

13 | 同样的本质，为何Spark可以更高效？

2018-11-27 李智慧

从0开始学大数据

[进入课程 >](#)



讲述：李智慧

时长 12:05 大小 5.54M



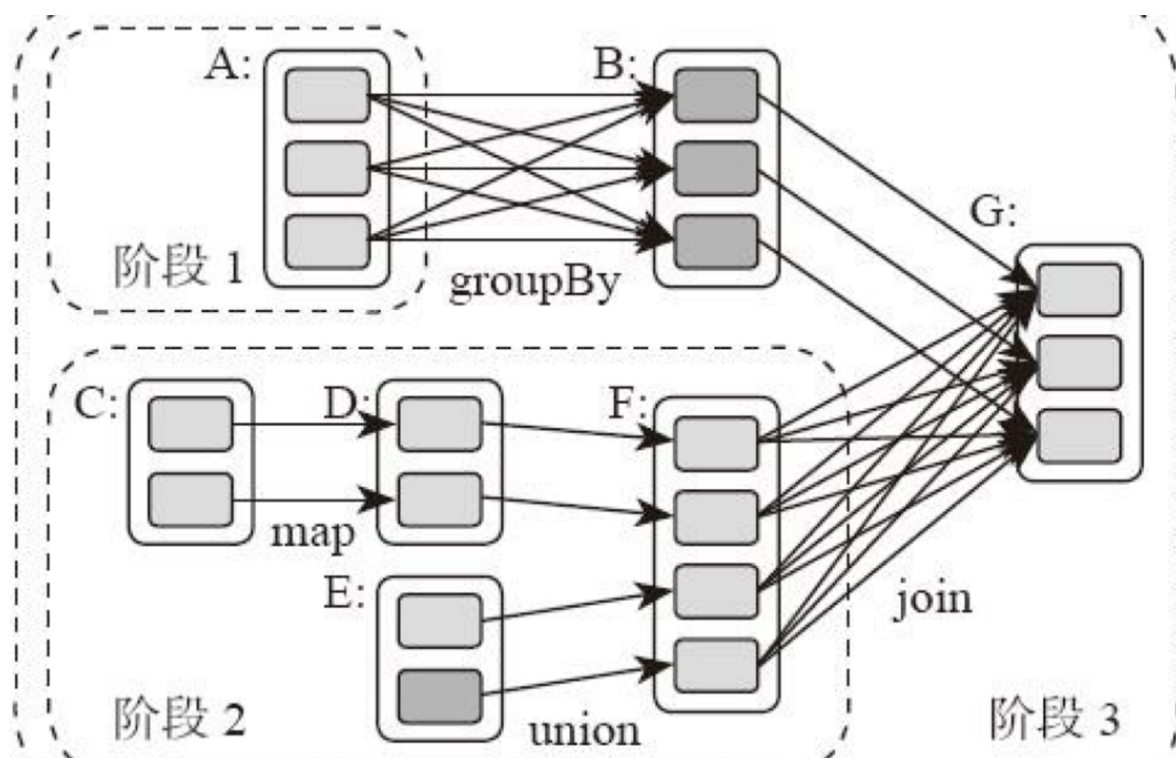
上一期我们讨论了 Spark 的编程模型，这期我们聊聊**Spark 的架构原理**。和 MapReduce 一样，**Spark 也遵循移动计算比移动数据更划算这一大数据计算基本原则**。但是和 MapReduce 僵化的 Map 与 Reduce 分阶段计算相比，Spark 的计算框架更加富有弹性和灵活性，进而有更好的运行性能。

Spark 的计算阶段

我们可以对比来看。首先和 MapReduce 一个应用一次只运行一个 map 和一个 reduce 不同，Spark 可以根据应用的复杂程度，分割成更多的计算阶段（stage），这些计算阶段组成一个有向无环图 DAG，Spark 任务调度器可以根据 DAG 的依赖关系执行计算阶段。


还记得在上一期，我举了一个比较逻辑回归机器学习性能的例子，发现 Spark 比 MapReduce 快 100 多倍。因为某些机器学习算法可能需要进行大量的迭代计算，产生数万个计算阶段，这些计算阶段在一个应用中处理完成，而不是像 MapReduce 那样需要启动数万个应用，因此极大地提高了运行效率。

所谓 DAG 也就是有向无环图，就是说不同阶段的依赖关系是有向的，计算过程只能沿着依赖关系方向执行，被依赖的阶段执行完成之前，依赖的阶段不能开始执行，同时，这个依赖关系不能有环形依赖，否则就成为死循环了。下面这张图描述了一个典型的 Spark 运行 DAG 的不同阶段。



从图上看，整个应用被切分成 3 个阶段，阶段 3 需要依赖阶段 1 和阶段 2，阶段 1 和阶段 2 互不依赖。Spark 在执行调度的时候，先执行阶段 1 和阶段 2，完成以后，再执行阶段 3。如果有更多的阶段，Spark 的策略也是一样的。只要根据程序初始化好 DAG，就建立了依赖关系，然后根据依赖关系顺序执行各个计算阶段，Spark 大数据应用的计算就完成了。

上图这个 DAG 对应的 Spark 程序伪代码如下。

 复制代码

```
1 rddB = rddA.groupBy(key)
2 rddD = rddC.map(func)
3 rddF = rddD.union(rddE)
4 rddG = rddB.join(rddF)
```

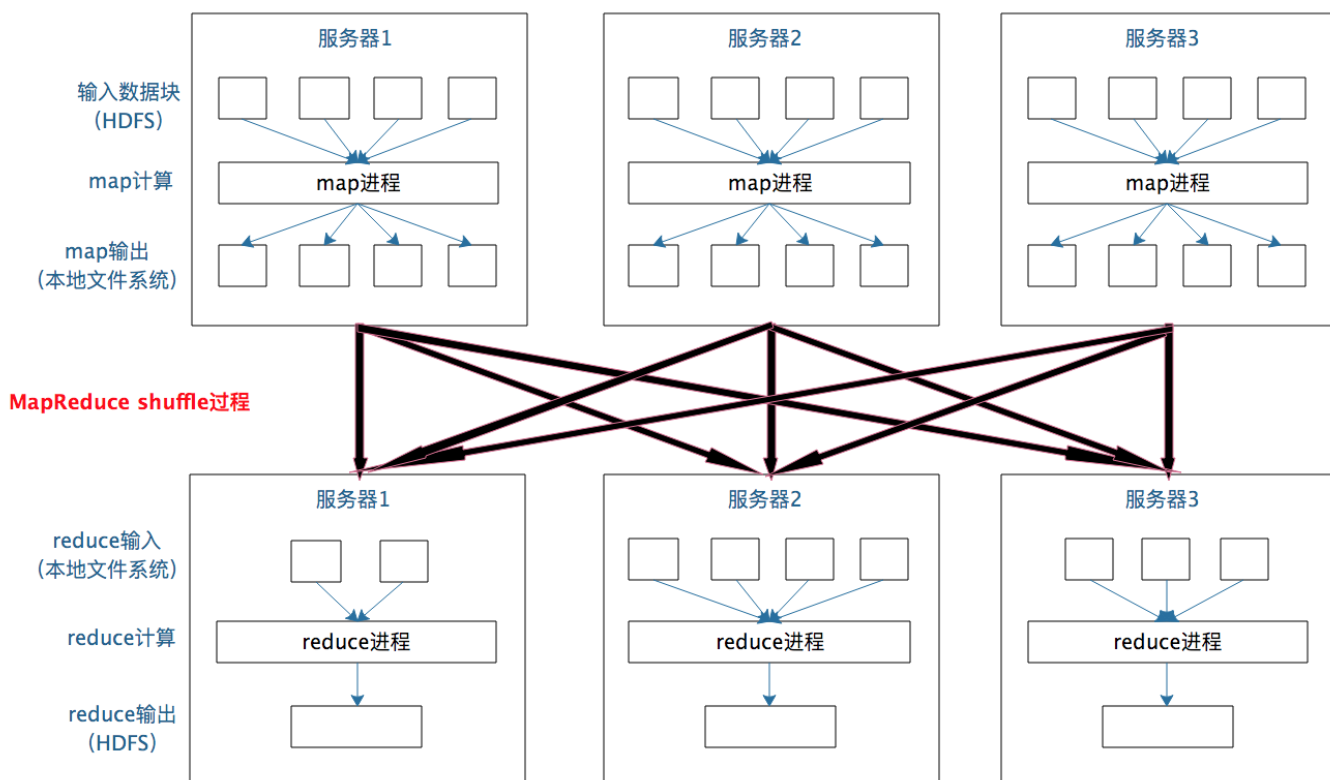
所以，你可以看到 Spark 作业调度执行的核心是 DAG，有了 DAG，整个应用就被切分成哪些阶段，每个阶段的依赖关系也就清楚了。之后再根据每个阶段要处理的数据量生成相应的任务集合（TaskSet），每个任务都分配一个任务进程去处理，Spark 就实现了大数据的分布式计算。

具体来看的话，负责 Spark 应用 DAG 生成和管理的组件是 DAGScheduler，DAGScheduler 根据程序代码生成 DAG，然后将程序分发到分布式计算集群，按计算阶段的先后关系调度执行。

那么 Spark 划分计算阶段的依据是什么呢？显然并不是 RDD 上的每个转换函数都会生成一个计算阶段，比如上面的例子有 4 个转换函数，但是只有 3 个阶段。

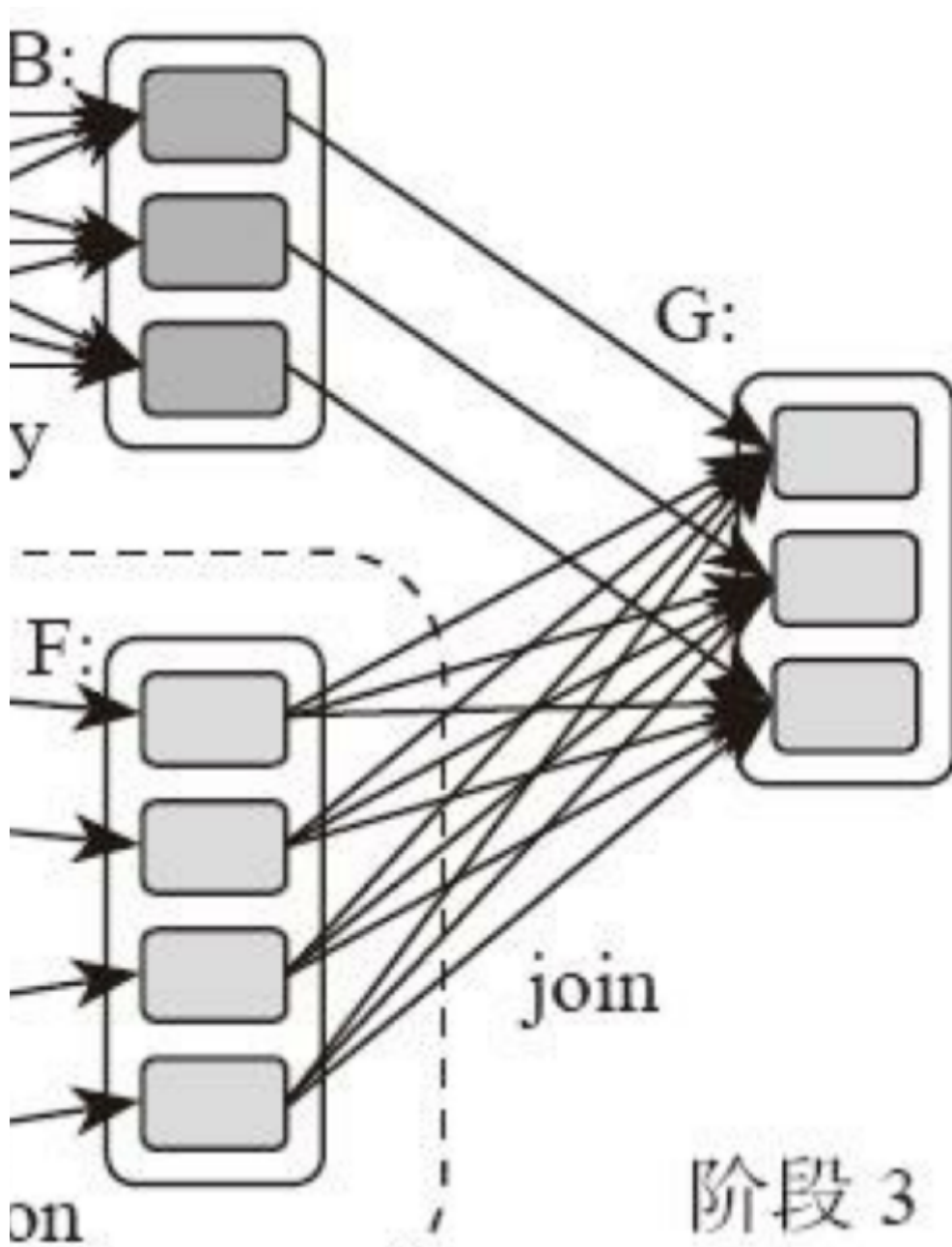
你可以再观察一下上面的 DAG 图，关于计算阶段的划分从图上就能看出规律，当 RDD 之间的转换连接线呈现多对多交叉连接的时候，就会产生新的阶段。一个 RDD 代表一个数据集，图中每个 RDD 里面都包含多个小块，每个小块代表 RDD 的一个分片。

一个数据集中的多个数据分片需要进行分区传输，写入到另一个数据集的不同分片中，这种数据分区交叉传输的操作，我们在 MapReduce 的运行过程中也看到过。

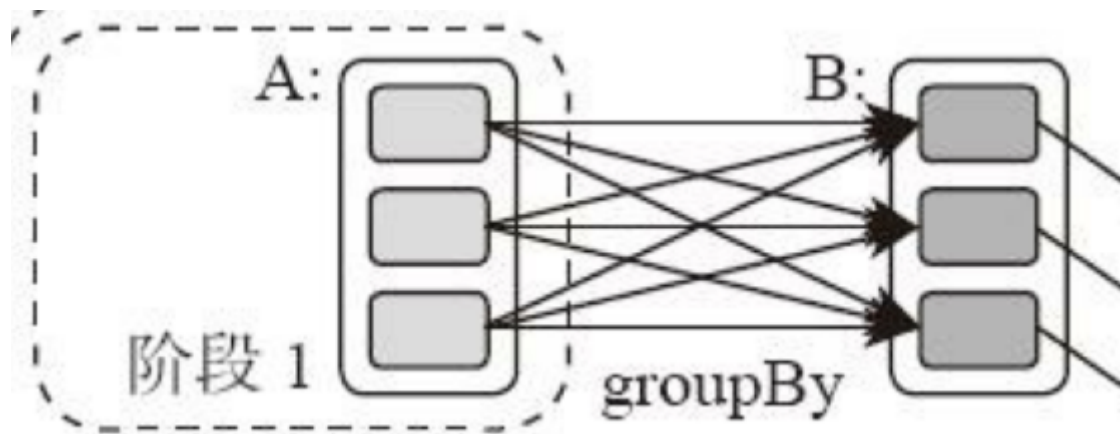


是的，这就是 shuffle 过程，Spark 也需要通过 shuffle 将数据进行重新组合，相同 Key 的数据放在一起，进行聚合、关联等操作，因而每次 shuffle 都产生新的计算阶段。这也是为什么计算阶段会有依赖关系，它需要的数据来源于前面一个或多个计算阶段产生的数据，必须等待前面的阶段执行完毕才能进行 shuffle，并得到数据。

这里需要你特别注意的是，**计算阶段划分的依据是 shuffle，不是转换函数的类型**，有的函数有时候有 shuffle，有时候没有。比如上图例子中 RDD B 和 RDD F 进行 join，得到 RDD G，这里的 RDD F 需要进行 shuffle，RDD B 就不需要。



因为 RDD B 在前面一个阶段，阶段 1 的 shuffle 过程中，已经进行了数据分区。分区数目和分区 Key 不变，就不需要再进行 shuffle。



这种不需要进行 shuffle 的依赖，在 Spark 里被称作窄依赖；相反的，需要进行 shuffle 的依赖，被称作宽依赖。跟 MapReduce 一样，shuffle 也是 Spark 最重要的一个环节，只有通过 shuffle，相关数据才能互相计算，构建起复杂的应用逻辑。

在你熟悉 Spark 里的 shuffle 机制后我们回到今天文章的标题，同样都要经过 shuffle，为什么 Spark 可以更高效呢？

其实从本质上看，Spark 可以算作是一种 MapReduce 计算模型的不同实现。Hadoop MapReduce 简单粗暴地根据 shuffle 将大数据计算分成 Map 和 Reduce 两个阶段，然后就算完事了。而 Spark 更细腻一点，将前一个的 Reduce 和后一个的 Map 连接起来，当作一个阶段持续计算，形成一个更加优雅、高效地计算模型，虽然其本质依然是 Map 和 Reduce。但是这种多个计算阶段依赖执行的方案可以有效减少对 HDFS 的访问，减少作业的调度执行次数，因此执行速度也更快。

并且和 Hadoop MapReduce 主要使用磁盘存储 shuffle 过程中的数据不同，Spark 优先使用内存进行数据存储，包括 RDD 数据。除非是内存不够用了，否则是尽可能使用内存，这也是 Spark 性能比 Hadoop 高的另一个原因。

Spark 的作业管理

我在专栏上一期提到，Spark 里面的 RDD 函数有两种，一种是转换函数，调用以后得到的还是一个 RDD，RDD 的计算逻辑主要通过转换函数完成。

另一种是 action 函数，调用以后不再返回 RDD。比如 `count()` 函数，返回 RDD 中数据的元素个数；`saveAsTextFile(path)`，将 RDD 数据存储到 path 路径下。Spark 的 DAGScheduler 在遇到 shuffle 的时候，会生成一个计算阶段，在遇到 action 函数的时候，会生成一个作业 (job)。

RDD 里面的每个数据分片，Spark 都会创建一个计算任务去处理，所以一个计算阶段会包含很多个计算任务（task）。

关于作业、计算阶段、任务的依赖和时间先后关系你可以通过下图看到。

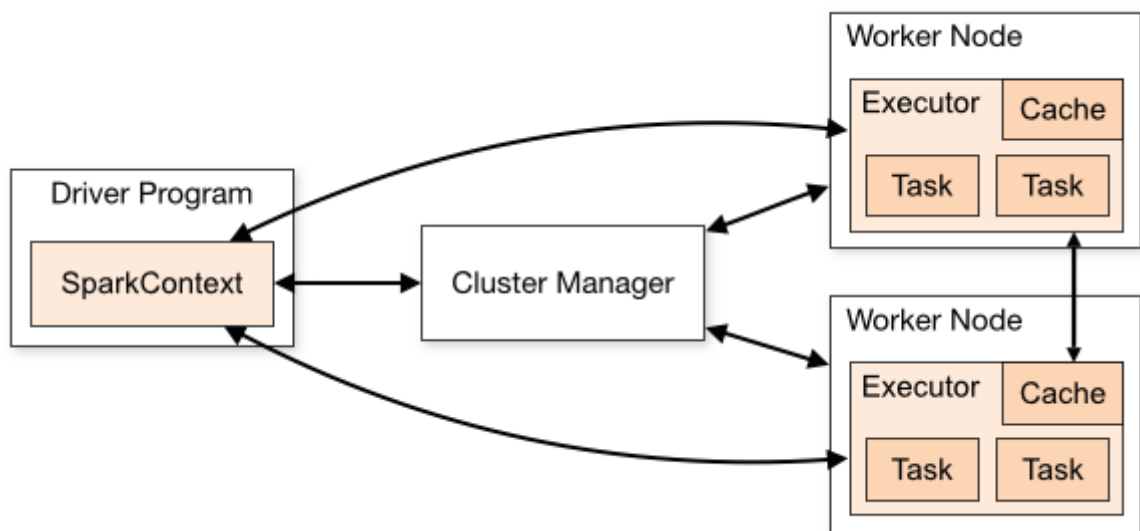


图中横轴方向是时间，纵轴方向是任务。两条粗黑线之间是一个作业，两条细线之间是一个计算阶段。一个作业至少包含一个计算阶段。水平方向红色的线是任务，每个阶段由很多个任务组成，这些任务组成一个任务集合。

DAGScheduler 根据代码生成 DAG 图以后，Spark 的任务调度就以任务为单位进行分配，将任务分配到分布式集群的不同机器上执行。

Spark 的执行过程

Spark 支持 Standalone、Yarn、Mesos、Kubernetes 等多种部署方案，几种部署方案原理也都一样，只是不同组件角色命名不同，但是核心功能和运行流程都差不多。



上面这张图是 Spark 的运行流程，我们一步一步来看。

首先，Spark 应用程序启动在自己的 JVM 进程里，即 Driver 进程，启动后调用 SparkContext 初始化执行配置和输入数据。SparkContext 启动 DAGScheduler 构造执行的 DAG 图，切分成最小的执行单位也就是计算任务。

然后 Driver 向 Cluster Manager 请求计算资源，用于 DAG 的分布式计算。Cluster Manager 收到请求以后，将 Driver 的主机地址等信息通知给集群的所有计算节点 Worker。

Worker 收到信息以后，根据 Driver 的主机地址，跟 Driver 通信并注册，然后根据自己的空闲资源向 Driver 通报自己可以领用的任务数。Driver 根据 DAG 图开始向注册的 Worker 分配任务。

Worker 收到任务后，启动 Executor 进程开始执行任务。Executor 先检查自己是否有 Driver 的执行代码，如果没有，从 Driver 下载执行代码，通过 Java 反射加载后开始执行。

小结

总结来说，Spark 有三个主要特性：**RDD 的编程模型更简单，DAG 切分的多阶段计算过程更快速，使用内存存储中间计算结果更高效**。这三个特性使得 Spark 相对 Hadoop MapReduce 可以有更快的执行速度，以及更简单的编程实现。

Spark 的出现和流行其实也有某种必然性，是天时、地利、人和的共同作用。首先，Spark 在 2012 年左右开始流行，那时内存的容量提升和成本降低已经比 MapReduce 出现的十年前强了一个数量级，Spark 优先使用内存的条件已经成熟；其次，使用大数据进行机器学习的需求越来越强烈，不再是早先年那种数据分析的简单计算需求。而机器学习的算法大多需要很多轮迭代，Spark 的 stage 划分相比 Map 和 Reduce 的简单划分，有更加友好的编程体验和更高效的执行效率。于是 Spark 成为大数据计算新的王者也就不足为奇了。

思考题

Spark 的流行离不开它成功的开源运作，开源并不是把源代码丢到 GitHub 上公开就完事大吉了，一个成功的开源项目需要吸引大量高质量开发者参与其中，还需要很多用户使用才能形成影响力。

Spark 开发团队为 Spark 开源运作进行了大量的商业和非商业活动，你了解这些活动有哪些吗？假如你所在的公司想要开源自己的软件，用于提升自己公司的技术竞争力和影响力，如果是你负责人，你应该如何运作？

欢迎你写下自己的思考或疑问，与我和其他同学一起讨论。



从 0 开始学大数据

智能时代你的大数据第一课

李智慧
同程艺龙交通首席架构师
前 Intel 大数据架构师



新版升级：点击「 请朋友读」，10位好友免费读，邀请订阅更有**现金**奖励。

© 版权归极客邦科技所有，未经许可不得传播售卖。页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 12 | 我们并没有觉得MapReduce速度慢，直到Spark出现

下一篇 14 | BigTable的开源实现：HBase

精选留言 (42)

 写留言



vivi

2018-12-25

 16

懂了原理，实战其实很简单，不用急着学部署啊，操作，原理懂了，才能用好，我觉得讲得很好

作者回复: 



My dream

2018-11-28

👍 12

老师，你讲的理论看的我头晕脑胀的，能不能讲点实战操作，搭建spark环境，通过案例来讲的话，对于我们这些初学学生来说是最直观，最容易弄明白它为什么会那么快，像你这样一味的讲理论，不讲实战，我们实在是吸收不了，理解不了你讲的这些知识点



落叶飞逝的...

2018-11-27

👍 12

总结：Spark的优点就是能够动态根据计算逻辑的复杂度进行不断的拆分子任务，而实现在一个应用中处理所有的逻辑，而不像MapReduce需要启动多个应用进行计算。

展开 ∨



scorpiozj

2018-11-28

👍 8

移动计算比移动数据划算 总结的真好 很多设计仔细想一想都是围绕这个中心

关于开源

- 1 准备好详细的使用 api文档并提供示例
- 2 撰写设计思路 和竞品比较的优势 以及创新点...

展开 ∨



arn

2018-11-28

👍 5

每一篇文章都认真的读了，有些东西还没真正的去在实际工作中体会到，但这种思维的启发还是受益匪浅。



追梦小乐

2018-11-27

👍 3

老师，我想请教几个问题：

- 1、“根据每个阶段要处理的数据量生成相应的任务集合（TaskSet），每个任务都分配一个任务进程去处理”，是一个任务集合TaskSet启动一个进程，taskSet里面的任务是用线程计算吗？还是每个TaskSet里面的任务启动一个进程？

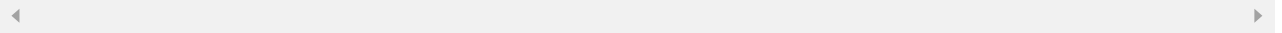
...

展开 ▾

作者回复: 每个任务一个进程

很多红色线条, 每条线代表一个任务

cache理解成存储rdd的内存



纯洁的憎恶

2018-11-27

👍 3

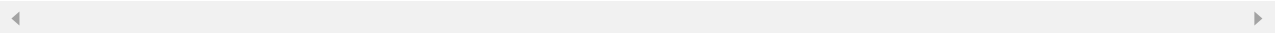
这两天的内容对我来说有些复杂, 很多知识点没有理解透。针对“而 Spark 更细腻一点, 将前一个的 Reduce 和后一个的 Map 连接起来, 当作一个阶段持续计算, 形成一个更加优雅、高效地计算模型”。这句话中“将前一个的 Reduce 和后一个的 Map 连接起来”在细节上该如何理解, 这也是明显的串行过程, 感觉不会比传统的MapReduce快? 是因为不同阶段之间有可能并行么?

展开 ▾

作者回复: 引用楼下的评论回复

落叶飞逝的恋

总结: Spark的优点就是能够动态根据计算逻辑的复杂度进行不断的拆分子任务, 而实现在一个应用中处理所有的逻辑, 而不像MapReduce需要启动多个应用进行计算。



Zach_

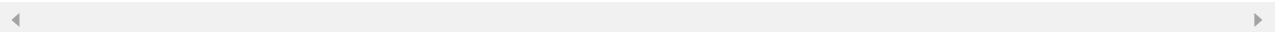
2018-11-27

👍 3

啊、老师现在的提问都好大, 我现在是老虎吃天无从下爪啊 ^_^

展开 ▾

作者回复: 有些问题不一定要得到答案或者回答出来, 只是关注到了思考一下, 就会有收获~



ming

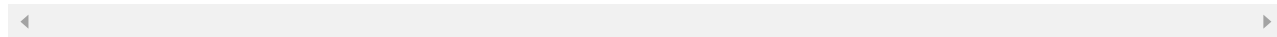
2018-12-08

👍 2

老师，有一句话我不太理解，请老师指导。“DAGScheduler 根据代码和数据分布生成 DAG 图”。根据代码生产DAG图我理解，但是为什么生成DAG图还要根据数据分布生成，数据分布不同，生成的DAG图也会不同吗？

展开 ∨

作者回复: 数据分布删掉，谢谢指正。



felix

2018-11-28

👍 2

老师有了解过微软sql server analysis service的tabular吗，内存型列式存储的数据库，数据压缩比很高。如果光做数据库查询的话，比spark还快很多，因为spark还要把数据load进内存。我们自己写了一个服务来分布聚合，还用redis以维度为key做缓存。



Jack Zhu

2018-11-28

👍 2

- 1.确定做开源的商业模式，是羊毛出在猪身上，还是送小羊推大羊等等
- 2.确定组织投入人员
- 3.建立社区，定期together讨论，为迭代版本做计划
- 4.做一些必要商业推广和免费客户服务



海

2019-03-07

👍 1

对于hbase和高速发展的es，不知道您怎么看，他们的优缺点是什么？

展开 ∨



落叶飞逝的...

2019-03-04

👍 1

现在回过头看，Spark的编程模型其实类似Java8的Stream编程模型

展开 ∨



张飞

2019-03-01

👍 1

1. “而 Spark 更细腻一点，将前一个的 Reduce 和后一个的 Map 连接起来，当作一个阶

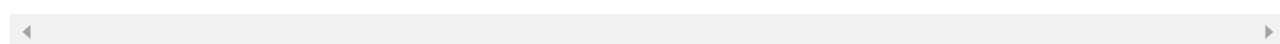
段持续计算，形成一个更加优雅、高效地计算模型”，stage之间是串行的，即便前一个的reduce和后一个的map连接起来，也是要从前一个stage的计算节点的磁盘上拉取数据的，这跟mapreduce的计算是一样的，老师所说的高效在这里是怎么提现的呢？

2. spark的内存计算主要体现在shuffle过程，下一个stage拉取上一个stage的数据的时...
展开 ∨

作者回复: 1 Spark的map和reduce的划分要更优雅一点，比如宽依赖和窄依赖，编程上看不出明显的map和reduce，这种优雅还有很多，多写一些spark和MapReduce程序就能感受到。

2 如果内存够用，Spark几乎总是使用内存。

3 可以这么理解。



数据化分析

2018-12-14

1

希望能讲讲实际怎么操作。

展开 ∨



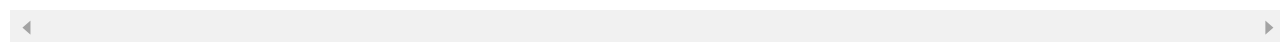
白鸽

2018-11-30

1

Executor 从 Diver 下载执行代码，是整个程序 jar包?还是仅 Executor 计算任务对应的一段计算程序（经SparkSession初始化后的）？

作者回复: 整个jar



Riordon

2018-11-27

1

为何Spark可以更高效？

1) MR使用多进程模型，Spark使用多线程模型，这样Spark构建一个可重用的资源池，而MR往往不会，MR会耗费更多的构建资源池的启动时间，Spark在进程内线程会竞争资源，导致Spark相比MR不稳定；

2) 对于多次读取的中间结果可以Cache，避免多次读盘，对于迭代计算友好； ...

展开 ∨





周小桥

2019-05-23



聚合操作是它最大的缺点。

展开 ▾



周小桥

2019-05-23



拥有大厂的支持，是最好的推广。

展开 ▾



冰镇可...

2019-04-30



之前讲mr时候也有提到生成dag，spark这里也是dag，二者的差异是mr中，map reduce为一组操作(可能没有reduce)的一个job job之间是依赖关系，而spark并非简单依照m r划分而是针对数据的处理情况，如果r后到下一个m是窄依赖，则属于同一个stage，属于一个流程，这样理解对吗？

作者回复: 可以这样理解。

MR模型本身不包含DAG，需要外部工具基于MR构建DAG实现复杂的计算，比如Hive。

Spark的计算模型本身就是包含DAG。

