**GFS读后感**

GFS为Google File System，谷歌文件系统。在看完GFS论文后，给我最大的感触就是，它的文件系统并不是单纯的我们电脑上的文件系统，而是一个基于分布式的文件系统，在主服务器和其他服务器之间密切联系的文件系统。其考虑了很多对于性能方面的优化以及用户体验。Bigtable论文中讲到，它的内部文件系统就是使用的GFS，还和MapReduce有联系。可见这三篇论文所阐述的内容不仅仅是一个递进的关系，也是一个相互联系、相互依赖的关系。

在我们本地的文件系统中，在操作系统课程中讲过，Linux使用的是i-Node方式，即在Meta Block中有索引index指向相应的文件位置。当储存大文件时，就将许多小块组合成一个大块，这样虽然解决了存储大文件的问题，但是如果我们使用一大块硬盘来存储一个小文件时就会造成空间的浪费。

那么，存储超大文件时，就不能使用单机存储了，这样就涉及到了分布式，需要一个主从结构。只有一个Master，可以拥有多个Chunk Server。这样，超大文件存储在子服务器中，而Master主机只需要记录每一个所存储的服务器，以及在该服务器上的偏移量即可。这样的结构就是：

Master + many Chunk Server

但是这样的结构是有缺点的，也就是如果某一个文件发生了改变，都需要和主机做交互，因为主机记录着该文件在子服务器上的偏移量，偏移量可能会发生改变。这样可能会导致主机处理大量的请求。所以，解决这个问题，即解决高耦合的问题，我们可以将主机中记录的文件-服务器-偏移量的索引结构，更改成文件-服务器结构，而在每一个子服务器上的索引部分记录文件-偏移量。这样，只要文件做出了修改，只需要在子服务器上做一些改动即可，不需要通知主服务器更改，因为文件虽然做出了修改，但是还是在那个服务器上。这才是高内聚、低耦合的文件结构。

对于检验数据的损坏，可以使用checksum方法，这在计算机网络课上学过，可以用来校验数据是否出错。每个block中存储一个32位的checksum校验码。每读取一个数据块，都对其中的校验码进行检验，判断是否出错。

如何解决子服务器故障所造成的损失呢？在数据库理论课上我记得曾经讲过一个很普遍的方法就是数据冗余。在文件系统里面也是同理，可以将同一个文件保存在多个子服务器中，只需要在Master主机上记录，存储本文件的三个服务器分别是哪三个即可。这样任何一个服务器上的数据丢失，都可以使用副本来恢复。当一个Chunk Server上的某一个Chunk损坏时，向Master主机发送请求，寻找还拥有本数据块的服务器，然后在里面最近的服务器要该数据块的副本，即可恢复数据了。

如何判断一个服务器是否整行运行，即有没有发生宕机或者出故障呢？每一台子服务器每隔一段时间都对主服务器发送消息，代表本服务器运行正常，如果有一台服务器长时间没有反馈的话就代表服务器挂掉了。

读和写是一个文件系统的核心。GFS是如何进行读写操作呢？

基于上面所展示的文件存储方式我们知道，master机器上只知道拥有某文件的服务器都是哪个。所以，当客户应用程序需要读一个文件时，会向master发送请求，询问文件所在的服务器，并请求一个handle获取内容权限。这样，再向其中最近的server请求该数据即可。

对于数据的写操作，由于有三个副本，所以需要写三次。和读操作类似，先向主服务器发送请求，查找该需要写的数据块所在位置，master返回三个服务器，并设置一个服务器为本操作的主服务器。然后用户程序向三个当中最近的一台服务器发送写操作请求，本服务器再向其他两个服务器发送写的信息。因为服务器和服务器之间的带宽更高，所以服务器和服务器之间的信息传递更快。当三个服务器都进行了缓存，最近的服务器通知客户程序后，用户程序通知主服务器，让其通知三个服务器开启写入进程，即可修改三处的文件内容。

在看过GFS文件系统的论文后给我最深的感触就是，它应用了很多我们已经学过的知识，但是我在之前学习的时候并没有仔细想过这些知识居然能够如此去应用。它涉及到了之前学过的操作系统、数据库和计网当中学习到的一些小知识，文件索引啊，数据冗余解决出错啊，checksum校验码啊，这些不得不佩服应用这些知识的人们。虽然GFS算是一个高内聚、低耦合的文件系统吧，但是我发现仍然是需要一个Master和许多Chunk Server的一个主从结构的，所以说还是有去中心化的很多的工作要做。