Shuffle过程原理

本小节介绍Shuffle过程原理。

Shuffle称为MapReduce的“心脏”和所谓“奇迹发生的地方”,描述着数据从map task输出到reduce task输入的这段过程。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。因为在分布式情况下，reduce task需要跨节点去拉取其它节点上的map task结果。这一过程将会产生网络资源消耗和内存、磁盘I/O的消耗。

通常shuffle分为两部分：Map阶段的数据准备和Reduce阶段的数据拷贝处理。

一般将在map端的Shuffle称之为Shuffle Write，在Reduce端的Shuffle称之为Shuffle Read。

map端的Shuffle过程简述:

1)input, 根据split输入数据，运行map任务;

2)patition, 每个map task都有一个内存缓冲区，存储着map的输出结果;

3)spill, 当缓冲区快满的时候需要将缓冲区的数据以临时文件的方式存放到磁盘;

4)merge, 当整个map task结束后再对磁盘中这个map task产生的所有临时文件做合并，生成最终的正式输出文件，然后等待reduce task来拉数据。

reduce task在执行之前的工作就是不断地拉取当前job里每个map task的最终结果，然后对从不同地方拉取过来的数据不断地做merge，也最终形成一个文件作为reduce task的输入文件。

具体过程是：

Reduce任务通过RPC向JobTracker询问Map任务是否已经完成，若完成，则领取数据

Reduce领取数据先放入缓存，来自不同Map机器，先归并，再合并，写入磁盘

多个溢写文件归并成一个或多个大文件，文件中的键值对是排序的

当数据很少时，不需要溢写到磁盘，直接在缓存中归并，然后输出给Reduce

概括一下，MapReduce应用程序执行过程：

1.待处理的大数据，被划分成大小相同的数据集(如64MB)，以及与此相应的用户作业程序。

2.系统中有一个负责调度的主节点(Master)，以及数据Map和Reduce工作节点(Worker).

3.用户作业提交到主节点。

4.主节点为作业程序寻找和配备可用的Map节点，并将程序传送给Map节点。

5.主节点也为作业程序寻找和配备可用的Reduce节点，并将程序传送给Reduce节点。

6.主节点启动每一个Map节点执行程序，每个Map节点尽可能读取本地或本机架的数据进行计算。(实现代码向数据靠拢，减少集群中数据的通信量)。

7.每个Map节点处理读取的数据块，并做一些数据整理工作(combining,sorting等)并将数据存储在本地机器上；同时通知主节点计算任务完成并告知主节点中间结果数据的存储位置。

8.主节点等所有Map节点计算完成后，开始启动Reduce节点运行；Reduce节点从主节点所掌握的中间结果数据位置信息，远程读取这些数据。

9.Reduce节点计算结果汇总输出到一个结果文件，即获得整个处理结果。

MapReduce适宜数据密集型，是大数据时代的海量数据处理利器，可以很好地应用于各种计算问题，诸如关系代数运算、聚类运算、矩阵乘法等。

在Google，MapReduce用在非常广泛的应用程序中，包括“分布grep，分布排序，web连接图反转，每台机器的词矢量，web访问日志分析，反向索引构建，文档聚类，机器学习，基于统计的机器翻译...”

Nutch项目开发了一个实验性的MapReduce的实现，也即是后来大名鼎鼎的hadoop。

Phoenix是斯坦福大学开发的基于多核/多处理器、共享内存的MapReduce实现。

本章总结

本章介绍了MapReduce编程模型的相关知识。

MapReduce将复杂的、运行于大规模集群上的并行计算过程高度地抽象到了两个函数：Map和Reduce，并极大地方便了分布式编程工作，编程人员在不会分布式并行编程的情况下，也可以很容易将自己的程序运行在分布式系统上，完成海量数据集的计算。

MapReduce执行的全过程包括以下几个主要阶段：从分布式文件系统读入数据、执行Map任务输出中间结果、通过 Shuffle阶段把中间结果分区排序整理后发送给Reduce任务、执行Reduce任务得到最终结果并写入分布式文件系统。在这几个阶段中，Shuffle阶段非常关键，必须深刻理解这个阶段的详细执行过程。

MapReduce具有广泛的应用，比如关系代数运算、分组与聚合运算、矩阵-向量乘法、矩阵乘法等。