一、Google File System

Google File System 是一个可扩展的分布式文件系统，用于大型的、分布式的、对大量数据进行访问的应用。它运行于廉价的普通硬件上，提供容错功能。从根本上说：文件被分割成很多块，使用冗余的方式储存于商用机器集群上。

GFS 由一个 Master 和大量的 Chunk Server 构成。Google 设置一个 Master 来保存目录和索引信息，这是为了简化系统结果，提高性能来考虑的，但是这就会造成主成为单点故障或者瓶颈。为了消除主的单点故障 Google 把每个 Chunk 设置的很大(64M)， 这样做的优点是只需要少量和 Master 节点的通信就可以获取 Chunk 的位置信息，之后就可以多次进行读写操作。减少了 Master 节点需要保存的元数据的数量。

文件的读取和写入，对于读取和写入效率，当读取和写入的客户机增加时，多个客户机同时读取和写入的几率也增加,导致整体的读取和写入效率降低。GFS 系的读取有两种操作，一种是大规模的流式读取，大规模的流式读取通常一次读取数百 KB 的数据，甚至更多。而另外一种是小规模的随机读取，并且在 GFS 中对于记录追加通过 Chunk 和 Chunk 的副本进行操作，保证至少有一次原子的写入操作成功执行。

Master 服务器在灾难恢复时，通过重演操作日志把文件系统恢复到最近的状态。Master 服务器在日志增长到一定量时对系统状态做一次快照。快照文件以压缩 B-树数据结构存储，可以直接映射到内存。

二、Map Reduce

论文描述了大数据的分布式计算方式，主要思想是将任务分解然后在多台处理能力较弱的计算节点中同时处理，然后将结果合并从而完成大数据处理。简单说 Map Reduce 是针对分布式并行计算的一套编程模型，Hadoop Map Reduce 基于“分而治之”的思想，将计算任务抽象成 Map(映射) 和 Reduce(归约) 两个计算过程，可以简单理解为“分散运算-归并结果”的过程。而 Shuffle 过程是 Map Reduce 的核心，Shuffle 的正常意思是洗牌或弄乱，它分布在 Map Reduce 的 Map 阶段和 Reduce 阶段，它会随机地打乱参数 List 里的元素顺序，它一般把从 Map 产生输出开始到 Reduce 取得数据作为输入之前的过程称作 Shuffle。

Map Reduce 是对 GFS 是对分割储存的数据进行利用，Map Reduce 由 Map 和 Reduce 组成，Map 的功能类似与 Master 是将 GFS 的进行映射，即是将数据进行连接并使之还原，在这时 GFS 储存数据的优势就体现出来了，即 CPU 可以对多个数据的同时处理，这样只要对 CPU 进行硬件上的优化和数据处理系统的优化，就可以从多方面对数据处理优化达到更好的效果。

Map Reduce 是利用 Map 函数处理一个输入 key/value pair 集合来产生一个具有相同 key 值的 value 值输出的中间 key/value pair 集合，使用 Reduce 函数合并所有相同 key 值的 value 值的编程模型， 也是一个处理和生成超大数据集的算法实现模型。Map Reduce 这个模型能够处理因为数据量大而将计算分布在成百上千的主机上产生的并行计算、分发数据、错误处理等问题,它将并行计算、容错、数据分布、负载均衡等复杂的细节封装在一起，也能让那些并没有并行计

算和分布式处理系统开发经验的程序员有效利用分布式系统的丰富资源。
Map Reduce 架构的程序能够在大量的普通配置的计算机上实现并行化处理。这个系统在运行时只关心:如何分割输入数据,在大量计算机组成的集群上的调度,集群中计算机的错误处理,管理集群中计算机之间必要的通信。采用 Map Reduce 架构可以使那些没有并行计算和分布式处理系统开发经验的程序员有效利用分布式系统的丰富资源。

三、Big Table

Big Table 是 Google 设计的一个分布式的结构化数据存储系统,被用来处理分布在数千台普通服务器上的 PB 级的海量数据,能够满足不同的应用需求,无论是对于数据量上,和响应速度上,Big Table 均能提供一个灵活的、高性能的解决方案。是对 GFS 和 Map Reduce 进一步分解与细化,即以一个 GFS 为单位,以 GFS 的数分解方法进行分解将数据进一步单元化,让数据的处理更加简单,易于计算机处理。

Big Table 实现了适用性广泛、可扩展、高性能和高可用性的 4 个目标,Big Table 使用了很多和并行数据库和内存数据库的实现策略,但是它不支持完整的关系数据模型,为客户提供的数据模型更加简单,客户可以动态控制数据的分布和格式、推测底层存储数据的位置相关性。

Big Table 的数据模型将存储的数据都视为字符串,但并不解析,客户程序将各种结构化和半结构化的数据串行化到这些字符串里。Big Table 是一个稀疏的、分布式的、持久化存储的多维度排序的 Map,Map 的索引是行关键字、列关键字以及时间戳,每个 value 是一个未经解析的 byte 数组,通过行关键字的字典顺序组织数据,每个行可以动态分区,每个分区叫做一个“Tablet”,用户对同一个行关键字的读或者写操作是原子的,这使得我们用户可以更加清楚的理解程序对同一个行进行并发更新操作时的行为;通过列关键字组成的集合组成的“列族”,访问控制、磁盘和内存使用统计;时间戳 主要是控制不同版本的数据,防止数据版本冲突,和列族一同使用。另外 Big Table 提供了建立和删除表以及列族的、修改集群、表和列族的元数据的 API,这些 API 帮助我们更好的管理和处理数据。这样的数据模型,让系统变得更加具有灵活性。

Big Table 的 3 个主要组件:链接到客户程序中的库、一个 Master 服务器和多个 Tablet 服务器。Master 用于管理 Tablets,处理模式的相关修改操作,Tablet 服务器都管理一个 Tablet 的集合,每个 Tablet 服务器负责处理它所加载的 Tablet 的读写操作,以及在 Tablets 过大时,对其进行分割。Tablet 服务器使用二级缓存策略提高读操作的性能。

总结

读完 Google 三篇论文之后,虽然在实现细节上还不能充分理解,理解起来甚至有点儿困难,但是大概的了解到 GFS 文件系统、Big Table 存储系统、Map Reduce 计算模型之间的关系和实现细节,三者之间互相关联,对于解决大数据时代各个应用领域的数据的计算和数据存储问题提供了有效的解决方案,能够

在各个领域都能够得到广泛的应用。

据谷歌三大论文最后一篇的发表至今，时间已经过去十三年，在这短短的时间里，计算机技术和网络已经融入到人们生活的方方面面，与各种生活方式紧密联系、息息相关，尤其是近几年出现了以计算机技术和大数据及云计算为基础的各种 APP 和平台数量激增。而它们都离不开谷歌的三大论文的支持。二十一世纪是一个信息化的时代，我们作为时代的接班人，理应努力并自觉地去了解和学习相关的知识，丰富自己的相关技能，掌握必备的相关思维。