**Google的三驾马车论文读后感**

宋廷泽 18301135

这学期学习了非关系型数据库这门课，老师推荐我们阅读Google的三驾马车（Bigtable,The Google File System,MapReduce）的论文，在读完论文后并通过部分资料的搜集，我对分布式系统以及大数据集处理有了初步的了解，其中对于Google的Bigtable系统感触更为深刻，接下来我将汇报一下我对Bigtable的简单理解以及读完论文的一些感受。

Bigtable是一个管理结构化数据的分布式存储系统，它被设计用来处理海量数据：分布在数千台通用服务器上的PB级的数据。它具有适用性广泛、可扩展、高效处理性能和高可靠性等特点，可以适应大量数据快速处理，可以把数据快速输送给大量用户，在Google很多产品中都有被使用。

 Bigtable使用了很多数据库的实现策略，与数据库很类似但是也有很多不同。比如：Bigtable提供了和并行数据库和内存数据库完全不同的接口；Bigtable为客户提供了简单的数据模型，利用这个模型，客户可以动态控制数据的分布和格式，用户也可以自己推测低层存储数据的位置相关性；Bigtable将存储的数据都视为字符串，但是Bigtable本身不去解析这些字符串，数据的下标是行和列的名字等等。

一．稀疏的数据模型

Bigtable的核心数据模型是一个稀疏的、分布式的、持久化存储的多维度排序Map，以 (行-row, 列-column, 时间戳-timestamp)为索引，在每个被索引的单元中存储对Bigtable不透明的用户数据。

1. 行：从逻辑上，Bigtable的数据是以行为单位以Row Key为index排序存储的，物理上，也是以row key为序存储的。此外Bigtable数据也是以行的范围来划分数据段（tablet）的，每个tablet可以由不同的TabletServer来管理，这样不同区段的数据由不同的Server节点服务，可以保证数据的吞吐率 。
2. 列族： Bigtable的列关键字组成的集合叫做“列族”，列族构成了访问控制的基本单位。分考虑到数据的稀疏性，以及不确定的Schema，具体列的数量是不限的，也无需事先声明。 但是，列族必须先创建，然后才能在列族中任何的列关键字下存放数据；列族创建后，其中的任何一个列关键字下都可以存放数据。
3. 时间戳：每行数据都会有用来当作版本号的时间戳，可以系统自动赋值，也可以用户自己指定。最新的数据行排在最前面。还可以利用时间戳来进行垃圾收集。

此外，Bigtable提供了建立和删除表以及列族的API函数。客户程序可以对Bigtable进行如下的操作：写入和删除Bigtable中的值，从每个行中查找值，或者遍历表中的一个数据子集。通过Wrapper类，Bigtable还可以作为MapReduce框架的输入和输出。

1. 两大控件

Bigtable构件是建立在其他几个Google基础构件上的。Bigtable使用Google的分布式文件系统GFS存储日志文件和数据文件。Bigtable集群通常运行在一个共享的机器池中，池中的机器还会运行其他的各种各样的分布式应用程序。

 Bigtable内部存储数据的文件是Google SSTable格式的。SSTable是一个持久化的，排序的，不可更改的Map结构，而Map是一个key-value映射的数据结构，key和value都是任意的Byte串。

 Bigtable还依赖一个高可用的，序列化的分布式锁服务器组件，叫做Chubby。Bigtable使用Chubby完成以下几个任务：

1.确保在任何时间内最多只有一个活动的Master副本；

2.存储BigTable数据的自引导指令的位置；

3.查找Tablet服务器，以及在Tablet服务器失效时进行善后；

4.存储BigTable模式信息；

5.存储访问控制列表。BigTable包括了三个主要的组件：链接到客户程序的库、一个Mater服务器和多个Tablet。针对系统工作负载的变化情况，BigTable可以动态的向集群添加或者删除Tablet服务器。

三．功能的实现

Bigtable的功能实现依靠三个主要部分：一个可以连接到每个用户的库文件，一个主要服务器控制其他，许多目录服务器来分配文件的储存。主服务器负责把目录分配到目录服务器上，检测目录服务器的改变，平衡目录服务器负载，对谷歌文件系统进行垃圾收集，还有控制不同列族等内容的改变。

每个tablet一次分配给一个tablet服务器。master服务器记录活跃的tablet服务器、当前tablet到tablet服务器的分配、包括哪些tablet还没有被分配。一个tablet 服务器管理着多个tablet。

BigTable使用Chubby跟踪记录Tablet服务器的状态。当一个Tablet服务器启动时，它在Chubby的一个指定目录下建立一个有唯一性名字的文件，并且获取该文件的独占锁。Master服务器实时监控着这目录，因此Master服务能够知道有新的Tablet服务器加入了。只要文件存在Tablet服务器就会试图重新获得对该文件的独占锁，如果文件不存在了，那么Tablet服务器就不能在提供服务了。

1.Master服务器从Chubby获取一个唯一的Master锁，用来阻止创建其它的Master 服务器实例；

2.Master服务器扫描Chubby的服务器文件锁存储目录，获取当前正在运行的服务器列表；

3.Master服务器和所有的正在运行的Tablet表服务器通信，获取每个Tablet服务器上Tablet的分配信息；

4.Master服务器扫描METADATA 表获取所有的Tablet的集合。在扫描的过程中，当Master服务器发现了一个还没有分配的Tablet，Master服务器就将这个Tablet 加入未分配的Tablet集合等待合适的时机分配。

四．读写操作

当片服务器收到一个写请求，片服务器首先检查请求是否合法。如果合法，先将写请求提交到日志去，然后将数据写入内存中的memtable。memtable相当于SSTable的缓存，当memtable成长到一定规模会被冻结，Bigtable随之创建一个新的memtable，并且将冻结的memtable转换为SSTable格式写入GFS，这个操作称为minor compaction。

当片服务器收到一个读请求，同样要检查请求是否合法。如果合法，这个读操作会查看所有SSTable文件和memtable的合并视图，因为SSTable和memtable本身都是已排序的，所以合并相当快。

为了提高读操作的性能，Tablet服务器使用二级缓存的策略，一级用来缓存Tablet服务器通过SSTable接口的Key-Value对；Block是二级缓存，用来缓存从GFS读取的SSTable的Block。  
 此外，Google还通过局部性群组，压缩，缓存，Bloom过滤器，提交日志，Tablet恢复提速等方法来优化功能，以此达到用户要求的高性能、高可用性和高可靠性的目标。

五．合并压缩

随着写操作的执行，memtable的大小会增加。当memtable的大小达到一个门限值时，这个memtable会被冻结，创建一个新的memtable，并将冻结的memtable转换成一个SSTable写入到GFS中。这里的次压缩（minor compaction）过程有两个目标：减少tablet服务器的内存使用，减少操作日志中在恢复tablet服务器时需要读取的数据总量。当压缩发生时，进来的读和写操作能够继续执行。

一个将所有SSTables写入到一个SSTable中的合并压缩称为主压缩（major compaction）。非主压缩产生的SSTable能够包含特定的删除条目，它阻止在仍然活着的旧SSTable中删除数据。另一方面，主压缩产生的SSTable不会包含删除信息或已删除的数据。Bigtable循环扫描所有的tablets，并定期的对它们执行主压缩。这些主压缩可以回收删除数据所使用的资源，并尽快的确保删除的数据在系统内彻底消失。

六．BigTable和GFS的关系

集群包括主服务器和片服务器，主服务器负责将片分配给片服务器，而具体的数据服务则全权由片服务器负责。但是不要误以为片服务器真的存储了数据（除了内存中memtable的数据），数据的真实位置只有GFS才知道，主服务器将片分配给片服务器的意思应该是，片服务器获取了片的所有SSTable文件名，片服务器通过一些索引机制可以知道所需要的数据在哪个SSTable文件，然后从GFS中读取SSTable文件的数据，这个SSTable文件可能分布在好几台chunkserver上。

1. 影响意义

整个BigTable设计符合大部分大数据程序的需求，打破了关系型数据库的结构化存储，能够部署在成千上万台服务器上，可以存储PB级数据，对整个互联网行业的快速发展提供了坚实的理论基础与成功案例。

八．感想总结

通过阅读并了解Google的三架马车（Bigtable,The Google File System, MapReduce），我发现这三项成果对很多nosql产生了比较的影响。比如：谷歌在2006年的论文中提出来的BigTable，为后来HBase的出现提供了理论基础。在当前信息化的时代，Google的三驾马车为大数据处理和存储提供了思路和方案，同时普及了云计算中非常核心的分布式技术。Google提出的这三大技术，不仅对自身产品实现创新和优化，同时促进了大数据和分布式的发展。

，