谷歌三驾马车论文读后感

19221213 陈晨

此次非关系型数据库课程中，我按照老师的要求仔细阅读了Google公司的三篇重要的论文分别为《Google File System》、《Google Bigtable》和《Google MapReduce》，让我对分布式系统和大体量数据的存储有了基本的认识和理解。

1 .《Google File System》

这篇文章主要讲的是Google的GFS文件系统，一个面向大规模数据密集型应用的、可伸缩的分布式文件系统。它与传统的分布式系统有着很多相同的设计目标，比如，性能、可伸缩性以及可用性，并且它还考虑了负载均衡问题和技术环境的影响。设计思路：组件失效是常态事件，由于其系统是由几百甚至几千台普通廉价设备组装而成的存储机器，因此在使用过程中难免会出现各种各样的问题。文件巨大，以TB为数量级的数据的存储。大多数文件采用文件尾部追加数据而不是覆盖原有数据。应用程序和文件系统API的协同设计提高了整个系统的灵活性。由于放松了GFS的一致性的要求，简化了GFS的设计，提高了系统的灵活性。
中间部分主要讲的是GFS文件系统的设计阶段，涉及到它要达到的预期效果 、接口的设计、架构设计、节点设计、元数据的结构与存储等主要的设计过程。过程中还包含了操作日志的记录与查看以及GFS文件系统为了达到它支持的一致性而设计的一些机制规则，比如有一致性保障机制、程序实现。在系统交互方面，它的原则就是最小化所有操作和Master节点的交互。那他使用的规则就是租赁和变更顺序、数据流、原子记录追加和快照，旨在数据更好地交互与调用。再一个讲的比较多的就是Master节点的操作，它执行所有的名字空间操作，管理整个系统里所有Chunk副本，Chunk副本有三个用途，分别是Chunk创建、重新复制和重新负载均衡。在垃圾回收方面，GFS文件删除不会立刻回收可用的物理空间，它是采用惰性回收的策略。对过期失效的Chunk副本GFS也能很好地检测，因为Master节点保存了每个Chunk的版本号，根据版本号可以区分当前副本和过期版本。
最后来看看GFS的容错机制和它的性能方面展示以及它的经验教训。GFS所面临的挑战是如何处理频繁发生的组件失效，可能的解决方法就是利用GFS自带的工具诊断系统的故障，比如说它输出的详尽的、深入的诊断日志。GFS所要达到的目标就是高可用性、数据完整性。在测试基准性能方面，Google主要使用自己的例子来说明GFS的读写速度和Master服务器的负载均衡，在服务器出现错误时，它可以迅速恢复运作的能力。经验和相关工作方面主要是写在做GFS时遇到一些问题以及自己寻求的解决方法和方案。

2. 《Google Bigtable》

这篇论文主要介绍的是Google的Bigtable，它是一个分布式的结构化数据存储系统，用来处理海量数据，通常是PB级的数据。它的目标是适用性广泛、可扩展、高性能和高可用性。后面就是介绍它的一些组成和它的数据模型，首先可以看到数据模型，Bigtable是一个稀疏的、分布式的、持久化存储的多维度排序Map，而Map的是行关键字、列关键字以及时间戳。Bigtable还提供了API函数用来修改集群、表和列族的元数据。它的构件是建立在几个Google基础构件上的，它使用的是Google的分布式系统GFS，来存储日志文件和数据文件。其中还介绍了Bigtable依赖的一个高可用、序列化的分布式锁服务组件，叫做Chubby。Bigtable主要包括三个组件：链接到客户端程序中的库、一个master服务器和多个Tablet服务器。根据任务的变化情况，它还可以动态的向集群中添加Tablet服务器。并且Google使用一个三层的、类似B+树的结构存储Tablet的位置信息，位置信息不超过三层。比如有Tablet分配指的是一个Tablet只能分配给一个Tablet服务器。文章中还对Tablet

服务和它的内存空间回收机制做了详细的说明
上面主要讲的是Google Bigtable的实现构件以及它底层的框架，这一段主要讲讲它的优化工作以及一些性能数据。首先说的是优化方面所做的努力，（1）它使用了一个局部性群组，也就是客户端程序可以将多个列族组合成一个局部性群组，对Tablet中的每一个局部性群组都会产生一个单独的SSTable，将通常不会一起访问的列族分割成不同的局部性群组可以提高读取操作的效率。（2）使用了压缩的技术，客户程序可以控制一个局部性群组的SSTable是否需要压缩，当然通过压缩可以减少存储空间，而且没有影响读取的速度。（3）使用缓存来提高读操作的性能，Tablet 服务器使用二级缓存的策略。扫描缓存是第一级缓存，主要缓存 Tablet服务器通过 SSTable 接口获取的 Key Value 对；Block 缓存是二级缓存，缓存的是从 GFS 读取的 SSTable 的Block。技术的话，比如使用了Bloom过滤器、实现Commit日志、恢复提速、不变性。其次在性能评估方面，为了测试Bigtable的性能和可扩展性，Google建立了一个包括N台Tablet服务器的Bigtable集群，并且它的台数是可变的。Bigtable正是通过服务器台数的增加来实现系统整体吞吐量的增长，并且这种增跨越式的。
最后说说Bigtable的实际应用方面。（1）Google Analytics，用来帮助Web站点的管理员分析他们网站的流量模式的服务，提供了整体状况的统计数据。（2）Google Earth，Google通过一组服务为用户提供了高分辨率的地球表面卫星图像，访问方式可以是基于Web的Google Maps访问接口，也可以是通过Google Earth制定的客户端软件访问。（3）个性化查询，是一个双向服务，这个服务记录用户的查询和点击，涉及到各种Google 的服务，比如 Web 查询、图像和新闻。用户可以浏览他们查询的历史，重复他们之前的查询和点击；用户也可以定制基于 Google 历史使用习惯模式的个性化查询结果。在经验教训和相关工作方面，主要讲的是Google在做Bigtable时遇到的一些问题和解决方案。

3．《Google MapReduce》

本篇论文主要介绍的是Google的MapReduce编程模型，MapReduce是一个处理和生成超大数据集的算法模型。MapReduce架构程序能够在大量的普通机器上进行并行处理，它可以实现运行在规模可以灵活调整的普通计算机集群之上，依然是利用了大量的普通机器组成集群来提升自己的性能。Google为了处理并行计算问题、分发数据问题和处理错误，并且受到了来自Lisp语言的灵感，以至后面可以写出Map和Reduce的原语。
中间部分主要讲的就是MapReduce的原理以及它的使用环境。首先介绍的是MapReduce的编程模型原理，它主要是利用一个输入集合来产生一个输出集合，对应于MapReduce库就是使用两个函数来表示这个计算，Map和Reduce。MapReduce模型可以有多种不同的实现方式，它是根据实际情况来进行配置，在Google内部就是用以太网进行连接的多台普通PC机组成的集群来实现的。在执行运算时，Map和Reduce都可以将数据分成多段然后部署到多台机器上运行，以此来达到并行计算的目的。数据结构使用的是Master，它存储数据的大小和位置。MapReduce还有很好的容错机制，比如出现worker故障时，若已完成Map任务，则重新访问；若已经完成Reduce任务则不需要再次执行。后面还有Master失败、数据失效方面的容错处理等。MapReduce在运行过程中数据是存储在GFS上的。除了Map和Reduce两个处理函数外，MapReduce还存在许多的扩展功能，比如有分区函数、顺序保证、Combiner函数，它还支持多种不同格式的输入数据。
最后主要来看看MapReduce的性能和它的经验教训方面的内容，Google通过再一个大型集群上的两个计算来衡量MapReduce的性能，最后可以看到它的读写速度是惊人的，这也体现出了并行计算的优势。MapReduce编程模型可以应用到Google的多个领域，这主要归功于MapReduce封装了并行处理、容错处理、数据本地化优化、负载均衡等等技术。它也为Google解决了需要大量计算的问题。

纵观Google的三篇论文，《Google File System》、《Google Bigtable》和《Google MapReduce》其实描述的就是Google的三种技术，GFS分布式文件系统，Bigtable分布式数据存储系统，MapReduce编程模型，都是基于分布式并行运行的，都是部署在大量普通机器组成的集群之上，其实相互之间都有相似之处，也能协调在一起运行和工作。这也是在2003到2006年之间Google陆续发表的论文，三篇文章的重要目的就是解决分布式并行计算的问题。这也为大数据技术的发展和应用提供了可能。