Apache Spark

—RDD 与 Storage 需求规格说明书

组员：

SY1506404 孟翰

SY1506409 苏若

SY1506425 李璇

SY1506406 孙敏芳

目录

版本变更历史

版本	提交日期	主要编制人	审核人	版本说明
1.0	2016.03.24	孟翰、李璇、敏芳、苏若		初始版本

1 引言

1.1 文档编写目的

以 Apache Spark 开源软件及相关资料为输入依据，在对其进行可行性研究后，对本课程实验相关问题进行进一步定位，确定获取和需求分析的方法，并对确定的软件需求进行定义和描述。之后将上述过程以可视的文档形式表达出来，输出为软件需求规格说明书。通过该文档的编写，可以保证以下目标的实现：

第一，便于开发人员进行理解和交流。

第二，反映用户问题的结构，可作为软件开发工作的基础和依据。

第三，作为确认测试和验收的依据。

1.2 系统标识

Spark 版本号： spark1.6.1

Spark 源码实现语言：Scala

可编程语言：Scala.+Scala shell、Java、Python+Scala shell

环境：jdk1.8

需求报告版本：V1.0

1.3 系统概述

1.3.1 Spark

Spark 是 UC Berkeley AMP lab 所开源的类 Hadoop MapReduce 的通用的并行计算框架，分布式资源工作交由集群管理软件（如 Mesos , Standalone ,YARN）.

从整体架构而言:

第一，Spark 提供了多种高级工具，如 Shark SQL 应用于查询、Spark Streaming 应用于流式计算、MLib 应用于机器学习、GraphX 应用于图处理。

第二，Spark 可以访问存储在 HDFS、Hbase、Cassandra、Amazon S3、本地文件系统等上的数据，并支持多种文件格式，如文本文件、序列文件及 Hadoop 的

InputFormat。

从核心模块而言：Spark 核心组件包括 RDD、Scheduler、Storage、Shuffle 四部分。在这里，主要针对需深入研究的 RDD 和 Storage 模块进行细节性概述。

1.3.2 RDD

a．数据处理模型——RDD 混合了 Iterative Algorithms，Relational Queries，Hadoop MapReduce 采用的 MapReduces 及 Storm 采用的 Stream Processing 四种模型，使得 Spark 可以应用于各种大数据处理场景。

b．5 个属性—— 一组分片（Partition）、一个计算每个分片的函数、依赖（Dependencies）、preferredLocations、partitioner。

c．两个操作——转换（transformation）和动作（action）。

d．11 种存储级别——RDD 根据 useDisk、useMemory、deserialized、off_heap、replication 五个参数的组合设置了 11 种存储级别，其中默认级别 MEMORY_ONLY。

e．两种依赖关系——窄依赖和宽依赖。

1.3.3 Storage

a．子模块 1——通信层。

在该层，采用 master-slave 结构传输控制信息和状态信息。

b．子模块 2——存储层。

该层为存储数据到 disk 或是 memory，甚至 replicate 到远端提供了相应接口。同时，提供了统一的操作类 BlockManager，以完成其他模块和 storage 模块的交互。

1.3.4 RDD 和 Storage 两模块的关系

RDD 提供了的各种 transformation 和 action 接口以实现应用，提高了抽象层次，在接口和实现上进行有效地隔离，使用户无需关心底层的实现。

但对于所操作的数据究竟放在哪里，如何存取？这是由 storage 模块来实现

和管理的。可以说，RDD 实现了用户的逻辑，而 Storage 则管理了用户的数据。举例来说，用户在实际编程中，面对的是 RDD，可以将 RDD 的数据通过调用 org.apache.spark.rdd.RDD#cache 将数据持久化，而持久化的动作都是由 Storage 模块完成的。

1.4 分析过程

在需求过程分析中，本小组首先选定Apache Spark开源软件作为分析对象。在对其特性、体系架构及源码进行综合分析后，决定对其主要的RDD和Storage模块进行分析研究，并期望在学习过程中发现其待改进之处并加以解决。

1.5 文档概述

本文档对 Apache Spark 的 RDD 及 Storage 模块的需求分析进行说明，包括这些模块的功能性需求和非功能性需求。

1.6 数字字典

数据字典名称	Scheduler
简介	Spark 组件
数据定义	Spark 的调度机制，可分为 DAGScheduler 和 TaskScheduler。

数据字典名称	Shuffle
简介	Spark 组件
数据定义	shuffle 是连接 Map 和 Reduce 之间的桥梁，Map 的输出要用到 Reduce 中必须经过 shuffle 这个环节，shuffle 的性能高低直接影响了整个程序的性能和吞吐量。

数据字典名称	RDD
简介	Spark 组件
数据定义	Resilient Distributed Datasets，是一个容错的、并行的数据结构，可以让用户显式地将数据存储到磁盘和内存中，并能控制数据的分区。

数据字典名称	Storage
简介	Spark 组件
数据定义	负责在 Spark 计算过程中，包括基于 Disk 的和基于 Memory 的数据存储模块。

数据字典名称	窄依赖
简介	RDD 间依赖关系
数据定义	一个父 RDD 最多被一个子 RDD 用。

数据字典名称	宽依赖
简介	RDD 间依赖关系
数据定义	子 RDD 的分区依赖于父 RDD 的所有分区。

数据字典名称	BlockManager
简介	Storage 模块内的类
数据定义	Storage 模块与其他模块交互最主要的类，它提供了读和 Block 的接口。

数据字典名称	Block
简介	Storage 模块存取数据的最小单位
数据定义	每一个 RDD partition 都会对应一个 Block，由唯一的标识，格式是"rdd_" + rddId + "_" + partitionId。

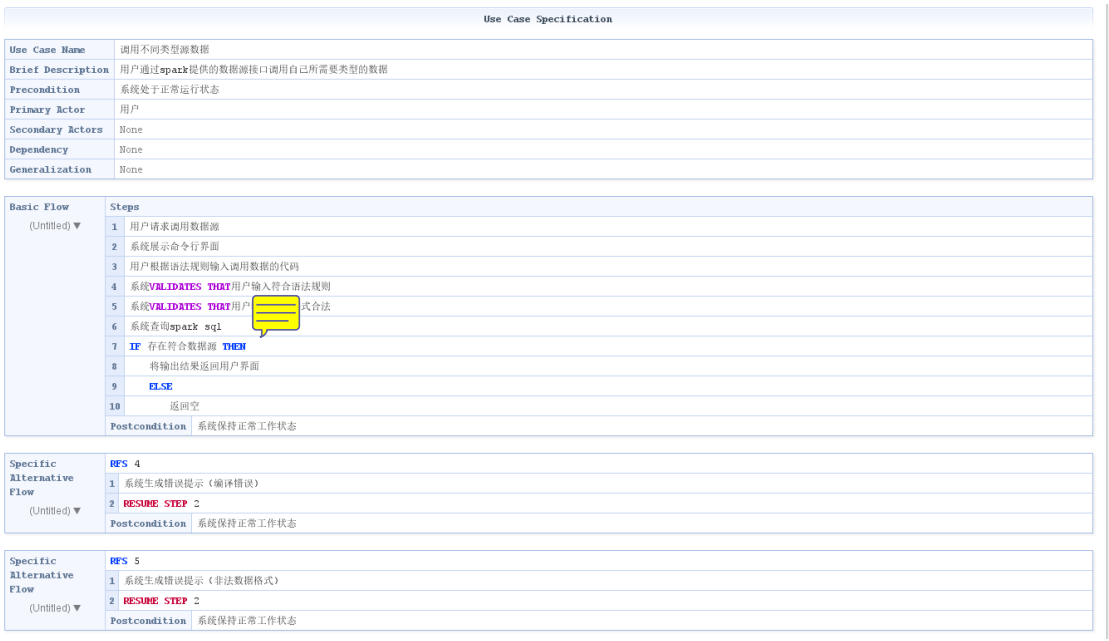
1.7 参考文献

[1] http://spark.apache.org/
[2] https://databricks.com/blog/2014/10/10/spark-petabyte-sort.html
[3] http://jerryshao.me/architecture/2013/10/08/spark-storage-module-analysis/
[4] http://jingyan.baidu.com/article/90bc8fc80960f1f653640ce0.html
[5] http://www.lxway.com/66982556.htm
[6] http://blog.csdn.net/colorant/article/details/8255958
[7] http://tech.uc.cn/?p=2116
[8] http://shiyanjun.cn/archives/744.html
[9] https://www.zhihu.com/question/36532979
[10] http://blog.csdn.net/hsluoyc/article/details/43977779
[11] http://www.chinacloud.cn/show.aspx?id=23013&cid=14
[12] 张安站. Spark 技术内幕. 机械工业出版社[M]. 2015-09.
[13] 许鹏.《Apache Spark 源码剖析》. 电子工业出版社: 2015-03-01.
[14] Ji, Changqing, et al. "Big data processing: Big challengesand opportunities."

Journal of Interconnection Networks 13.03n04 (2012).
[15] Rahimian, Fatemeh. "Gossip-Based Algorithms for InformationDissemination and Graph Clustering." (2014).

2 功能性需求

2.1 可以处理多种类型的数据



Use Case Specification

Use Case Name	调用不同类型源数据
Brief Description	用户通过spark提供的数据源接口调用自己所需要类型的数据
Precondition	系统处于正常运行状态
Primary Actor	用户
Secondary Actors	None
Dependency	None
Generalization	None

Basic Flow (Untitled)	Steps
	1 用户请求调用数据源
	2 系统展示命令行界面
	3 用户根据语法规则输入调用数据的代码
	4 系统VALIDATES THAT用户输入符合语法规则
	5 系统VALIDATES THAT用户 式合法
	6 系统查询spark sql
	7 IF 存在符合数据源 THEN
	8 将输出结果返回用户界面
	9 ELSE
	10 返回空
	Postcondition 系统保持正常工作状态

Specific Alternative Flow (Untitled)	
	RFS 4
	1 系统生成错误提示（编译错误）
	2 RESUME STEP 2
	Postcondition 系统保持正常工作状态

Specific Alternative Flow (Untitled)	
	RFS 5
	1 系统生成错误提示（非法数据格式）
	2 RESUME STEP 2
	Postcondition 系统保持正常工作状态

2.2 可应用于各种大数据处理场景

数据处理模型——RDD 混合了 Iterative Algorithms，Relational Queries，Hadoop MapReduce 采用的 MapReduces 及 Storm 采用的 Stream Processing 四种模型，使得 Spark 可以应用于各种大数据处理场景。

1. Stream 流计算

Use Case Specification	
Use Case Name	Spark Streaming流计算
Brief Description	Spark处理大规模实时流式数据
Precondition	搭建好Spark Streaming框架，Spark系统处于正常工作状态.
Primary Actor	用户
Secondary Actors	None
Dependency	None
Generalization	None

Basic Flow (Untitled) ▼	Steps
	1 用户输入实时数据源.
	2 系统VALIDATES THAT输入数据源类型为Kafk、Flume、Twitter、ZeroMQ、Kinesis或者TCP sockets之一.
	3 系统VALIDATES THAT获取到数据.
	4 系统将实时输入数据流以时间片Δt （如1秒）为单位切分成块.
	5 系统对数据进行分析.
	6 系统返回数据处理结果.
	Postcondition 系统空闲.

Specific Alternative Flow (Untitled) ▼	RFS 2
	1 输入数据源不属于Kafk、Flume、Twitter、ZeroMQ、Kinesis或者TCP sockets其中之一.
	2 系统报错，提示无法获取数据.
	Postcondition 系统显示提示信息.

Specific Alternative Flow (Untitled) ▼	RFS 3
	1 系统未获取到数据.
	2 系统提示未获取到数据.
	Postcondition 系统显示提示信息.

2. Sql 分析计算

Use Case Specification	
Use Case Name	Spark Sql分析计算
Brief Description	Spark进行Sql数据处理
Precondition	搭建好Spark Sql框架，Spark系统处于正常工作状态.
Primary Actor	用户
Secondary Actors	None
Dependency	None
Generalization	None

Basic Flow (Untitled) ▼	Steps
	1 用户在在客户端上启动spark-shell.
	2 用户创建sqlContext或者构建hiveContext.
	3 用户使用Spark Sql的数据增删改查方法进行数据操作.
	4 系统VALIDATES THAT数据存在.
	5 系统VALIDATES THAT数据格式符合要求.
	6 系统返回查询结果.
	Postcondition 系统空闲.

Specific Alternative Flow (Untitled) ▼	RFS 4
	1 所访问的数据不存在.
	2 系统提示数据访问失败.
	Postcondition 系统显示提示信息.

Specific Alternative Flow (Untitled) ▼	RFS 5
	1 数据格式不符合Sql分析数据处理场景模式下数据格式.
	2 系统更换到其它数据处理场景.
	Postcondition 系统处于其他数据处理场景.

3. MLib 机器学习

Use Case Specification	
Use Case Name	spark MLib机器学习
Brief Description	使用Spark的机器学习处理数据
Precondition	Spark系统处于正常工作状态.
Primary Actor	用户
Secondary Actors	None
Dependency	None
Generalization	None

Basic Flow (Untitled) ▼	**Steps**	
	1	用户装载数据.
	2	系统**VALIDATES THAT**数据读取成功.
	3	用户选择机器学习算法.
	4	系统**VALIDATES THAT**系统MLib库中存在该机器学习算法的实现库.
	5	系统对数据进行迭代计算分析.
	6	系统打印输出分析结果.
	Postcondition	系统空闲.

Specific Alternative Flow (Untitled) ▼	**RFS** 2	
	1	系统读取数据失败.
	2	系统显示提示数据读取失败的信息.
	Postcondition	系统空闲.

Specific Alternative Flow (Untitled) ▼	**RFS** 4	
	1	系统调用实现库失败.
	2	系统报错，提示调用实现库失败信息.
	Postcondition	系统空闲.

4. GraphX 图计算

Use Case Specification	
Use Case Name	Spark GraphX图计算
Brief Description	使用Spark的GraphX图计算进行数据处理
Precondition	Spark系统处于正常工作状态.
Primary Actor	用户
Secondary Actors	None
Dependency	None
Generalization	None

Basic Flow (Untitled) ▼	Steps
	1 用户配置GraphX.
	2 用户将分析的数据构建图模型.
	3 系统VALIDATES THAT图模型构建成功.
	4 用户设置模型训练次数、每次的迭代次数.
	5 系统VALIDATES THAT用户设置模型训练次数有效、每次的迭代次数有效.
	6 系统进行模型训练.
	7 系统进行图计算，分析数据.
	8 系统输出数据分析结果.
	Postcondition 系统空闲.

Specific Alternative Flow (Untitled) ▼	RFS 3
	1 图模型构建失败.
	2 系统运行出错.
	Postcondition 系统空闲，等待用户修改错误.

Specific Alternative Flow (Untitled) ▼	RFS 5
	1 用户设置模型训练次数无效或者每次的迭代次数无效.
	2 系统提示用户重新设置训练次数和每次的迭代次数.
	Postcondition 系统等待.

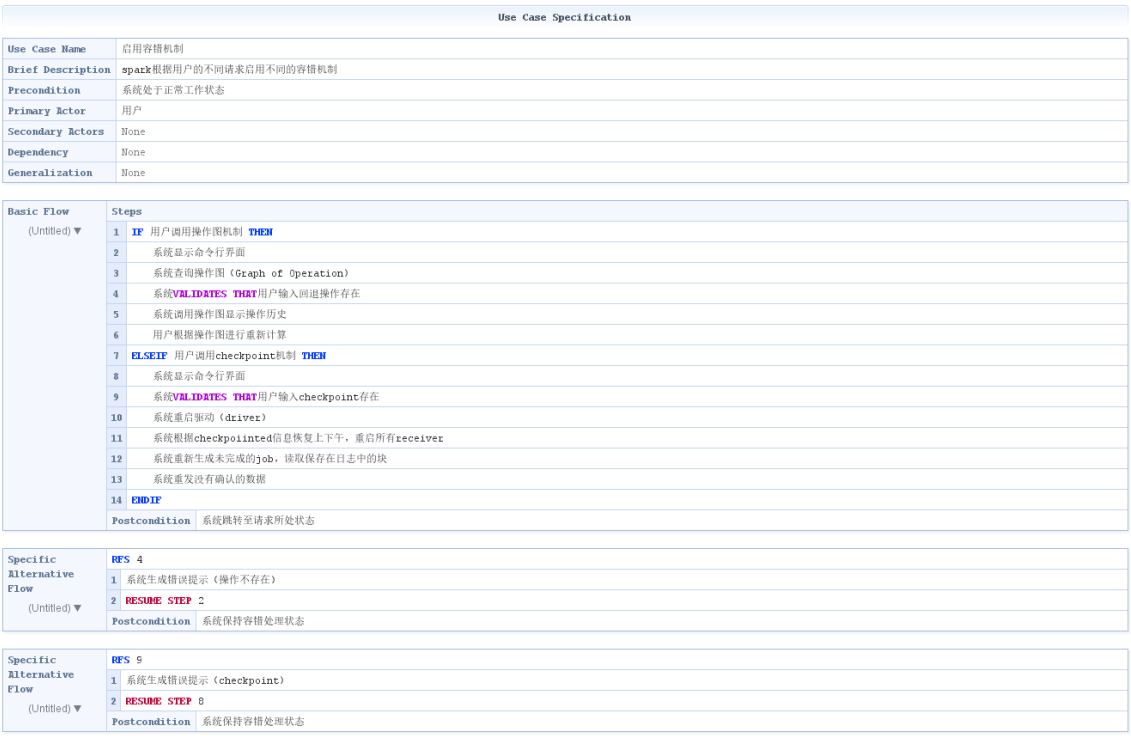
5. 场景融合：

Use Case Specification	
Use Case Name	Spark多重场景的融合
Brief Description	Spark支持多种数据处理场景的融合.
Precondition	Spark系统处于正常工作状态.
Primary Actor	用户
Secondary Actors	None
Dependency	INCLUDE USE CASE Spark Streaming流计算,INCLUDE USE CASE Spark Sql分析计算,INCLUDE USE CASE spark MLib机器学习,INCLUDE USE CASE Spark GraphX图计算
Generalization	None

Basic Flow (Untitled) ▼	Steps
	1 用户装载数据.
	2 系统分析数据类型.
	3 系统选择数据处理场景.
	4 IF 数据类型为流式数据 THEN
	5 INCLUDE USE CASE Spark Streaming流计算
	6 ELSEIF数据类型为Sql THEN
	7 INCLUDE USE CASE Spark Sql分析计算
	8 ELSEIF 数据处理方式为MLib机器学习数据处理 THEN
	9 INCLUDE USE CASE spark MLib机器学习
	10 ELSEIF 数据类型为图模型 THEN
	11 INCLUDE USE CASE Spark GraphX图计算
	12 ENDIF
	13 系统进行数据处理.
	14 系统输出数据处理结果.
	Postcondition 系统空闲.

2.3 用户可自主选择容错方式

在 RDD 计算，通过 checkpint 进行容错，做 checkpoint 有两种方式，一个是 checkpoint data，一个是 logging the updates。用户可以控制采用哪种方式来实现容错，默认是 logging the updates 方式，通过记录跟踪所有生成 RDD 的转换（transformations）也就是记录每个 RDD 的 lineage（血统）来重新计算生成丢失的分区数据。



Use Case Specification

Use Case Name	启用容错机制
Brief Description	spark根据用户的不同请求启用不同的容错机制
Precondition	系统处于正常工作状态
Primary Actor	用户
Secondary Actors	None
Dependency	None
Generalization	None

Basic Flow (Untitled)	Steps
	1 IF 用户调用操作图机制 THEN
	2 系统显示命令行界面
	3 系统查询操作图（Graph of Operation）
	4 系统VALIDATES THAT用户输入回退操作存在
	5 系统调用操作图显示操作历史
	6 用户根据操作图进行重新计算
	7 ELSEIF 用户调用checkpoint机制 THEN
	8 系统显示命令行界面
	9 系统VALIDATES THAT用户输入checkpoint存在
	10 系统重启驱动（driver）
	11 系统根据checkpoiinted信息恢复上下午，重启所有receiver
	12 系统重新生成未完成的job，读取保存在日志中的块
	13 系统重发没有确认的数据
	14 ENDIF
	Postcondition 系统跳转至请求所处状态

Specific Alternative Flow (Untitled)	RFS 4
	1 系统生成错误提示（操作不存在）
	2 RESUME STEP 2
	Postcondition 系统保持容错处理状态

Specific Alternative Flow (Untitled)	RFS 9
	1 系统生成错误提示（checkpoint）
	2 RESUME STEP 8
	Postcondition 系统保持容错处理状态

2.4 用户可以命名、物化、控制中间结果的存储、分区。

3 非功能性需求

3.1 鲁棒性

鲁棒性就是系统的健壮性，指系统或组件在面对非法输入、相连接的系统或组件故障、或非预期的运行条件下能够持续正确运行的程度。

为了保证 RDD 中数据的鲁棒性，RDD 数据集通过所谓的血统关系(Lineage)记住了它是如何从其它 RDD 中演变过来的。相比其它系统的细颗粒度的内存数

据更新级别的备份或者 LOG 机制，RDD 的 Lineage 记录的是粗颗粒度的特定数据变换（Transformation）操作（filter, map, join etc.)行为。当这个 RDD 的部分分区数据丢失时，它可以通过 Lineage 获取足够的信息来重新运算和恢复丢失的数据分区。这种粗颗粒的数据模型，限制了 Spark 的运用场合，但同时相比细颗粒度的数据模型，也带来了性能的提升。

总之，Spark 的核心思路就是将数据集缓存在内存中加快读取速度，同时用 lineage 关联的 RDD 以较小的性能代价保证数据的鲁棒性。

3.2 容错性

对于大规模数据处理来讲，容错性是其中比较重要的一环。与在软件故障或违反指定接口的情况下维持规定的性能水平的能力有关的软件属性（如离线录入支持等）叫做容错性。

一般来说，分布式数据集的容错性有两种方式：即数据检查点和记录数据的更新。我们面向的是大规模数据分析，数据检查点操作成本很高：需要通过数据中心的网络连接在机器之间复制庞大的数据集，而网络带宽往往比内存带宽低得多，同时还需要消耗更多的存储资源（在内存中复制数据可以减少需要缓存的数据量，而存储到磁盘则会拖慢应用程序）。所以，我们选择记录更新的方式。但是，如果更新太多，那么记录更新成本也不低。因此，RDD 只支持粗粒度转换，即在大量记录上执行的单个操作。将创建 RDD 的一系列转换记录下来（即 Lineage），以便恢复丢失的分区。

同时，RDD 独有的宽/窄依赖的概念，在 RDD Transformation 操作中间发生计算失败时，对容错性会有很大帮助。具体描述如下：

（1） 运算是窄依赖。只需把丢失的父 RDD 分区重算即可，这样可以大大加快场景恢复的开销。

（2） 增加检查点。当 Lineage 特别长时或者有宽依赖时，主动调用 checkpoint 把当前的数据写入稳定存储，作为检查点。

3.3 安全性

3.3.1 Web UI 安全

a．Spark 提供的 javax.servlet.filters 可以提高 Web UI 安全性,支持用户自主选择允许其他人访问自己数据的权限。具体来看，用户通过使用 javax.servlet.filters 对其他用户进行验证，一旦其他用户登录进入 Spark，系统此时会在该用户与视图访问控制列表之间进行比较分析，以确保该用户有权查看所有者用户的 UI 界面。

b．Spark 支持通过修改访问控制列表的方式来控制哪个用户可以访问、修改正在运行的 Spark 应用，其中包括终止一个应用或任务。

c．Spark 允许管理员在访问控制列表中指定哪个用户对所有应用程序总都具有查看、修改权限。

3.3.2 事件审计安全

Spark 支持事件审计安全设置。通过进行相应设置，可以设置非所有者的其他用户的使用权限。如：允许非所有者的其他用户可以在该文件夹下写，但是不能移动或者重命名文件。这样事件日志只会被 root 用户和 Spark 系统生成和修改，从而保证其安全性。

3.3.3 网络端口安全

Spark Streaming 支持大规模流式数据处理，其安全威胁与多媒体云安全有共通之处。在利用 Spark Streaming 进行多媒体数据处理时，需要进行身份认证，同时利用安全协议如 RTMP 将多媒体数据进行加密传输。

3.4 效率

Spark 的 RDD 模块的高效率主要体现在以下几方面：

1、RDD 只能从持久存储或通过 Transformations 操作产生，相比于分布式共

享内存（DSM）可以更高效实现容错，对于丢失部分数据分区只需根据它的 lineage 就可重新计算出来，而不需要做特定的 Checkpoint。

2、RDD 的数据分区特性，可以通过数据的本地性来提高性能。

3、RDD 在需要进行分区把数据分布于集群中时会根据每条记录 Key 进行分区（如 Hash 分区），保证了两个数据集在 Join 时能高效。

4、RDD 操作分为 transformation 和 action。所有 transformation 都是“惰性”的，只有在执行 action 操作之后，所有的 operation 才会被真正执行，大大提高类系统的性能。