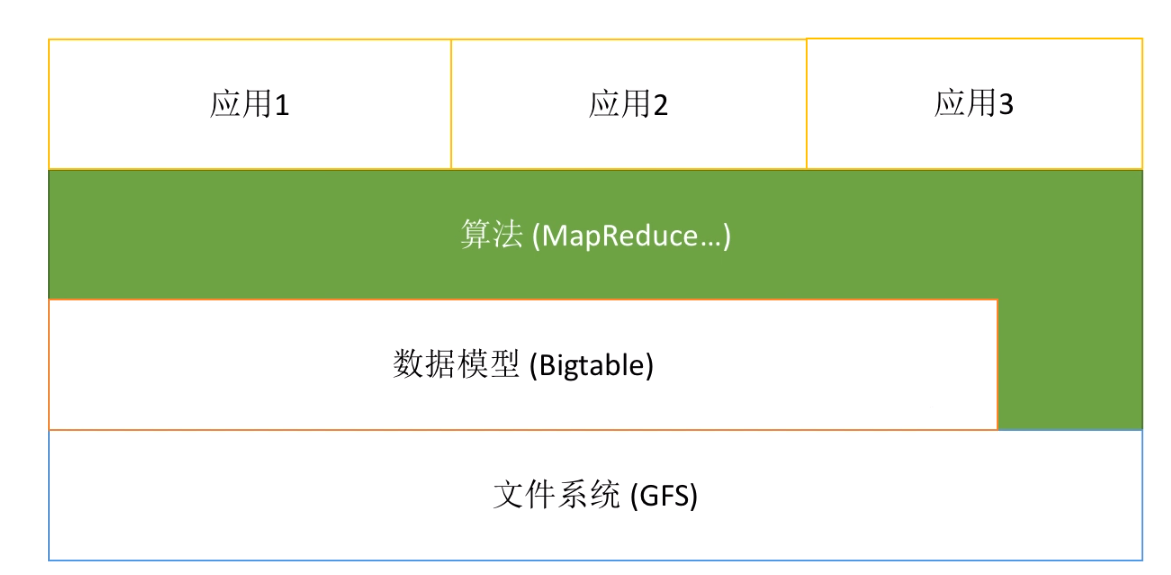
# MapReduce的产生

MapReduce的概念来自于“谷歌三大论文”中的最后一篇《MapReduce: Simplified Data Processing on Large Clusters》[2]， 并作为算法模型建立在另外两篇论文中分别提出的文件系统Google File System(GFS)和数据模型Bigtable之上应用于大规模集群环境下的分布式计算问题，三者之间的关系如下图所示。

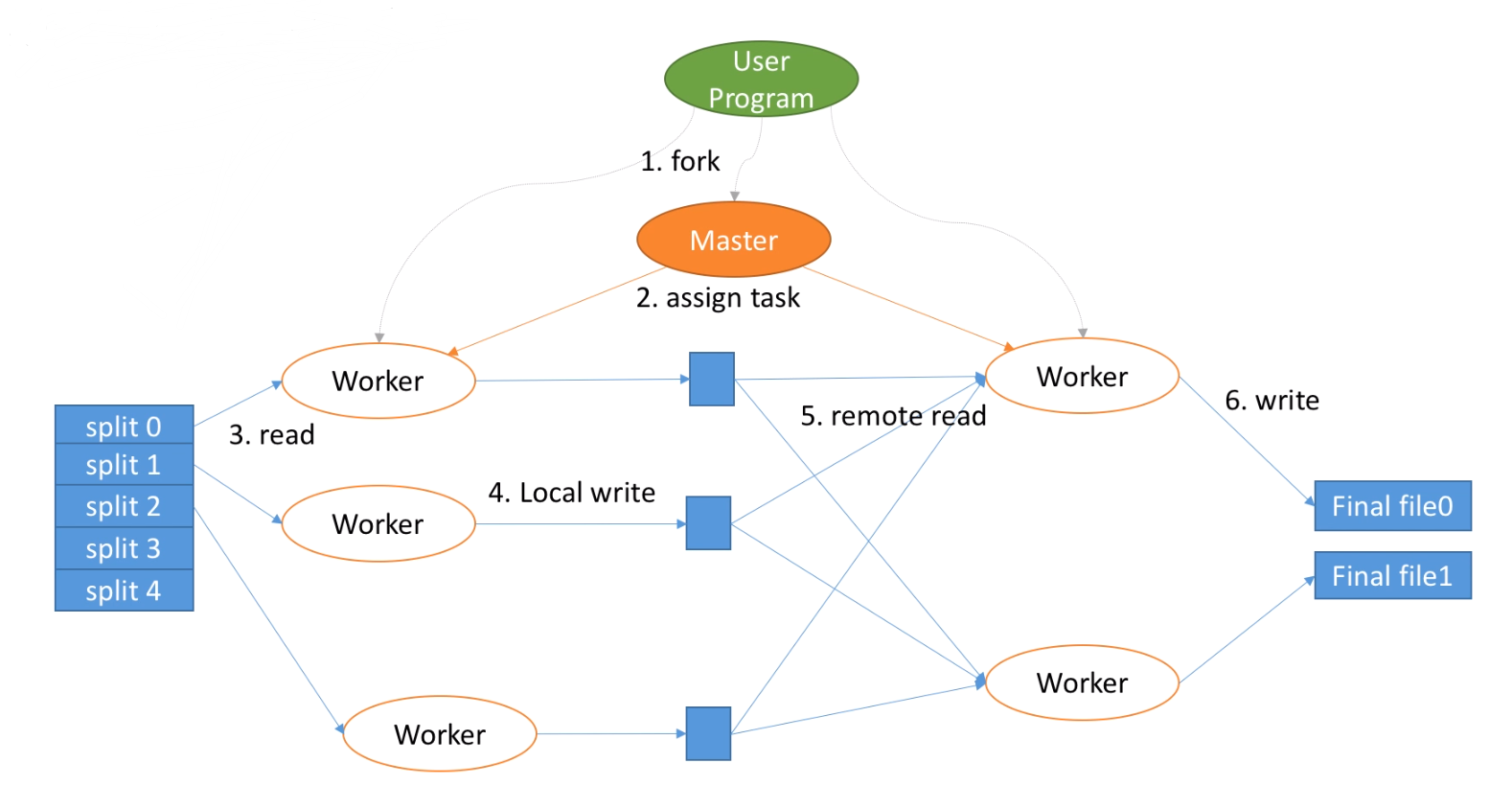


# MapReduce的架构

MapReduce的应用场景包含一个定义程序功能，协调程序运行的用户进程User Program。User Program根据问题的规模和需求指派一个单独的Master Job Tracker和集群节点上的Slave Task Tracker。Job Tracker负责调度构成一个作业的所有任务，而这些任务分布在不同的Task Tracker上，Job Tracker监视它们的执行情况并重新执行已经失败的任务，而Task Tracker仅负责执行由Job Tracker指派的任务。

MapReduce在执行时先指定一个Map（影射）函数，把输入的键值对映射成一组新的键值对，经过一定的处理后交给Reduce，Reduce对相同的Key下的所有Value进行处理后再输出键值对作为最终的结果。

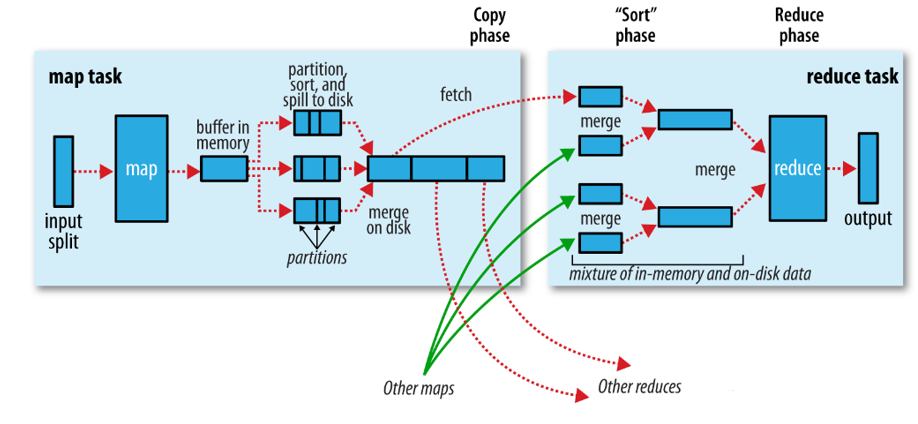
运行MapReduce架构算法时，Job Tracker会指派若干个集群节点为Map Worker或Reduce Worker。每个Map Worker从本地读取HDFS文件系统中数据文件分块的结果Split存入本地的内存或磁盘完成Map操作，而Reduce Worker负责从Map Worker节点上取数据在本地进行归并以完成Reduce操作。上述操作完成后即可交付数据，完成一项完整的MapReduce任务[2]。MapReduce的架构图[3]如下图所示。



# MapReduce的过程

MapReduce过程的核心操作时Map与Reduce。集群环境下的并行计算中，大部分Map Task与Reduce Task的执行是在不同节点上的。Reduce执行时需要跨节点地去拉取其它节点上的Map Task结果。如果集群正在运行的作业有很多，那么Map Task的正常执行对集群内部网络资源的消耗会很严重。同时在节点内，相比于内存，磁盘IO对job完成时间的影响也是可观的。此时为了减少拉取数据量和增加内存的使用比例（相对于硬盘），还需要一个逻辑上介于Map过程和Reduce过程之间的Shuffle过程。

Map，Reduce和Shuffle三个过程的官方示意图如下图所示。



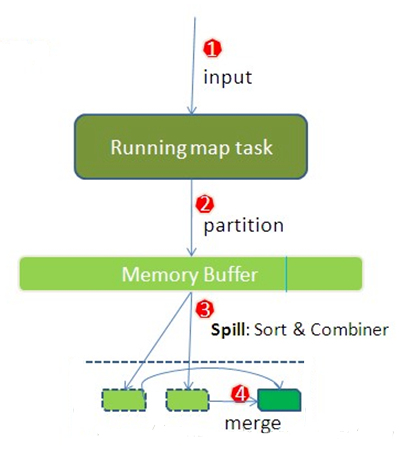
## Map过程

在Map Task 执行时，对其输入来源HDFS的Block，Map Task只读取Split。Split与 Block的对应关系默认为一对一，这一过程称之为Input过程。 Map Worker的输出是 Key/Value对，然后将数据写入内存缓冲区中，缓冲区的作用是批量收集Map结果，减少磁盘IO的影响。写入缓冲区之前，Key与Value值都会被序列化成字节数组，这一过程称之为Partition过程。

## Shuffle过程

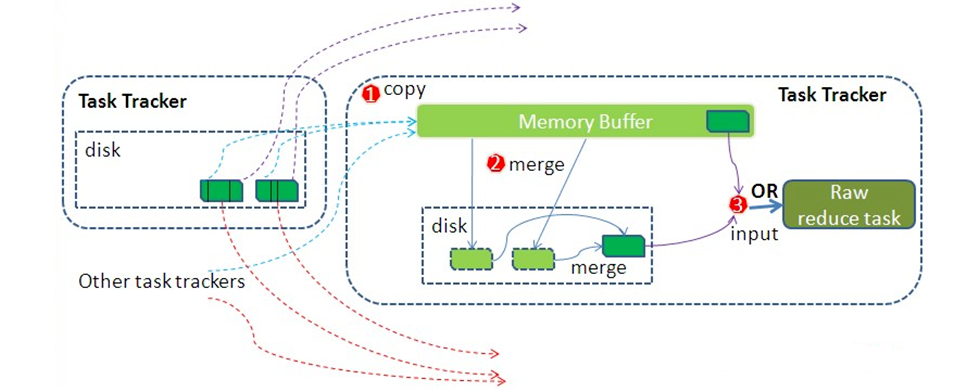
由于内存缓冲区的大小限制，当Map Task输出结果很多时就可能发生内存溢出，所以需要在一定条件下将缓冲区的数据临时写入磁盘。这个从内存往磁盘写数据的过程被称为Spill。Spill是由单独线程来完成，不影响往缓冲区写Map结果的线程。在Reduce Worker从Map Worker取数据之前，为了减少Reduce Worker Spill到磁盘的数据量。Map Worker会对具有相同Key的 Key/Value 对进行Combine操作，即将其相同Key的 Key/Value相加。当Map很大时，每次Spill会产生一个Spill File，这样会有多个Spill File，而最终的输出只有一个文件，在最终输出之前会对多个中间过程多次产生的Spill文件Spill file进行合并，此过程被称为Merge。

上述过程都是在Map Worker中完成的，故称之为Map端的Shuffle操作。流程示意图如下。



在Reduce task之前，不断拉取当前 Map Worker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，最终形成一个文件作为Reduce Task的输入文件。Reduce进程启动一些数据Copy线程，通过HTTP方式请求Map Task所在的Task Tracker获取Map Task的输出文件。如果Map输出足够小，它们会被拷贝到Reduce Task Tracker的内存中；如果缓冲区空间不足，则会被拷贝到磁盘上。当内存中的缓冲区用量达到一定比例阀值，或者达到了Map 输出的阀值大小，缓冲区中的数据将会被归并然后Spill 到磁盘。拷贝来的数据叠加在磁盘上，有一个后台线程会将它们归并为更大的排序文件，这样做节省了后期归并的时间。

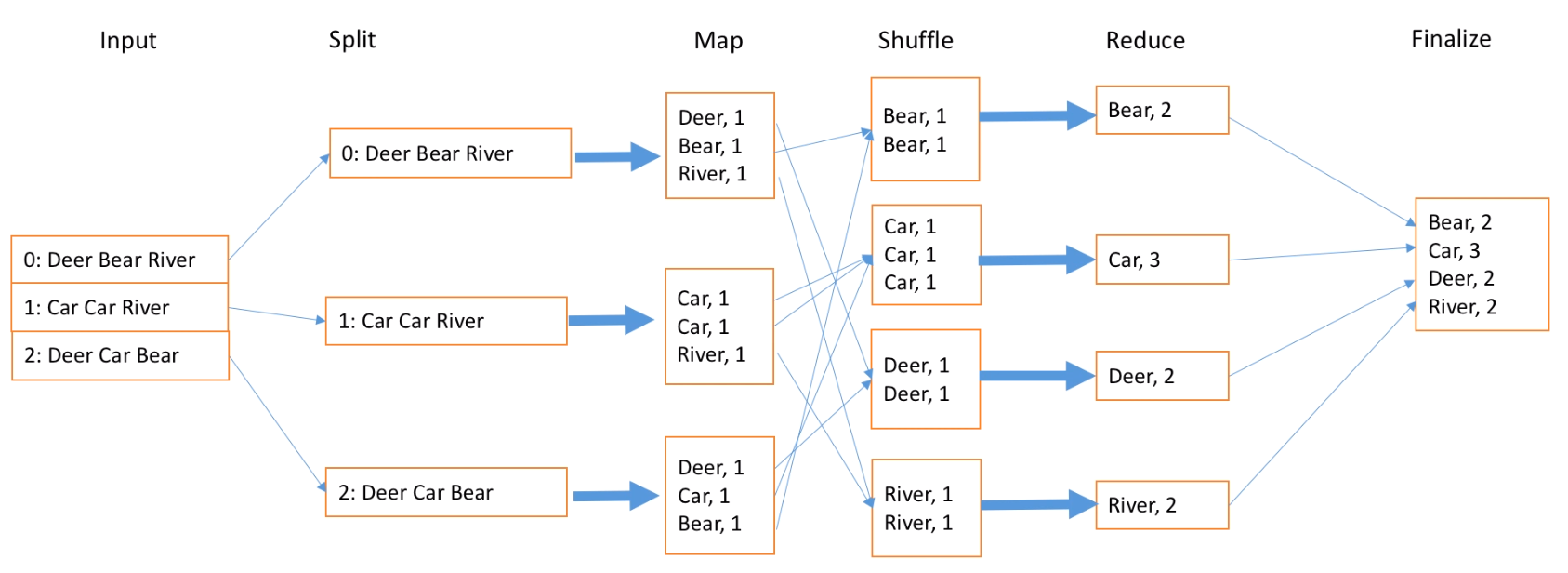
上述过程都是在Reducer Worker中完成的，故称之为Reduce端的Shuffle操作。流程示意图如下。



## Reduce过程

Reduce过程就是在Shuffle过程完成Copy过程和Merge过程后，从Reduce Worker的内存或磁盘中读取价值对做归并操作，Reduce的具体功能由User Program确定，用以完成最终的集群环境下的并行计算任务。

# MapReduce的实例——Word Count



1. **Input：**输入为一个有多行的文本文件，内容为由英文单词组成的句子；
2. **Split：**将文本数据的每一行切分成一个Block，作为Map过程的输入；
3. **Map：**没个Map Worker统计取得的Block中每个单词出现的频数，生成键值对；
4. **Shuffle：**在数据被Reduce Worker取得之前，对Map过程得输出数据进行简单得归并以减少单一节点上内存和磁盘交换的次数，节约网络传输成本；
5. **Reduce：**把所有值相同的键值对相加，实现的数据的归并，得到每个单词在全文档中出现的频数；
6. **Finalize：**把Reduce过程的输出取得作为最终MapReduce任务的输出结果，完成Word Count任务。
7. Wadkar S, Siddalingaiah, Venner. 深入理解Hadoop[M]. 原书第2版. 北京:机械工业出版社, 2015.10.
8. White T. Hadoop 权威指南[M]. 原书第2版. 北京:清华大学出版社, 2011.7.
9. Ghemawat J A S. MapReduce: Simpliﬁed Data Processing on Large Clusters[D]. Google, Inc., 2004.