MapReduce读后感

通过摘要，我大致了解到MapReduce 是一个编程模型，也是一个处理和生成超大数据集的算法模型的相关实现。用户可以通过创建一个 Map函数处理一个基于 key/value pair 的数据集合，输出中间的基于 key/value pair 的数据集合；然后再创建一个 Reduce 函数用来合并所有的具有相同中间 key 值的中间 value 值。同时，作者在摘要中就指出了MapReduce的两大特点：程序能够在大量的普通配置的计算机上实现并行化处理，同时还可以实现运行在规模可以灵活调整的由普通机器组成的集群上。

在第一节介绍了，作者介绍了开发此模型的应用需求背景，也就是数据处理运算在概念上很容易理解。然而由于输入的数据量巨大，因此要想在可接受的时间内完成运算，只有将这些计算分布在成百上千的主机上。而为了解决这一问题，Google团队提出并设计了这一新的抽象模型。而通过这一模型的运用，使用者只要表述我们想要执行的简单 运算即可，而不必关心并行计算、容错、数据分布、负载均衡等复杂的细节，这些问题都被封装在了一个库里面。

在第二节中，作者简要介绍了这一编程模型的原来，也就是利用一个输入 key/value pair 集合来产生一个输出的 key/value pair 集 合 。MapReduce 库的用户用两个函数表达这个计算：Map 和 Reduce。

在第三节中，作者详细介绍了Google团队开发词模型的实现。MapReduce 模型可以有多种不同的实现方式。如何正确选择取决于具体的环境。首先是Master的数据结构，Master 持有一些数据结构，它存储每一个 Map 和 Reduce 任务的状态以及 Worker 机器的标识。因为 MapReduce 库的设计初衷是使用由成百上千的机器组成的集群来处理超大规模的数据，所以，这个库必须要 能很好的处理机器故障。同时也有失效方面的处理机制。

在第四节中，作者指出，虽然简单的 Map 和 Reduce 函数提供的基本功能已经能够满足大部分的计算需要，但还是发掘出了一些有价值的扩展功能，包含了分区函数、顺序保证、Combiner函数、以及输入输出模型，状态信息、计数器等待。想要完全理解谷歌的这篇论文还需要长期的积累，现在最重要的是通过这篇论文学习大数据的思想，放宽视野。

在第五节中，作者通过集群配置、GREP、排序、高效backup任务、失效机器等方面描述了MapReduce数据模型的效率。并在第六节中，通过实际案例展现出了MapReduce的成功，被用作重写了 Google 网络搜索服务所使用到的 index 系统。索引系统的输入 数据是网络爬虫抓取回来的海量的文档，这些文档数据都保存在 GFS 文件系统里。这些文档原始内容的大小超过 了 20TB。索引程序是通过一系列的 MapReduce 操作来建立索引。实现索引部分的代码简单、小巧、容易理解，因为对于容错、分布式以及并行计算的处理都是 MapReduce 库提供的。比如，使用 MapReduce 库，计算的代码行数从原来的 3800 行 C++代码减少到大概 700 行代码，这无疑是大大地简化了程序员的开发任务。缩短了大规模任务的开发周期。

通过这篇论文的阅读，认识到了成功的数据模型的重要性，以及在公司中所起到的重要地位，可以大幅度提升开发效率，缩短开发成本。