Spark Streaming

实时流处理入门与精通

[印度] Sumit Gupta 著

[中国] 韩燕波　刘晨　苏申 等 译

用 Spark Streaming 轻松实现高可扩展、高容错的流数据应用

[PACKT] PUBLISHING

工信出版集团　電子工業出版社 PUBLISHING HOUSE OF ELECTRONICS INDUSTRY http://www.phei.com.cn

Spark Streaming
实时流处理入门与精通

[印度] Sumit Gupta 著

[中国] 韩燕波 刘晨 苏申 等 译

電子工業出版社

Publishing House of Electronics Industry

北京·BEIJING

内 容 简 介

本书主要对Spark和Spark的安装、配置、主要架构和组件进行介绍，并介绍如何利用Spark Streaming进行实时数据的处理，讨论利用Spark Streaming的多种API和操作进行近实时的分布式日志流的处理。本书要求读者对Scala有很好的认识和理解，以便能够利用核心组件和应用进行高效编程。

版权贸易合同登记号 图字：01-2016-4602

图书在版编目（CIP）数据

Spark Streaming：实时流处理入门与精通 /（印）苏密特·古普塔（Sumit Gupta）著；韩燕波等译. 北京：电子工业出版社，2017.4
书名原文：Learning Real-Time Processing with Spark Streaming

ISBN 978-7-121-31049-2

Ⅰ. ①S… Ⅱ. ①苏… ②韩… Ⅲ. ①数据处理软件 Ⅳ. ①TP274

中国版本图书馆CIP数据核字（2017）第044547号

策划编辑：董亚峰
责任编辑：董亚峰
特约编辑：彭　瑛　赵海军等
印　　刷：北京嘉恒彩色印刷有限责任公司
装　　订：北京嘉恒彩色印刷有限责任公司
出版发行：电子工业出版社
　　北京市海淀区万寿路173信箱　　邮编：100036
开　　本：787×980　1/16　　印张：11.5　　字数：240千字
版　　次：2017年4月第1版
印　　次：2017年4月第1次印刷
定　　价：39.00元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888，88258888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

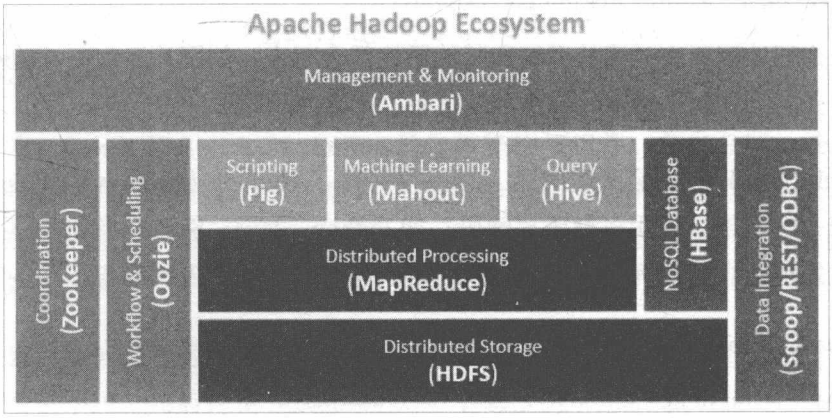
本书咨询联系方式：QQ3502629。

序　言

处理大规模数据并产生具有商业前瞻性的信息是当下的流行趋势，它的目标是在历史数据上的商务智能（Business Intelligence，BI）挖掘。很多企业致力于开发数据仓库（https://en.wikipedia.org/wiki/Data_warehouse），希望尽可能地获取多方面的数据，并选择合适的 BI 工具以分析数据仓库中的数据。然而，开发数据仓库是复杂、耗时且昂贵的过程，可能会花费几个月甚至几年的时间。

毫无疑问，Hadoop 及其生态系统提供了解决大数据问题的新框架。这种解决方案廉价且可扩展，能够在几小时内处理 TB 级别的数据，而这在过去要花上几天时间。

下图为典型的 Apache Hadoop 生态系统及其用于解决大数据问题的多种组件。



图片来源：http://www.dezyre.com/article/big-data-and-hadoop-training-hadoop-components-and-architecture/114

如果说设计 Hadoop 只是为进行批处理，就有点片面了。因为在一些商务案例中，实时的或者近实时的商业数据分析需求量也很大，这被称作实时商务智能（Real-

Time Business Intelligence，RTBI）或者近实时商务智能（Near Real-Time Business Intelligence，NRTBI）（https://en.wikipedia.org/wiki/Real-time_business_intelligence），也被称作“快数据”（Fast Data），它代表近实时的决策能力和缩短商务决策时间数量级的能力。

目前已经出现了多款平台来满足这些企业的实时数据分析需求，这些平台功能强大、易用而且开源。其中最引人注目的两个平台就是 Apache Storm 和 Apache Spark，它们给潜在用户提供了更宽泛的实时数据处理能力。这两个项目同属 Apache 软件基金会，彼此的处理能力既有重叠的部分，又各具特色，因此在项目开发中扮演着不同的角色。

对于分布式流数据的可靠处理来讲，Apache Storm 是一个出色的框架，或者可以说它适用于复杂事件处理（Complex Event Processing，CEP）风格的数据处理（https://en.wikipedia.org/wiki/Complex_event_processing）。它能解决大多数近 / 实时数据处理案例，但是它解决不了下面这些问题：

- 如果同样的数据既需要批处理，也需要近实时的处理怎么办？难道要部署两个不同的框架吗？
- 怎么融合两种不同来源的数据流？
- 除了 Java，我还能用别的编程语言吗？
- 怎么把处理近实时流的系统和其他系统整合起来，如图片、SQL、Hive 等？
- 近实时的推荐、聚类、分类怎么处理？

Apache Spark 解决了上述问题，它不仅保留了 Hadoop 和 Storm 的优点，还可以在一个统一的框架下使用多种语言编码，如 Python、Java、Scala。你甚至还可以在数据流与批处理的应用案例中使用同一段代码。同时它也支持多种库与扩展，例如：

- Spark GraphX：用于开发图形编程。
- Spark DataFrames and SQL：执行 SQL 语句。
- Spark MLlib：执行用于推荐、聚类和分类的机器学习算法。
- Spark Streaming：用于实时处理流数据。

Apache Spark 的一个值得注意的特点是这些库和扩展的互用。例如，近实时数据

流可以被转换成图形或者用 SQL 语句进行分析，或者可以在流数据上执行机器学习算法来提供推荐、聚类或者分类。

很有趣，不是吗？

Apache Spark 最开始是加利福尼亚大学伯克利分校的 AMP 实验室开发的一个项目，后来加入 Apache 育成计划并于 2014 年 2 月成为顶级项目。Spark 不仅是一个通用的分布式计算平台，它同时支持批处理和近实时数据的处理。

接下来我们将讨论用 Spark Streaming 进行实时处理的实质问题。

在本书中，我们将介绍 Spark Streaming 的多方面内容，包括它的安装和配置、体系结构、操作、与其他 Spark 库及 NoSQL 数据库的融合，以及产品环境中的部署问题。

本书主要内容

第 1 章，Spark 和 Spark Streaming 的安装与配置。首先详细介绍 Spark 和 Spark Streaming 的安装和配置，包括运行 Spark 作业的软硬件配置，Spark 封装的一些工具和实用程序的功能与用法。然后介绍开发者如何用 Java 和 Scala 编写他们的第一个 Spark 作业，以及如何在集群上执行这个作业。最后讨论 Spark 和 Spark Streaming 常见错误的定位技巧。

第 2 章，Spark 和 Spark Streaming 的体系结构与组件。首先介绍批处理和实时数据处理的总体模型和复杂性。然后详细描述 Spark 体系结构，以及 Spark Streaming 在体系结构中的位置。最后介绍开发者如何编写他们的第一段 Spark Streaming 代码并在 Spark 集群上执行。

第 3 章，实时处理分布式日志文件。首先讨论 Spark 和 Spark Streaming 的包结构和多个重要的 API。然后讨论 Spark 和 Spark Streaming 的两个核心组件——弹性分布式数据集和离散流。最后介绍一个分布式日志处理案例，这个案例利用 Apache Flume 从分布式数据源加载数据并交由 Spark Streaming 分发和执行作业。

第 4 章，在流数据中应用 Transformation。首先讨论 Spark Streaming 提供的多种

转换操作，包括功能、转换和窗口操作，然后在流数据上应用这些转换操作来改进我们的分布式日志处理案例。同时讨论了改善 Spark Streaming 作业性能的多种因素和一些考虑。

第 5 章，日志分析数据的持久化。介绍 Spark Streaming 的输出操作，并用 Apache Cassandra 来持久化 Spark Streaming 接收和处理的分布式日志数据。

第 6 章，与 Spark 高级库集成。改善分布式日志文件处理案例并讨论了 Spark Streaming 与 Spark 高级库（比如 GraphX 和 Spark SQL）的集成。

第 7 章，产品部署。讨论了在产品中部署 Spark Streaming 时需要考虑的多方面的问题，包括高可用性、延迟容忍、监控等。同时讨论了在其他集群计算框架（如 YARN 和 Apache Mesos）中部署 Spark Streaming 作业的过程。

看这本书需要哪些准备

你应该已经有一些 Scala 的编程经验，并对某个分布式计算平台（如 Apache Hadoop）有一些基本的知识和了解。

本书的目标读者

本书需要读者对 Scala 有很好的认识和理解，以便能够利用核心组件和应用进行高效编程。

如果你正在阅读本书，那么你可能已经对 Scala 有了丰富的了解。这本书讲的是利用 Spark Streaming 进行实时数据的处理，同时讨论了利用 Spark Streaming 的多种 API 和操作进行近实时的分布式日志流的处理。

惯例

在本书中，你会看到一些文本的格式和其他信息的格式不同，这里先举几个例子并解释它们的含义。

文本、数据库表名、文件夹名、文件名、文件扩展名、路径、伪 URL、用户输

入和 Twitter 标签的字体像下面这样：“在 Linux 操作系统上执行 `sudo ulimit -n 20000` 可以增加 `ulimit`。”

语句块都写成下面的形式：

```
public class JavaFirstSparkExample {
  public static void main(String args[]){
   //Java Main Method
  }
}
```

命令行的输入和输出都写成下面的形式：

```
export FLUME_HOME=<path of extracted Flume binaries>
```

新定义和重要的名词都会用粗体标明。在屏幕上看到的词，比如在菜单或者对话框中看到的词，会以如下文本的形式显示：“单击‘下一步’按钮可以移动到下一屏。”

警告或者重要提示会出现在这样的方框里。

提示和技巧会这么写。

读者反馈

欢迎我们的读者反馈意见。你喜欢或者不喜欢本书的哪一点，请告诉我们。读者的反馈对我们来说非常重要，这能帮助我们做出读者最能获益的内容来。

请将普通反馈通过 E-mail 发送到 `feedback@packtpub.com`，并附上书名和你的反馈信息。

如果你希望在你精通的或者感兴趣的话题上写本书或者贡献内容，那么请参看 `www.packtpub.com/authors` 的作者指南。

用户支持

你现在已经是 Packt 书籍的荣誉用户了，我们有一系列方法来帮助你从你购买的书中获取最大的价值。

下载样例代码

你可以从 http://www.packtpub.com 的账户里下载所有你购买过的 Packt 书籍的样例代码。如果你是在其他地方买到的这本书，则可以访问 http://www.packtpub.com/support 并注册，我们会把样例代码通过电子邮件发送给你。

勘误

虽然我们已经十分小心地确保本书内容的准确性，但犯错是难免的。如果你在这本书里发现了错误（可能是文本错误，也可能是代码错误），那么十分感谢你能反馈给我们。这能避免其他读者因同样的错误而懊恼，也能帮助我们在本书后续的版本中纠正这个错误。如果你发现了任何错误，那么请通过 http://www.packtpub.com/submit-errata 反馈给我们，你只需选择书名，单击“勘误提交表”链接，并输入你的勘误具体信息。一旦你的勘误信息被核实，你提交的信息就会被接受，并上传到我们的网站上，或者加入该书的已有勘误信息列表中。

你可以在 https://www.packtpub.com/books/content/support 上看到以前提交的勘误内容，在查询区域里输入本书的名字进行搜索就可以了。你需要的信息会在勘误区中出现。

盗版行为

将受版权保护的内容放在 Internet 上传播对任何一种媒体形式都是一个持续存在的问题。Packt 对版权和授权的保护非常重视。如果你在 Internet 上以任何形式看到我们的工作的非法传播版本，那么请尽快把地址或者网站名字发给我们，以便我们进行维权。

请通过 copyright@packtpub.com 联系我们，并附上疑似盗版内容的链接。

十分感激你帮助我们保护读者，让我们能够带给你有价值的内容。

其他问题

如果你对本书还有什么问题，则可以通过电子邮件 question@packtpub.com 联系我们，我们会尽力解决你的问题。

目 录

第 1 章

Spark 和 Spark Streaming 的安装与配置

Apache Spark（http://spark.apache.org/）是一个通用的、开源的集群计算框架，由加利福尼亚大学伯克利分校的 AMP 实验室于 2009 年开发。

Spark 不但给不同商业场景带来新的数据处理方式，同时也给通用框架下的实时批处理操作提供了一个整合的平台。根据用户企业的商业需求，数据的处理频率可以高至每秒一次（甚至更快），也可以低至每天一次。

作为一个通用的分布式处理框架，Spark 可以进行快速应用开发（**Rapid Application Development，RAD**），同时也允许将代码在不同的批和流应用之间复用。Spark 的一个最具诱惑的特征就是：你在自己的台式机或者笔记本电脑上写的代码可能被 Apache Mesos（http://mesos.apache.org/）或者 Apache Hadoop YARN（https://hadoop.apache.org/）上的其他集群管理者一行不改地部署。

在接下来的几章里，我们将陆续介绍 Spark 的特征，但首先我们需要准备 Apache Spark 的开发环境（安装和配置）。

本章将帮助你理解 Apache Spark 和 Spark Streaming 的范例、适用性和特征。同时也将详细介绍 Spark 的安装流程，并让你在 Spark 框架下运行你的第一个程序。看

完这一章，你应该已经充分地配置了你的 Spark 工作环境，并且可以进行进一步的应用开发。本章内容包括：

- 安装 Spark。
- 配置和运行一个 Spark 集群。
- 你的第一个 Spark 程序。
- 管理员工具。
- 疑难解答。

安装 Spark

本节讨论 Spark 的不同安装问题和支撑组件。

Spark 支持多种软硬件平台。它既可以安装在商品硬件上，也可以安装在高端服务器上。Spark 集群既可以由云提供，也可以由自己配置的机器提供。虽然 Spark 没有配置方面的标准，但是我们依然可以定义“必要配置”和“优选配置”，其他配置视用例需求而定。

在第 7 章中，我们将更详细地讨论部署方面的问题，现在我们先跳过这些讨论，说说 Spark 应用开发的软硬件需求。

硬件需求

本节讨论 Spark 的批和实时应用的开发硬件需求。

中央处理器

Spark 提供两种数据处理方法：批处理和实时处理。这两种方法的负载都是 CPU 密集的。在大规模部署下，计算资源的使用必须精雕细琢，而 Spark 解决这个问题的方式是减少线程间的资源共享或者上下文切换。这么做是为了给每个线程提供足够的计算资源，以便每个线程都能独立且实时地计算结果。以下是 Spark 集群的每个节点的 CPU 的推荐配置。

- 必要配置：双核。
- 优选配置：16 核。

内存

实时数据处理和低延迟任务都是从内存中读 / 写的，从硬盘中读 / 写会影响性能。

Spark 通过在内存中缓存数据集来优化内存密集任务，这样就可以直接在内存中读取和处理数据，而基本不需要从硬盘中读取数据。

内存配置的基本规则是“越大越好”，但这要基于你的实际应用需求。Spark 是在 Scala 中实现的（http://www.scala-lang.org/），Scala 需要一个 Java 虚拟机运行环境。这一点和其他基于 Java 的应用一样。为了优化配置，需要优化内存配置。一般来讲，我们只能将不多于 75% 的可用内存分配给 Spark 应用，剩下的留给操作系统和其他系统进程。考虑到 Java 垃圾收集器的各方面限制，以下为内存配置需求。

- 必要配置：8GB。
- 优选配置：24GB。

硬盘

有些数据在内存里放不下，那么最终就需要持续存储介质来存放这些数据。Spark 会自动将内存中溢出的数据存放在硬盘里，或者在必要的时候重算一遍。和内存的配置需求一样，硬盘的配置需求视应用而定，但我们推荐如下配置。

- 必要配置：每分钟 15000 转的 SATA 硬盘，每块容量至少为 1 ~ 2TB。
- 优选配置：非磁盘阵列（RAID）体系结构——部署简单的磁盘捆绑（**Just Bunch of Disks，JBOD**）而不做数据冗余。如果是固态硬盘则更好，因为固态硬盘吞吐率高、响应快。