分布式文件系统综合研究
——以 Google 文件系统为例

3180103772 张溢弛

一、分布式文件系统简介

1.1 分布式系统

分布式文件系统（Distributed File System）首先是一个分布式系统（Distributed System），而分布式系统的定义是什么呢？我查阅资料得知，分布式系统是一种建立在计算机网络上的软件系统，正是因为软件的特性，所以分布式系统具有高内聚性和透明性，在一个分布式系统中，一组独立的计算机构成了一个整体，仿佛一个完整的系统，拥有一定的资源，可以动态地分配和执行任务，不同的计算机之间通过计算机网络进行信息的交换。

而分布式系统中最重要的就是模型（也叫范型），负责协调和调度不同计算机之间的通信，这就是在操作系统的基础上构建一层软件中间件来实现的，一台台独立的计算机就是分布式系统中的一个个“节点”。

1.2 分布式文件系统

而分布式文件系统（DFS）就是说文件系统管理的物理存储资源不一定直接连接在本地节点上，而是通过一个分布式系统与一个节点相连，或者是若干不同的逻辑磁盘分区组合在一起而形成的完整的具有层次的文件系统，分布式文件系统为分布在计算机网络中的**任意位置的资源提供一个逻辑上的树形文件系统结构，从而使用户可以访问分布在网络中的共享文件，并且更加便捷和迅速。**

分布式文件系统的出现是为了解决高速发展的信息化社会中人们对于大容量存储的迫切需求，而增加硬盘个数的方式不管是在存储容量、数据备份还是安全性上都表现得逊色于分布式文件系统，人们使用分布式文件系统得时候不需要关系数据存储在哪个节点之上，或者需要从哪个节点获取，只需要把它们当作本地磁盘中存在的文件一样来访问即可。

分布式文件系统基于 C/S 的软件架构，将大量的文件分散到了不同的节点上存储，不仅增大了文件系统的存储量，也使得文件数据的安全性大大提高，就算有一部分节点出现了故

障，另一部分节点中的数据仍然可以正常存取，不会影响正常使用，与此同时，分布式文件系统还具有高并发性和非常强的可扩展性。

二、Google 文件系统

2.1 简介

Google 文件系统（GFS）是非常成功的分布式文件系统，被广泛地用于 Google 内部进行部署，虽然已经诞生了比较长的时间，但是依然是非常经典和具有代表性的分布式文件系统。因此我选择它作为本次调研的主要对象，并阅读了 Google 公司发表的经典论文《The Google File System》，根据该论文来学习和了解分布式文件系统的基本特性以及 Google 文件系统设计的成功之处。

GFS 是一个可扩展的分布式文件系统，设计的目的是用于大型的分布式数据密集型应用程序，可以使用非常廉价的计算机硬件设备来提供容错功能，并且为大量的客户端提供较高的聚合性能。此外，该文件系统使用大量经常发生故障的廉价硬件设备组成，因此会不断地自我监控并定期检测故障并恢复。支持各种规模和大小的文件的存储，并且支持对同一个文件的高并发访问。

2.2 文件系统架构

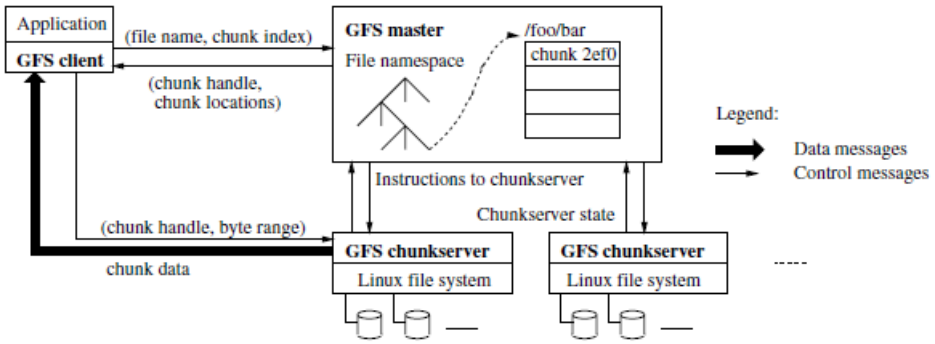
GFS 是一个计算机集群，而一个 GFS 由一台主服务器和多个块服务器组成，并且支持多个客户端的并发访问，文件被分成了若干固定大小的块（chunks），每个块被分配了一个 64 位的标识号，块服务器将块作为 Linux 文件存储在本地磁盘上，并且读取或者写入块数据，为了保证块中存储的数据的可靠性，每一个块被复制到多个块服务器上保存，而 GFS 的块大小设定成了 64MB，这也比经典的文件系统的块要大得多。

同时，主服务器维护所有文件系统的元数据，包括命名空间、访问控制信息、从文件到块的映射关系和块的位置等等，同时还进行块管理和垃圾回收。

而客户端可以调用文件系统的 API 并和 GFS 的主服务器和块服务器进行通信来进行数据的读写，其中和主服务器通信以获取文件系统的元数据，而读写等承载具体文件数据的通

信就直接发送给块服务器。

这就是 GFS 的基本架构，原论文中使用了下面的插图来形象地表示了这一精简的架构设计



2.3 GFS 优秀设计

2.3.1 块的设计

GFS 创造性地将文件的块大小设计成了 64MB，这笔典型的文件系统的块大小要大得多，选用大 size 的块有这样几个好处：

- 减少了客户和主服务器的交互，因为块变大使得需要读写的块的数目减少了
- 在较大的块上，客户端更有可能在给定的块上执行许多操作，因此可以通过在演唱的时间段内保持与块服务器的持续 TCP 连接来减少网络 I/O 的开销
- 减小了存储在主服务器上的元数据的规模，这使得主服务器中的元数据可以全部保留在内存中

2.3.2 取消文件数据缓存

GFS 的客户端和块服务器都不会缓存文件中的数据，这是因为大多数应用程序流经大文件或者工作集太大到底无法缓存，客户端的缓存也不能给客户端带来什么好处，因此 GFS 干脆取消了文件数据缓存这一功能，这就消除了缓存一致性的问题，从而简化了整个文件系统，块服务器不需要缓存文件数据，因为 Linux 的缓冲区缓存已经将经常访问的数据保存在了内存中。

2.3.3 日志系统

GFS 的日志系统包含关键的元数据更改的历史记录，这也是 GFS 的核心，它不仅是主服务器元数据的唯一的持久的记录，而且还用来定义并发操作顺序的逻辑时间表。每个文件和块的版本号均由创建时候的逻辑时间来唯一确定。

为了保证日志的可靠性，GFS 使用多台计算机来存储日志，并且仅在将响应的日志记录刷新到本地和远程磁盘之后才响应客户端的操作，主服务器在刷新之前将一批日志记录一起批处理，从而减少了刷新和复制对整个系统吞吐量的影响。

主服务器通过 redo 操作日志来恢复文件系统的状态，为了最大程度地缩短启动时间，必须使得日志的规模较小，因此每当日志的规模增长到一定规模的时候，主服务器就会检查状态并建立一个 checkpoint，这样一来这个 checkpoint 之前的日志记录就都可以认为是有效的，事务恢复阶段就可以直接从 checkpoint 开始进行相关的 redo 和 undo 操作，这也和我们上个学习所学的《数据库系统》课程中的日志和事务回滚的相关知识对应了起来，可谓是完全一致。

由于建立检查点可能要花费一些时间，因此主机的内部状态的构造方式可以在不延迟传入突变的情况下创建新的检查点。主服务器切换到新的日志文件，并在单独的线程中创建新的检查点。新的检查点包括切换之前的所有突变。对于具有数百万个文件的群集，可以在一分钟左右的时间内创建它。完成后，将其本地和远程写入磁盘。

2.3.4 原子的记录追加

GFS 提供了一种原子性的记录追加操作，在传统的文件写入中，客户端指定要写入数据的偏移量，并发写入一个同一个区域，该区域最终可能包含来自多个客户端的数据片段，但是在记录追加中，客户端仅仅指定数据，GFS 会以 GFS 选择的偏移量将这个数据至少一次原子性地写入文件中，然后再把偏移量返回给客户端。

这一操作在分布式的软件中被大量使用，这是为了避免复杂而高代价的同步操作（比如通过分布式锁），避免了文件的并发读写产生竞态条件。

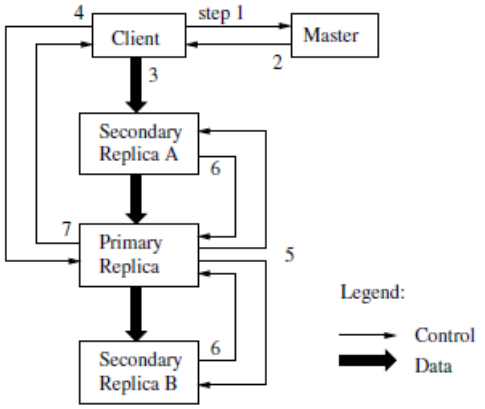
如果记录住家在任何副本上均失败，则客户端会重新尝试这一操作，导致的结果是副本可能包含不同的数据，GFS 不保证所有副本精确到每一个字节都是相同的，它仅仅保证客户端输入的数据被原子性地至少写入一次并报告操作成功。而一致性的问题又会有 GFS 的专

门子系统来解决。

2.4 基于租约的 GFS 工作流程

GFS 基于一种被称为租约的机制来进行工作，来应对数据的突变（元数据的更改也被叫做“变异”），进行记录的追加。租约机制是中心节点分配给块的某个副本一个被称为租约的锁，持有租约的副本才可以处理客户端的更新请求，客户端更新数据之前会从中心节点获取该块持有的租约的副本并向其发送更新请求。

租约本质上是一种有时间限制的锁，租约的持有者需要定期向中心节点申请续约，如果超过了期限就会被收回，而数据写入过程中，这一过程又可以细分为如下几步：



- 客户端询问主服务器，获取租约和副本的位置，当没有块有租约的时候就由主服务器进行分配
- 主服务器答复主服务器的身份和副本的位置，客户端缓存这一元数据（注意不是文件数据，和上面提到的并无矛盾），以备用，仅当主要节点无法访问或者不再持有租约的时候才需要再次和主节点联系
- 客户端将数据推送到所有副本，客户可以按照任何顺序进行操作，每个块服务器将数据存储在内部的 LRU 高速缓存中
- 一旦所有副本都已经确认收到数据，客户端将向主数据库发送写的请求，该请求表示了更早推送到所有副本的数据，主服务器为可能从多个客户端接收到的所有突变分配连续的而序列号并进行必要的序列化
- 主服务器将请求转发到所有的辅助副本，每个辅助副本均以主副本分配的相同序列号顺序应用突变

- 所有辅助副本答复主副本完成了响应
- 主副本给客户端返回写入成功的响应

2.5 总结

以上是我在阅读论文之后学习到的 GFS 的一些基本特性和系统架构，我们可以认识到这一大规模分布式文件系统的设计是非常精妙和有效的，无论是大容量的块设计，还是日志系统和租约的设计，都在为“大规模”和“分布式”的文件系统服务。下面一节我们将重点探究近几年来尤其是国内的分布式文件系统的发展的最新情况。

三、分布式文件系统发展现状

除了 GFS 之外，国外比较著名的分布式文件系统还有 HDFS，这是专门为 Hadoop 这样的云计算系统设计的分布式文件系统，在离线批量处理大数据上有先天的优势，HDFS 是 GFS 的开源实现；而 Ceph 是一个可以按对象/块/文件方式存储的开源分布式文件系统，其设计之初，就将单点故障作为首先要解决的问题，因此该系统具备高可用性、高性能及可 扩展等特点。该文件系统支持目前还处于试验阶段的高性能文件系统 BTRFS(B-Tree 文件系统)，同时支持按 OSD 方式存储，因此其性能是很卓越的， 因为该系统处于试商用阶段，需谨慎引入到生产环境；Lustre 是一个由 SUN 公司开发和维护的大规模、安全可靠的、具有高可用性的下一代集群文件系统，针对的是小文件和图片的分布式存储中遇到的各种痛点。

而我国国内的一些著名互联网企业也纷纷推出了自己的分布式文件系统，比较知名的有这样几个：

- 百度文件系统（BFS）：和 HDFS 等专门为离散批处理设计的分布式文件系统不同，百度文件系统针对百度的搜索引擎的搜索业务的特点，可以在高吞吐的情况下做到低延迟和可持续用，具有持续可用，高吞吐，低延时和水平扩展的特性，通过 Raft 维护一致性，实现了全局负载均衡，慢节点自动规避
- 淘宝文件系统（TFS）：针对高可用性、高性能、低成本的分布式文件系统，TFS 是一种基于 linux 的文件系统，通过冗余、备份和负载均衡技术，提供高可靠性和并发访问。

TFS 主要为小于 1MB 的小文件设计。它采用平面结构代替了传统的目录结构。TFS 将在存储用户上传的数据后生成一个 18 字节长的文件名。用户可以通过 uniqle 文件名访问他们的数据。而阿里巴巴集团后续又开发出了轻量级的 FastDFS，解决了大容量存储和负载均衡的问题，特别适合以文件为载体的在线服务，比如视频网站和相册网站等等

- 京东的 ChubaoFS（CFS）：是京东开发的分布式文件系统和对象存储系统，是一个云原生的分布式文件系统，支持多源数据服务器和 POSIX 接口，主要用于 k8s 容器环境，支持容器化的存储和容器对文件的并发访问，是一个可持久化、高并发访问、可共享访问的文件存储系统。

四、总结与展望

我们从中可以感受到，无论是国内还是国外，分布式文件系统近几年以来都出现了比较大的进步，具体的发展趋势是分布式文件系统的设计目标已经不满足于大容量和分布式，而对访问速度提出了更高的要求，也对分布式文件系统的应用领域进行了进一步细化，出现了一大批针对特定场景（如小文件、视音频的存储）设计的分布式文件系统。

因此，在阅读了这篇论文和其他相关资料之后，我认为未来分布式文件系统的发展趋势是更加精细化和专业化，针对不同的大容量文件存储领域开发出更多解决特定需求的分布式文件系统，并和云计算、人工智能等热点技术相结合，产生更高的性能和更多的功能。