本书全面、系统地介绍Spark源码
为读者提供了一系列分析源码的实用技巧
始终抓住资源分配、消息传递、容错处理等基本问题
一步步寻找答案，所有问题迎刃而解

Broadview
www.broadview.com.cn

Apache Spark 源码剖析

许鹏 | 著

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

Apache Spark 源码剖析

许鹏 | 著

電子工業出版社
Publishing House of Electronics Industry
北京•BEIJING

内容简介

本书以Spark 1.02版本源码为切入点，着力于探寻Spark所要解决的主要问题及其解决办法，通过一系列精心设计的小实验来分析每一步背后的处理逻辑。

本书第3～5章详细介绍了Spark Core中作业的提交与执行，对容错处理也进行了详细分析，有助读者深刻把握Spark实现机理。第6～9章对Spark Lib库进行了初步的探索。在对源码有了一定的分析之后，读者可尽快掌握Spark技术。

本书对于Spark应用开发人员及Spark集群管理人员都有极好的学习价值；对于那些想从源码学习而又不知如何入手的读者，也不失为一种借鉴。

图书在版编目（CIP）数据

Apache Spark源码剖析 / 许鹏著. — 北京：电子工业出版社，2015.3
ISBN 978-7-121-25420-8
I. ① A…II. ① 许…III. ① 互联网络－网络服务器 ② 数据处理软件 IV. ① TP368.5 ② TP274

中国版本图书馆CIP数据核字（2015）第010897号

策划编辑：付　睿
责任编辑：李云静
印　　刷：北京天来印务有限公司
装　　订：北京天来印务有限公司
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：18.5　字数：432千字
版　　次：2015年3月第1版
印　　次：2015年3月第1次印刷
定　　价：68.00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。

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。

服务热线：（010）88258888。

前言

笔者接触Spark时间不算很长，而本书之所以能够出版，凭借的是浓厚的兴趣和执着之心。

这一切还要从Storm说起。笔者一直在做互联网相关工作，但接触大数据的时间并不长，当时Hadoop和Storm等非常红火，引起了笔者的“窥视”之心。从2013年开始，笔者打算看看Hadoop的源码实现，观察其代码规模，发觉所花时间可能会很长。恰好其时Storm风头正劲，于是转向Storm源码，0.8版的Storm代码规模不过20 000行左右，感觉还是比较好入手的。

Storm源码分析期间，笔者还学习了Clojure、ZeroMQ、Thrift、ZooKeeper、LMAX Disruptor等新技术，对于实时流数据处理算是有了一个大概的了解。由于听说在实时流数据处理领域Spark技术也很强悍，而且在容错性方面具有天生的优势，更引发了笔者的兴趣，为了弄清楚究竟，于是开始了Spark的源码走读过程。

笔者是以读Spark论文开始的，说老实话觉得晦涩难懂，因为无法将其映射到内存使用、进程启动、线程运行、消息传递等基本问题上。或许换个方法会更好，故笔者选择直接从源码入手，如此一来事情反而变简单了。在源码分析的过程中，笔者始终抓住资源分配、消息传递、容错处理等基本问题设问，然后一步步努力寻找答案，所有的问题渐渐迎刃而解。

笔者关于源码分析有一个心得，就是要紧紧把握住计算的基本模型，然后结合新分析问题的业务领域，将业务上的新问题转换到计算处理的老套路上来，然后就可以以不变应万变，而不被一些新技术名词晃花了眼。这里所说的老套路是指从操作系统的角度来看，如果能事先深度了解操作系统，将对分析一些新应用程序大有裨益。

Spark源码采用Scala语言编写，那么阅读Spark源码之前，是否一定要先学Scala呢？笔者个人以为不必，只要你有一些Java或C++编程语言的基础，就可以开始看Spark源码，遇到不懂的地方再去学习，效率反而会大大提高，做到有的放矢。将学习中遇到的知识点，从函数式编程、

泛型编程、面向对象、并行编程等几个方面去整理归纳，这样能够快速将Scala语言的框架勾勒出来。

本书第1章和第2章简要介绍了大数据分析技术的产生背景和演进过程；第3～5章详细分析了Spark Core中的作业规划、提交及任务执行等内容，对于要深刻把握Spark实现机理的读者来说，这几章值得反复阅读；第6～9章就Spark提供的高级Lib库进行了简要的分析，分析的思路是解决的主要问题是什么、解决的方案是如何产生的，以及方案是如何通过代码来具体实现的。

在对源码有了一定的分析和掌握之后，再回过头来看一下Spark相关的论文，这时候对论文的理解可能会更顺畅。

Spark的整体框架非常庞大，涵盖的范围也很广，随着笔者在工作中使用得越来越具体，这样的感受也越来越深。另外，必须要说对于Spark来说，笔者所做的分析实在有限，个中错误在所难免，读者诸君还请多多谅解。

在本书成稿期间，电子工业出版社的付睿编辑和李云静编辑给出了极为详细的改进意见，在这里表示衷心的感谢。最后感谢家人的支持和鼓励，亲爱的老婆和懂事的儿子给了笔者坚持的理由和勇气。

许　鹏

2015年2月

目录

第一部分

Spark 概述

第1章

初识Spark

“物有本末，事有终始，知所先后，则近道矣。”

《大学》

1.1 大数据和Spark

大数据和大数据分析是目前IT领域最炙手可热的概念，不谈一下自己对大数据的看法都不好意思说自己是在IT圈混的。

那么大数据从何而来，大量的数据到底是如何产生的呢？这些问题鲜有人来回答。

同样对于大数据分析来说，也有不少问题：（1）对大数据要做哪些分析；（2）相对于小规模数据来说，分析中又有哪些不同；（3）这些不同会给理论研究和工程实施方面带来哪些难点。

谈到大数据，不由会想到Hadoop。现在Hadoop的流行已有相当一段时间了，大数据和Hadoop之间的关系如何，Hadoop解决了大数据中的哪些问题？

本书的主角是Spark，离开主角不提，却一开始聊什么大数据和Hadoop的关系，有没有搞错啊？

其实不然，要讲清楚Spark的由来，必然要涉及这几层关系：

（1） 大数据的由来。

（2） 大数据和Hadoop。

（3） Hadoop的不足及Spark的诞生。

任何工具的诞生都会涉及以下几个基本问题：（1） 现实问题的产生；（2） 理论模型的提出；（3） 工程实现。

1.1.1 大数据的由来

正所谓“物有本末，事有终始”，要想深入地搞清楚一件事情，理一理它的由来必定会有不少收获。

大数据是一个相对的概念，这个“大”是相对于“小”来说的，一般认为大数据具有如下3个特点。

- **Volume**：数据规模大。
- **Velocity**：处理速度快，时效性要求比较高。
- **Variety**：数据有丰富的多样性。

当然也有一种说法是4V，即加上一个Value。

以Google为例来说明大数据的产生。在互联网刚兴起的日子，尽管还是一种静态网页的展现方式，但对于普罗大众来说，却找到了一种很方便表达自己想法的途径。于是网页规模日趋扩大，内容也日渐丰富。

搜索引擎应时而起，面对这么多的网页、这么多的信息，如何快速找到自己感兴趣的内容呢？对于这个问题的解答，一种非常直观的思路就是先将网页抓取、存储起来，再按关键词进行查询。Yahoo和Google也是这么做的。

数以亿计的网页在抓取下来之后就变成了一个规模巨大的数据集。

网页抓取带来的第一个技术难题就是如何将抓取结果合理地存储起来。

1.1.2 大数据的分析

有数据就会有分析。

为了让数据产生更大的价值，需要对其进行相应的分析。这个时候就会遇到许多技术上的实际问题。

比如，数据规模大到一台机器无法处理时，如何在有限的时间内对整个数据集进行遍历及分析？

1.1.3 Hadoop

针对上述在数据存储和数据分析方面的技术挑战，Google抛出了自己的解决方案。即常说的三大论文：（1） MapReduce；（2） GFS；（3） BigTable。

这三大论文各自解决哪方面的问题呢？具体如下所示。

- **MapReduce**：计算框架。
- **GFS**：数据存储。
- **BigTable**：NoSQL始祖。

Hadoop是根据GFS和MapReduce两大论文所做的开源实现。讲到这里你大概明白了Hadoop所要解决的两大主要问题了：（1） 数据存储；（2） 分布式计算框架。

基于MapReduce这样一个计算框架对数据做哪些方面的分析，则是各显神通了，如Hive、Pig。这也是Hadoop生态圈异常庞杂的一个主因。

Hadoop生态圈非常庞大，看一看图1.1，会留下一个直观的印象。

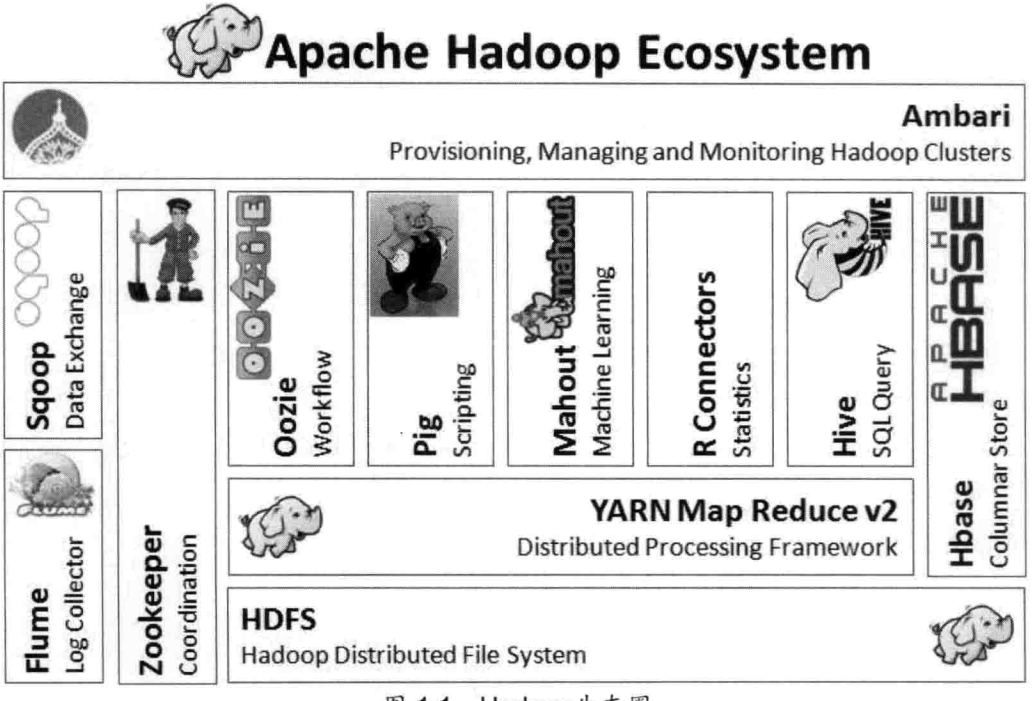

图 1.1 Hadoop生态圈

YARN是Hadoop2和Hadoop1的最大区别，将集群管理本身独立出来，而计算模型则更加专注于问题本身。

1.1.4　Spark简介

到了这里，终于可以讲一讲Spark了。这跟小说没什么分别，没有哪一部小说一开始就让主角上台表演的，精彩总是要慢慢地铺陈开来才行。

Spark由UC Berkeley的AMPLab出品，主要创作者是Matei Zaharia。目前Spark已经成为Apache的顶级项目，这也是称为Apache Spark的原因。

参照上面提到的Hadoop生态圈，问到的第一个问题就是Spark 处在哪一层，它要解决哪方面的问题呢？

好的，采用这样的思维方式，确定参考体系，然后找到问题的定位，将笛卡儿坐标在实际思维中加以运用。

在Hadoop的整个生态系统中，Spark和MapReduce在同一个层级，即主要解决分布式计算框架的问题。

与MapReduce提供的编程模型相比，Spark具有如下两个鲜明的特点：（1） 计算更为快速，速度可以提高10到100倍不等；（2） 计算过程中，如果某一节点出现问题，事件重演的代价远远小于MapReduce。

Spark是如何达到上述效果的，会在后续章节中一一详细讲解。

“天下武功，唯快不破”，算得快、算得准就是Spark的最大特色。

Spark用最精简的话来定义就是通用的分布式计算框架。如果非要加上一些定语的话，那就是适用于大数据的高可靠、高性能分布式并行计算框架。

从上面的简短定义可以看到Spark所要涉及的知识领域：

（1） 分布式并行处理，集群管理。
（2） 高可靠的两重含义，一是服务的有效性，二是计算结果的准确性。
（3） 高性能计算在可以接受的时间内完成。

从Spark的应用领域理论上来说，原先基于MapReduce来开发的分析工具都可以基于Spark来实现，通过这样一种迁移来达到更快的处理效果。

目前Spark已经对表 1.1中的应用开发提供了支持。

表 1.1　Spark支持的应用

Streaming	流数据的实时处理
SQL	类SQL的数据分析
GraphX	常用的图计算
MLLib	机器学习算法

Spark和Hadoop有如下几重关联：

（1） Spark和Hadoop中的MapReduce处在同一层面。

（2） Spark可以部署在YARN上。

（3） Spark原生支持对HDFS文件系统的访问。

至此，基本介绍清楚了Spark因何而生。以图 1.2来做个小小的总结。

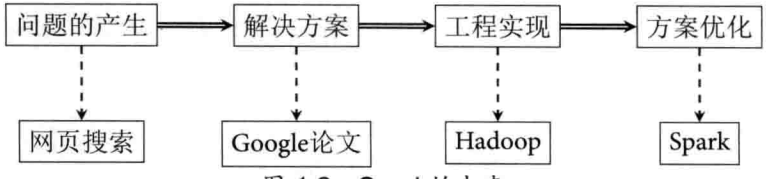

图 1.2 Spark的由来

1.2 与Spark的第一次亲密接触

简要介绍之后，来做一个简单实验，实际使用一下Spark，增加一点感性认识。

1.2.1 环境准备

在真正开始之前，我们还是先来提一提需要哪些准备工作。

- **机器配置**：Spark是非常“吃”内存的，即便是用于学习、不用于实际生产环境，建议内存也不少于4GB。
- **操作系统**：这个当然是Linux，在其所有的发行版中，建议使用Debian或Arch。
- **软件包**：JDK、Scala、Sbt、Maven。

本文后续的所有例子，都假设运行在Arch Linux之上。在Arch Linux，使用以下指令来安装上述软件包。

代码清单 1.1 安装Scala

```
pacman -S scala maven sbt
```

JDK的安装稍微复杂一些，在Linux环境下，由于License的问题，默认使用 OpenJDK。如果要使用Oracle提供的JDK的话，则在安装之外还需要做相应的配置。

安装及配置的具体步骤如下。

代码清单 1.2 安装JDK

```
yaourt -S jdk
```

上述指令将JDK安装在/opt/java目录。打开/etc/profile，修改PATH的值，同时添加JAVA_HOME变量。

代码清单 1.3　将Oracle JDK作为默认的JDK

```
PATH="/usr/local/sbin:/usr/local/bin:/usr/bin:/opt/java/bin:"
export PATH
export JAVA_HOME="/opt/java"
```

1.2.2　下载安装Spark

从官方网站下载Spark发行版，本文以Spark 1.0为例。

下载pre-built版本，选择适用于Hadoop1或Hadoop2的版本。假设下载之后文件存储在$HOME/downloads目录，按以下指令进行解压。

代码清单 1.4　下载pre-built Spark

```
cd $HOME/downloads
tar zvxf spark-1.0.0-bin-hadoop2.tgz
export SPARK_HOME=$HOME/downloads/spark-1.0.0-bin-hadoop2
```

1.2.3　Spark下的WordCount

比如想统计一下某个文件中含有的单词数，WordCount就是一件经常发生的事情。只要文件不是太大，用简单的awk就可以搞定了。下面是用awk来进行单词个数统计的脚本。

代码清单 1.5　workdcount.awk

```
{
  for (i = 1; i<=NF; i++)
    freq[$i]++
}
END{
  for (word in freq)
  printf "%s\t%d\n",word,freq[word]
}
```

将上述脚本保存到文件wordcount.awk，使用该脚本很简单。