在容器里，该容器由运行着 NodeManager 进程的主机分配。

大多数容器只是运行任务。不过，ApplicationMaster 会额外负责管理整个应用。前面介绍过，ApplicationMaster 是由 ResourceManager 从 NodeManager 上分配的第一个容器。它的任务是规划整个应用，包括决定需要什么资源（通常基于要处理多少数据）以及为应用的各阶段（稍后会介绍）安排资源。ApplicationMaster 代表应用向 ResourceManager 申请这些资源。ResourceManager 从 NodeManager 上（可以是同一个 NodeManager，也可以是其他的 NodeManager）给 ApplicationMaster 分配资源以供该应用使用，直到该应用退出。后面会详细介绍，对于 Spark 而言，ApplicationMaster 会监控任务、阶段（一组可以并发执行的 Spark 任务）还有依赖的进度。综述信息会传给 ResouceManager，展示在前面介绍过的用户界面中。图 1.6 展示了 YARN 应用提交、调度和执行的过程的概况。

图 1.6 描述的过程如下所述：