Spark内存管理及缓存策略研究

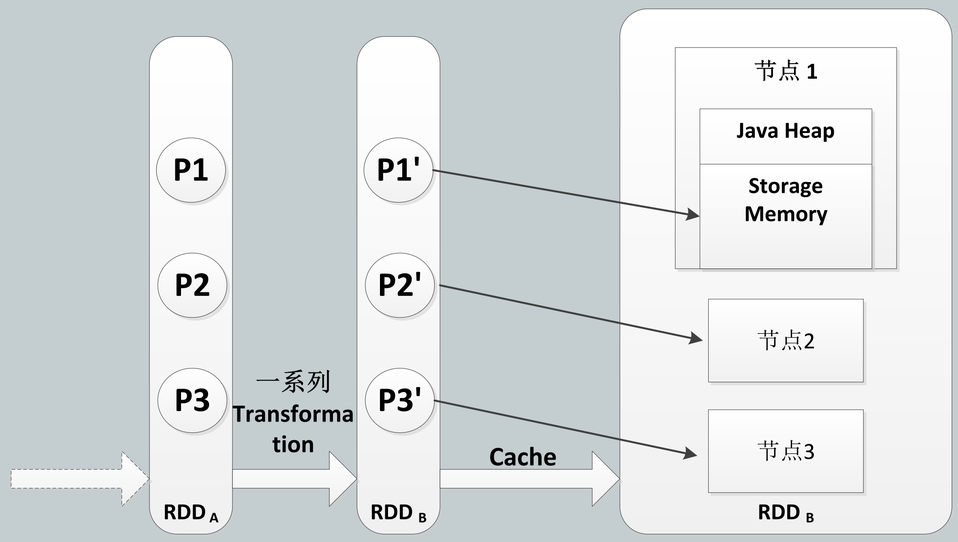
孟红涛 余松平 刘芳 肖侬
国防科技大学计算机学院，长沙 410073
(mht0228@163.com)

摘要

Spark系统是基于Map-Reduce模型的大数据处理框架，Spark能够充分利用集群的内存，从而加速数据的处理速度。Spark按照功能把内存分成不同的区域：Shuffle Memory、Storage Memory、Unroll Memory，不同的区域有不同的使用特点。本文测试并分析了Shuffle Memory和Storage Memory的使用特点。RDD是Spark系统最重要的抽象，RDD能够缓存在集群的内存中，在内存不足时，需要淘汰部分RDD分区。本文提出一种新的RDD分布式权值缓存策略，通过RDD分区的存储时间、大小、使用次数等来分析RDD分区的权值，并根据RDD的分布式特征对需要淘汰的RDD分区进行选择，最后测试和分析了多种缓存策略的性能。

Spark内存管理及缓存策略

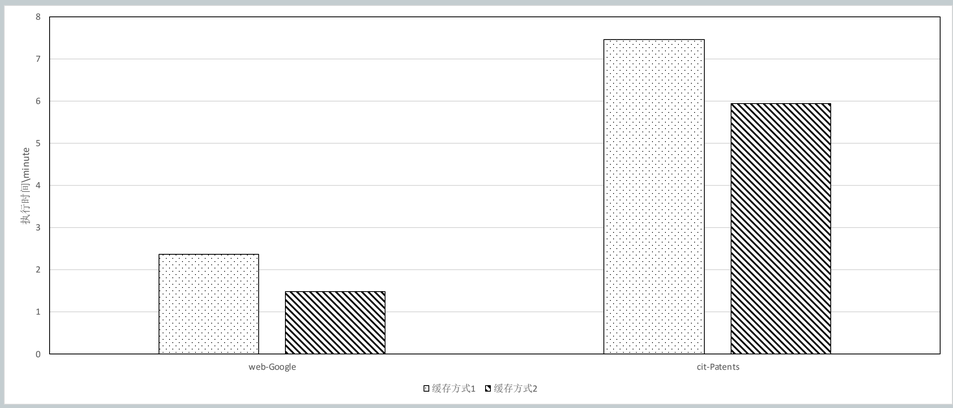
Spark应用在执行过程中，需要用到某个RDD时，首先在集群的内存中查找，看该RDD是否已经被缓存，如果已经缓存，则直接读取，如果未被缓存，则需要重新计算。重新计算时，按照其依赖关系在集群的内存中查找其依赖的父RDD，如果集群内存中已缓存，直接使用父RDD计算得到该RDD，如果集群内存中没有，则继续逆推查找和计算。



Spark Memory使用特点

在使用Google-Web标准数据集来测试PageRank，使用默认配置，不会发生Spill。所以，**计算密集型应用需要配置较大的Shuffle Memory空间。**

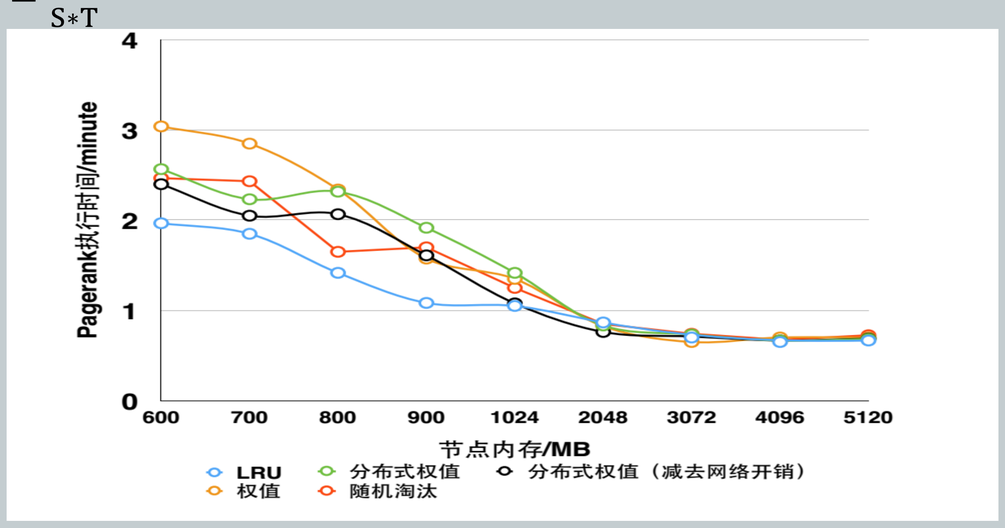
Spark集群上运行占用较大内存资源的应用时，如果缓存太多的RDD反而有可能会影响程序的性能。



一种分布式权值缓存策略

如果该RDD的所有分区都缓存在Storage Memory中，该阶段的所有Task处理速度都很快。但如果该RDD的某一个分区被淘汰了出去，处理这个分区的Task首先要按照RDD的依赖关系重新计算该RDD分区 ，然后再处理该RDD分区 。RDD分区的三个特征：（1）在其他条件相同的条件下应该优先淘汰占据内存空间大的RDD分区。因为淘汰出大的RDD分区后，会释放更多的内存资源。（2）RDD分区被使用的次数越多，说明该RDD在Spark应用中越重要。在内存不足时，应该尽量保留被访问次数多的RDD分区，淘汰被访问次数较少的RDD分区。（3）某个RDD分区存入内存的时间越久，说明该RDD分区是相对“旧”一些的RDD分区，在其他条件相同的情况下，优先淘汰这些“旧”的数据。所以，RDD分区的权值 的计算如下

$$P = \frac{N}{S*T}$$



结论

本文测试并分析了Spark系统的两个内存使用特点:(1)计算密集型应用更倾向于使用Spark节点的Shuffle Memory。(2)Spark系统中，适度缓存RDD能够加速Spark应用的执行速度，但缓存太多RDD反而可能影响Spark应用的整体性能。基于RDD的多个特征，本文提出了一种新的分布式权值缓存策略，在淘汰RDD分区时能够综合考虑RDD的多种特点，最后测试采用多种缓存策略时PageRank的执行时间，分析Spark系统在不同缓存策略下的性能。