HZ BOOKS
华章科技

根据最新技术版本，系统、全面、详细讲解Spark的各项功能使用、原理机制、技术细节、应用方法、性能优化，以及BDAS生态系统的相关技术

大数据技术丛书

Data Processing with Spark
Technology, Application and Performance Optimization

Spark大数据处理

技术、应用与性能优化

高彦杰◎著

机械工业出版社
China Machine Press

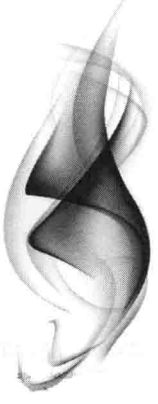
技术丛书

Data Processing with Spark
Technology, Application and Performance Optimization

Spark大数据处理

技术、应用与性能优化

高彦杰◎著



机械工业出版社
China Machine Press

图书在版编目（CIP）数据

Spark 大数据处理：技术、应用与性能优化 / 高彦杰著．—北京：机械工业出版社，2014.11
（大数据技术丛书）

ISBN 978-7-111-48386-1

I. S… II. 高… III. 数据处理软件 IV. TP274

中国版本图书馆 CIP 数据核字（2014）第 246890 号

Spark 大数据处理：技术、应用与性能优化

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：高婧雅　　责任校对：殷　虹
印　　刷：三河市宏图印务有限公司　　版　　次：2014 年 11 月第 1 版第 1 次印刷
开　　本：186mm × 240mm　1/16　　印　　张：16.75
书　　号：ISBN 978-7-111-48386-1　　定　　价：59.00 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　88361066　　投稿热线：（010）88379604
购书热线：（010）68326294　88379649　68995259　　读者信箱：hzjsj@hzbook.com

Preface 前 言

Spark 是发源于美国加州大学伯克利分校 AMPLab 的大数据分析平台，它立足于内存计算，从多迭代批量处理出发，兼顾数据仓库、流处理和图计算等多种计算范式，是大数据系统领域的全栈计算平台。Spark 当下已成为 Apache 基金会的顶级开源项目，拥有庞大的社区支持，技术也逐渐走向成熟。

为什么要写这本书

大数据还在如火如荼地发展着，突然之间，Spark 就火了。还记得最开始接触 Spark 技术时资料匮乏，只有官方文档和源码可以作为研究学习的资料。写一本 Spark 系统方面的技术书籍，是我持续了很久的一个想法。由于学习和工作较为紧张，最初只是通过几篇笔记在博客中分享自己学习 Spark 过程的点滴，但是随着时间的推移，笔记不断增多，最终还是打算将笔记整理成书，也算是一个总结和分享。

在国外 Yahoo!、Intel、Amazon、Cloudera 等公司率先应用并推广 Spark 技术，在国内淘宝、腾讯、网易、星环等公司敢为人先，并乐于分享。在随后的发展中，IBM、MapR、Hortonworks、微策略等公司纷纷将 Spark 融进现有解决方案，并加入 Spark 阵营。Spark 在工业界的应用也呈星火燎原之势。

随着 Spark 技术在国内的大范围落地、Spark 中国峰会的召开，及各地 meetup 的火爆举行，开源软件 Spark 也因此水涨船高。随着大数据相关技术和产业的逐渐成熟，公司生产环境往往需要同时进行多种类型的大数据分析作业：批处理、各种机器学习、流式计算、图计算、SQL 查询等。在 Spark 出现前，要在一个平台内同时完成以上数种大数据分析任务，就不得不与多套独立的系统打交道，这需要系统间进行代价较大的数据转储，但是这无疑会增加运维负担。

在 1 年之前，关注 Spark 的人和公司不多，由于它包含的软件种类多，版本升级较快，

技术较为新颖，初学者难以在有限的时间内快速掌握 Spark 蕴含的价值。同时国内缺少一本实践与理论相结合的 Spark 书籍，很多 Spark 初学者和开发人员只能参考网络上零星的 Spark 技术相关博客，自己一点一滴地阅读源码和文档，缓慢地学习 Spark。本书也正是为了解决上面的问题而编写的。

本书从一个系统化的视角，秉承大道至简的主导思想，介绍 Spark 中最值得关注的内容，讲解 Spark 部署、开发实战，并结合 Spark 的运行机制及拓展，帮读者开启 Spark 技术之旅。

本书特色

本书是国内首本系统讲解 Spark 编程实战的书籍，涵盖 Spark 技术的方方面面。

1）对 Spark 的架构、运行机制、系统环境搭建、测试和调优进行深入讲解，以期让读者知其所以然。讲述 Spark 最核心的技术内容，以激发读者的联想，进而衍化至繁。

2）实战部分不但给出编程示例，还给出可拓展的应用场景。

3）剖析 BDAS 生态系统的主要组件的原理和应用，让读者充分了解 Spark 生态系统。

本书的理论和实战安排得当，突破传统讲解方式，使读者读而不厌。

本书中一些讲解实操部署和示例的章节，比较适合作为运维和开发人员工作时手边的书；运行机制深入分析方面的章节，比较适合架构师和 Spark 研究人员，可帮他们拓展解决问题的思路。

读者对象

- ❑ Spark 初学者
- ❑ Spark 二次开发人员
- ❑ Spark 应用开发人员
- ❑ Spark 运维工程师
- ❑ 开源软件爱好者
- ❑ 其他对大数据技术感兴趣的人员

如何阅读本书

本书分为两大部分，共计 9 章内容。

第 1 章 从 Spark 概念出发，介绍了 Spark 的来龙去脉，阐述 Spark 生态系统全貌。

第 2 章 详细介绍了 Spark 在 Linux 集群和 Windows 上如何进行部署和安装。

第 3 章 详细介绍了 Spark 的计算模型，RDD 的概念与原理，RDD 上的函数算子的原理

和使用，广播和累加变量。

第 4 章 详细介绍了 Spark 应用执行机制、Spark 调度与任务分配、Spark I/O 机制、Spark 通信模块、容错机制、Shuffle 机制，并对 Spark 机制原理进行了深入剖析。

第 5 章 从实际出发，详细介绍了如何在 Intellij、Eclipse 中配置开发环境，如何使用 SBT 构建项目，如何使用 SparkShell 进行交互式分析、远程调试和编译 Spark 源码，以及如何构建 Spark 源码阅读环境。

第 6 章 由浅入深，详细介绍了 Spark 的编程案例，通过 WordCount、Top K 到倾斜连接等，以帮助读者快速掌握开发 Spark 程序的技巧。

第 7 章 展开介绍了主流的大数据 Benchmark 的原理，并对比了 Benchmark 优劣势，指导 Spark 系统性能测试和性能问题诊断。

第 8 章 围绕 Spark 生态系统，介绍了 Spark 之上的 SQL on Spark、Spark Streaming、GraphX、MLlib 的原理和应用。

第 9 章 详细介绍了如何对 Spark 进行性能调优，以及调优方法的原理。

如果您是一位有着一定经验的资深开发人员，能够理解 Spark 的相关基础知识和使用技巧，那么可以直接阅读 4 章、7 章、8 章、9 章。如果你是一名初学者，请一定从第 1 章的基础知识开始学起。

勘误和支持

由于笔者的水平有限，编写时间仓促，书中难免会出现一些错误或者不准确的地方，恳请读者批评指正。如果您有更多的宝贵意见，欢迎访问我的个人 Github 上的 Spark 大数据处理专版：https://github.com/YanjieGao/SparkInAction，您可以将书中的错误提交 PR 或者进行评论，我会尽量在线上为读者提供最满意的解答。您也可以通过微博 @ 高彦杰 gyj、微信公共号 @Spark 大数据、博客 http://blog.csdn.net/gaoyanjie55 或者邮箱 gaoyanjie55@163.com 联系到我，期待能够得到读者朋友们的真挚反馈，在技术之路上互勉共进。

致谢

感谢中国人民大学的杜小勇老师、何军老师、陈跃国老师，是老师们将我带进大数据技术领域，教授我专业知识与学习方法，并在每一次迷茫时给予我鼓励与支持。

感谢微软亚洲研究院的闫莺老师和其他老师及同事，在实习工作中给予我的帮助和指导。

感谢 IBM 中国研究院的陈冠诚老师和其他老师及同事，在实习工作中给予我的帮助和指导。

感谢连城、明风、Daoyuan Wang 等大牛以及 Michael Armbrust、Reynold Xin、Sean Owen

等多位社区大牛，在开发和技术学习中对我的点拨和指导，以及社区的各位技术专家们的博客文章。本书多处引用了他们的观点和思想。

感谢机械工业出版社华章公司的首席策划杨福川和编辑高婧雅，在近半年的时间中始终支持我的写作，给我鼓励和帮助，引导我顺利完成全部书稿。

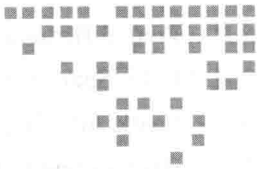
特别致谢

最后，我要特别感谢我的女友蒋丹彤对我的支持，我为写作这本书，牺牲了很多陪伴你的时间。同时也要感谢你花了很大的精力帮助我进行书稿校对。正因为有了你的付出与支持，我才能坚持写下去。

感谢我的父母、姐姐，有了你们的帮助和支持，我才有时间和精力去完成全部写作。

谨以此书献给我最亲爱的家人，以及众多热爱大数据技术的朋友们！

Contents 目 录



第 1 章 Chapter 1

Spark 简 介

本章主要介绍 Spark 大数据计算框架、架构、计算模型和数据管理策略及 Spark 在工业界的应用。围绕 Spark 的 BDAS 项目及其子项目进行了简要介绍。目前，Spark 生态系统已经发展成为一个包含多个子项目的集合，其中包含 SparkSQL、Spark Streaming、GraphX、MLlib 等子项目，本章只进行简要介绍，后续章节再详细阐述。

1.1 Spark 是什么

Spark 是基于内存计算的大数据并行计算框架。Spark 基于内存计算，提高了在大数据环境下数据处理的实时性，同时保证了高容错性和高可伸缩性，允许用户将 Spark 部署在大量廉价硬件之上，形成集群。

Spark 于 2009 年诞生于加州大学伯克利分校 AMPLab。目前，已经成为 Apache 软件基金会旗下的顶级开源项目。下面是 Spark 的发展历程。

1. Spark 的历史与发展

- 2009 年：Spark 诞生于 AMPLab。
- 2010 年：开源。
- 2013 年 6 月：Apache 孵化器项目。
- 2014 年 2 月：Apache 顶级项目。
- 2014 年 2 月：大数据公司 Cloudera 宣称加大 Spark 框架的投入来取代 MapReduce。
- 2014 年 4 月：大数据公司 MapR 投入 Spark 阵营，Apache Mahout 放弃 MapReduce，

将使用 Spark 作为计算引擎。

- 2014 年 5 月：Pivotal Hadoop 集成 Spark 全栈。
- 2014 年 5 月 30 日：Spark 1.0.0 发布。
- 2014 年 6 月：Spark 2014 峰会在旧金山召开。
- 2014 年 7 月：Hive on Spark 项目启动。

目前 AMPLab 和 Databricks 负责整个项目的开发维护，很多公司，如 Yahoo!、Intel 等参与到 Spark 的开发中，同时很多开源爱好者积极参与 Spark 的更新与维护。

AMPLab 开发以 Spark 为核心的 BDAS 时提出的目标是：one stack to rule them all，也就是说在一套软件栈内完成各种大数据分析任务。相对于 MapReduce 上的批量计算、迭代型计算以及基于 Hive 的 SQL 查询，Spark 可以带来上百倍的性能提升。目前 Spark 的生态系统日趋完善，Spark SQL 的发布、Hive on Spark 项目的启动以及大量大数据公司对 Spark 全栈的支持，让 Spark 的数据分析范式更加丰富。

2. Spark 之于 Hadoop

更准确地说，Spark 是一个计算框架，而 Hadoop 中包含计算框架 MapReduce 和分布式文件系统 HDFS，Hadoop 更广泛地说还包括在其生态系统上的其他系统，如 Hbase、Hive 等。

Spark 是 MapReduce 的替代方案，而且兼容 HDFS、Hive 等分布式存储层，可融入 Hadoop 的生态系统，以弥补缺失 MapReduce 的不足。

Spark 相比 Hadoop MapReduce 的优势[⊖]如下。

（1）中间结果输出

基于 MapReduce 的计算引擎通常会将中间结果输出到磁盘上，进行存储和容错。出于任务管道承接的考虑，当一些查询翻译到 MapReduce 任务时，往往会产生多个 Stage，而这些串联的 Stage 又依赖于底层文件系统（如 HDFS）来存储每一个 Stage 的输出结果。

Spark 将执行模型抽象为通用的有向无环图执行计划（DAG），这可以将多 Stage 的任务串联或者并行执行，而无须将 Stage 中间结果输出到 HDFS 中。类似的引擎包括 Dryad、Tez。

（2）数据格式和内存布局

由于 MapReduce Schema on Read 处理方式会引起较大的处理开销。Spark 抽象出分布式内存存储结构弹性分布式数据集 RDD，进行数据的存储。RDD 能支持粗粒度写操作，但对于读取操作，RDD 可以精确到每条记录，这使得 RDD 可以用来作为分布式索引。Spark

⊖ 参见论文：Reynold Shi Xin，Joshua Rosen，Matei Zaharia，Michael Franklin，Scott Shenker，Ion Stoica Shark: SQL and Rich Analytics at Scale。

的特性是能够控制数据在不同节点上的分区，用户可以自定义分区策略，如 Hash 分区等。Shark 和 Spark SQL 在 Spark 的基础之上实现了列存储和列存储压缩。

（3）执行策略

MapReduce 在数据 Shuffle 之前花费了大量的时间来排序，Spark 则可减轻上述问题带来的开销。因为 Spark 任务在 Shuffle 中不是所有情景都需要排序，所以支持基于 Hash 的分布式聚合，调度中采用更为通用的任务执行计划图（DAG），每一轮次的输出结果在内存缓存。

（4）任务调度的开销

传统的 MapReduce 系统，如 Hadoop，是为了运行长达数小时的批量作业而设计的，在某些极端情况下，提交一个任务的延迟非常高。

Spark 采用了事件驱动的类库 AKKA 来启动任务，通过线程池复用线程来避免进程或线程启动和切换开销。

3. Spark 能带来什么

Spark 的一站式解决方案有很多的优势，具体如下。

（1）打造全栈多计算范式的高效数据流水线

Spark 支持复杂查询。在简单的“map”及“reduce”操作之外，Spark 还支持 SQL 查询、流式计算、机器学习和图算法。同时，用户可以在同一个工作流中无缝搭配这些计算范式。

（2）轻量级快速处理

Spark 1.0 核心代码只有 4 万行。这是由于 Scala 语言的简洁和丰富的表达力，以及 Spark 充分利用和集成 Hadoop 等其他第三方组件，同时着眼于大数据处理，数据处理速度是至关重要的，Spark 通过将中间结果缓存在内存减少磁盘 I/O 来达到性能的提升。

（3）易于使用，Spark 支持多语言

Spark 支持通过 Scala、Java 及 Python 编写程序，这允许开发者在自己熟悉的语言环境下进行工作。它自带了 80 多个算子，同时允许在 Shell 中进行交互式计算。用户可以利用 Spark 像书写单机程序一样书写分布式程序，轻松利用 Spark 搭建大数据内存计算平台并充分利用内存计算，实现海量数据的实时处理。

（4）与 HDFS 等存储层兼容

Spark 可以独立运行，除了可以运行在当下的 YARN 等集群管理系统之外，它还可以读取已有的任何 Hadoop 数据。这是个非常大的优势，它可以运行在任何 Hadoop 数据源上，如 Hive、HBase、HDFS 等。这个特性让用户可以轻易迁移已有的持久化层数据。

（5）社区活跃度高

Spark 起源于 2009 年，当下已有超过 50 个机构、260 个工程师贡献过代码。开源系统

的发展不应只看一时之快，更重要的是支持一个活跃的社区和强大的生态系统。

同时我们也应该看到 Spark 并不是完美的，RDD 模型适合的是粗粒度的全局数据并行计算。不适合细粒度的、需要异步更新的计算。对于一些计算需求，如果要针对特定工作负载达到最优性能，还是需要使用一些其他的大数据系统。例如，图计算领域的 GraphLab 在特定计算负载性能上优于 GraphX，流计算中的 Storm 在实时性要求很高的场合要比 Spark Streaming 更胜一筹。

随着 Spark 发展势头日趋迅猛，它已被广泛应用于 Yahoo!、Twitter、阿里巴巴、百度、网易、英特尔等各大公司的生产环境中。

1.2 Spark 生态系统 BDAS

目前，Spark 已经发展成为包含众多子项目的大数据计算平台。伯克利将 Spark 的整个生态系统称为伯克利数据分析栈（BDAS）。其核心框架是 Spark，同时 BDAS 涵盖支持结构化数据 SQL 查询与分析的查询引擎 Spark SQL 和 Shark，提供机器学习功能的系统 MLbase 及底层的分布式机器学习库 MLlib、并行图计算框架 GraphX、流计算框架 Spark Streaming、采样近似计算查询引擎 BlinkDB、内存分布式文件系统 Tachyon、资源管理框架 Mesos 等子项目。这些子项目在 Spark 上层提供了更高层、更丰富的计算范式。

图 1-1 为 BDAS 的项目结构图。

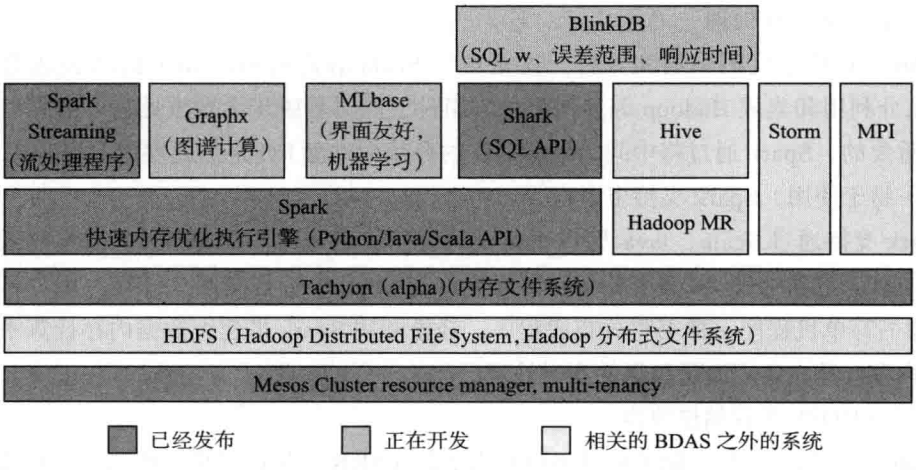


图 1-1 伯克利数据分析栈（BDAS）项目结构图

下面对 BDAS 的各个子项目进行更详细的介绍。

（1）Spark

Spark 是整个 BDAS 的核心组件，是一个大数据分布式编程框架，不仅实现了

MapReduce 的算子 map 函数和 reduce 函数及计算模型，还提供更为丰富的算子，如 filter、join、groupByKey 等。Spark 将分布式数据抽象为弹性分布式数据集（RDD），实现了应用任务调度、RPC、序列化和压缩，并为运行在其上的上层组件提供 API。其底层采用 Scala 这种函数式语言书写而成，并且所提供的 API 深度借鉴 Scala 函数式的编程思想，提供与 Scala 类似的编程接口。

图 1-2 为 Spark 的处理流程（主要对象为 RDD）。

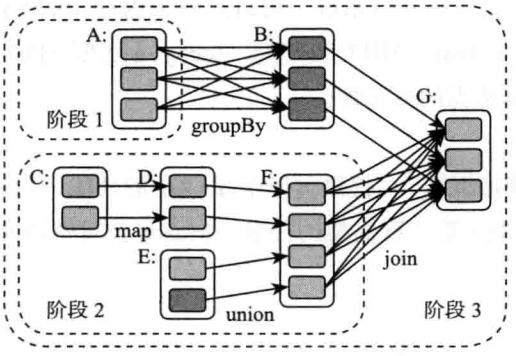


图 1-2 Spark 的任务处理流程图

Spark 将数据在分布式环境下分区，然后将作业转化为有向无环图（DAG），并分阶段进行 DAG 的调度和任务的分布式并行处理。

（2）Shark

Shark 是构建在 Spark 和 Hive 基础之上的数据仓库。目前，Shark 已经完成学术使命，终止开发，但其架构和原理仍具有借鉴意义。它提供了能够查询 Hive 中所存储数据的一套 SQL 接口，兼容现有的 Hive QL 语法。这样，熟悉 Hive QL 或者 SQL 的用户可以基于 Shark 进行快速的 Ad-Hoc、Reporting 等类型的 SQL 查询。Shark 底层复用 Hive 的解析器、优化器以及元数据存储和序列化接口。Shark 会将 Hive QL 编译转化为一组 Spark 任务，进行分布式运算。

（3）Spark SQL

Spark SQL 提供在大数据上的 SQL 查询功能，类似于 Shark 在整个生态系统的角色，它们可以统称为 SQL on Spark。之前，Shark 的查询编译和优化器依赖于 Hive，使得 Shark 不得不维护一套 Hive 分支，而 Spark SQL 使用 Catalyst 做查询解析和优化器，并在底层使用 Spark 作为执行引擎实现 SQL 的 Operator。用户可以在 Spark 上直接书写 SQL，相当于为 Spark 扩充了一套 SQL 算子，这无疑更加丰富了 Spark 的算子和功能，同时 Spark SQL 不断兼容不同的持久化存储（如 HDFS、Hive 等），为其发展奠定广阔的空间。

（4）Spark Streaming

Spark Streaming 通过将流数据按指定时间片累积为 RDD，然后将每个 RDD 进行批处

理，进而实现大规模的流数据处理。其吞吐量能够超越现有主流流处理框架 Storm，并提供丰富的 API 用于流数据计算。

（5）GraphX

GraphX 基于 BSP 模型，在 Spark 之上封装类似 Pregel 的接口，进行大规模同步全局的图计算，尤其是当用户进行多轮迭代时，基于 Spark 内存计算的优势尤为明显。

（6）Tachyon

Tachyon 是一个分布式内存文件系统，可以理解为内存中的 HDFS。为了提供更高的性能，将数据存储剥离 Java Heap。用户可以基于 Tachyon 实现 RDD 或者文件的跨应用共享，并提供高容错机制，保证数据的可靠性。

（7）Mesos

Mesos 是一个资源管理框架[⊖]，提供类似于 YARN 的功能。用户可以在其中插件式地运行 Spark、MapReduce、Tez 等计算框架的任务。Mesos 会对资源和任务进行隔离，并实现高效的资源任务调度。

（8）BlinkDB

BlinkDB 是一个用于在海量数据上进行交互式 SQL 的近似查询引擎。它允许用户通过在查询准确性和查询响应时间之间做出权衡，完成近似查询。其数据的精度被控制在允许的误差范围内。为了达到这个目标，BlinkDB 的核心思想是：通过一个自适应优化框架，随着时间的推移，从原始数据建立并维护一组多维样本；通过一个动态样本选择策略，选择一个适当大小的示例，然后基于查询的准确性和响应时间满足用户查询需求。

1.3 Spark 架构

从上文介绍可以看出，Spark 是整个 BDAS 的核心。生态系统中的各个组件通过 Spark 来实现对分布式并行任务处理的程序支持。

1. Spark 的代码结构

图 1-3 展示了 Spark-1.0 的代码结构和代码量（不包含 Test 和 Sample 代码），读者可以通过代码架构对 Spark 的整体组件有一个初步了解，正是这些代码模块构成了 Spark 架构中的各个组件，同时读者可以通过代码模块的脉络阅读与剖析源码，这对于了解 Spark 的架构和实现细节都是很有帮助的。

下面对图 1-3 中的各模块进行简要介绍。

⊖ Spark 自带的资源管理框架是 Standalone。

scheduler：文件夹中含有负责整体的 Spark 应用、任务调度的代码。

broadcast：含有 Broadcast(广播变量）的实现代码，API 中是 Java 和 Python API 的实现。

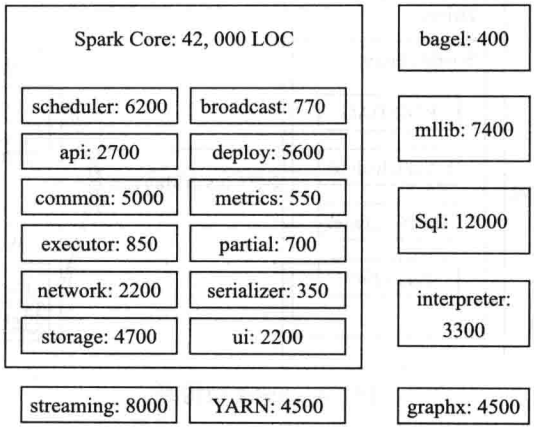


图 1-3 Spark 代码结构和代码量

deploy：含有 Spark 部署与启动运行的代码。

common：不是一个文件夹，而是代表 Spark 通用的类和逻辑实现，有 5000 行代码。

metrics：是运行时状态监控逻辑代码，Executor 中含有 Worker 节点负责计算的逻辑代码。

partial：含有近似评估代码。

network：含有集群通信模块代码。

serializer：含有序列化模块的代码。

storage：含有存储模块的代码。

ui：含有监控界面的代码逻辑。其他的代码模块分别是对 Spark 生态系统中其他组件的实现。

streaming：是 Spark Streaming 的实现代码。

YARN：是 Spark on YARN 的部分实现代码。

graphx：含有 GraphX 实现代码。

interpreter：代码交互式 Shell 的代码量为 3300 行。

mllib：代表 MLlib 算法实现的代码量。

sql 代表 Spark SQL 的代码量。

2. Spark 的架构

Spark 架构采用了分布式计算中的 Master-Slave 模型。Master 是对应集群中的含有 Master 进程的节点，Slave 是集群中含有 Worker 进程的节点。Master 作为整个集群的控制器，负责整个集群的正常运行；Worker 相当于是计算节点，接收主节点命令与进行状态汇