GFS

阅读了GFS这篇论文后，我对GFS了解如下：

谷歌文件系统，这是一个可扩展的分布式文件系统，适用于大型分布式数据密集型应用。它在廉价的商用硬件上运行时提供容错，并向大量客户端提供高聚合性能服务。

1. GFS与以往的文件系统的不同点有：

1、组件故障不再被当作例外，而是将其作为常态处理。该文件系统由数百甚至数千个由廉价商品部件构建的存储机器组成，并可由数量相当的客户端机器访问。组件的数量和质量实际上保证了一些组件在任何给定的时间都不起作用，而一些组件将无法从当前的故障中恢复。因此，持续监控、错误检测、容错和自动恢复必须是系统的组成部分。

2、文件按传统标准来说是巨大的。大小达几个GB的文件是很常见的。每个文件通常包含很多应用程序对象。当我们定期处理由数十亿个对象组成的快速增长的大小达TB的数据集时，即使文件系统能够支持，管理成千上万的KB规模的文件块也是不方便的。因此，GFS重新考虑设计假设和参数，如输入/输出操作和块大小。

3、大多数文件的更新是通过添加新数据完成的，而不是改变已存在的数据。文件中的随机写入实际上是不存在的。一旦写入，文件是只读的，并且通常是有顺序的。各种数据都具有这些特征。一些数据可能组成一个大仓库以供数据分析程序扫描。有些是运行中的程序连续产生的数据流。有些是档案性质的数据，有些是在某个机器上产生、在另外一个机器上处理的中间数据。由于这些对大型文件的访问方式，添加操作成为性能优化和原子性保证的焦点。而在客户机中缓存数据块则失去了吸引力。

4、共同设计应用程序和文件系统应用编程接口通过增加我们的灵活性使整个谷歌文件系统受益。例如，放宽GFS的一致性模型，以极大地简化文件系统，而不会给应用程序带来繁重的负担。还引入了原子追加操作，这样多个客户端可以同时追加到一个文件中，而无需它们之间的额外同步。

二、设计概述

1、假设

谷歌文件系统由许多通常会出现故障的廉价商品组件构成。它必须不断地自我监控，并在常规的基础上检测、容忍和迅速恢复组件故障。多个GB大文件是常见情况，应进行高效管理。小文件必须得到支持，但不需要对它们进行优化。支持文件中任意位置的小写操作，但不一定要高效。

2、连接

GFS提供了一个熟悉的文件系统接口，尽管它没有实现标准的应用编程接口，如POSIX。文件在目录中分层组织，并由路径名标识。支持创建、删除、打开、关闭、读取和写入文件的常规操作。

此外，GFS还有快照和记录追加操作。快照以低成本创建文件或目录树的副本。记录追加允许多个客户端同时向同一个文件追加数据，同时保证每个客户端追加的原子性。

3、体系结构

1）一个GFS集群由一个master和大量的chunkserver构成，并被许多客户（Client）访问。

2）文件被分成固定大小的块。每个块由一个不变的、全局唯一的64位的chunk－handle标识，chunk－handle是在块创建时由 master分配的。

3）Master维护文件系统所以的元数据（metadata），包括名字空间、访问控制信息、从文件到块的映射以及块的当前位置。

4）与每个应用相联的GFS客户端代码实现了文件系统的API并与master和chunkserver通信以代表应用程序读和写数据。

5）客户端和chunkserver都不缓存文件数据。

4、单master

只有一个master也极大的简化了设计并使得master可以根据全局情况作出先进的块放置和复制决定。

5、块大小

块大小是设计中的一个关键参数。选择的是64MB，这比一般的文件系统的块大小要大的多。

块大小较大的好处有：

1）减少client和master之间的交互。

2）Client在一个给定的块上很可能执行多个操作，和一个chunkserver保持较长时间的TCP连接可以减少网络负载。   
3）这减少了master上保存的元数据的规模，从而可以将元数据放在内存中。

不好的一面：

一个小文件可能只包含一个块，如果很多Client访问该文件，存储这些块的chunkserver将成为访问的热点。

6、元数据

master存储了三种类型的元数据：文件的名字空间和块的名字空间，从文件到块的映射，块的副本的位置。所有的元数据都放在内存中。

7、一致性模型

GFS有一个宽松的一致性模型，它很好地支持我们高度分布式的应用程序。

三、系统交互

1、租约（lease）和修改顺序

租赁机制旨在最大限度地减少主机的管理开销。

修改顺序是一种改变块的内容或元数据的操作，如写入或追加操作。

2、数据流

为了有效的利用网络，将数据流和控制流分离。

3、原子记录追加

GFS提供一个原子性的追加操作：record append。

4、快照

快照操作几乎瞬间复制了一个文件或目录树，同时将正在进行的其他修改操作对它的影响减至最小。

四、MASTER操作

MASTER执行所有名字空间的操作，除此之外，他还在系统范围管理数据块的复制：决定数据块的放置方案，产生新数据块并将其备份，和其他系统范围的操作协同来确保数据备份的完整性，在所有的数据块服务器之间平衡负载并收回没有使用的存储空间。

1、名字空间管理和锁定

允许多个操作处于活动状态，并在命名空间的区域上使用锁来确保正确的序列化。

与传统文件系统不同的是，GFS没有与每个目录相关的能列出其所有文件的数据结构，它也不支持别名。

MASTER在执行某个操作前都要获得一系列锁。 MASTER操作的并行性和数据的一致性就是通过这些锁来实现的。   
2、副本放置  
一个GFS集群文件系统是多层分布的。

区块副本放置策略有两个目的:最大化数据可靠性和可用性，以及最大化网络带宽利用率。

3、创建、重新复制、重新平衡数据块

当MASTER创建新的数据块时，如何放置新数据块，要考虑如下几个因素：

1）尽量放置在磁盘利用率低的数据块服务器上，这样慢慢地各服务器的磁盘利用率就会达到平衡。

2）尽量控制在一个服务器上的“新创建”的次数。

3）把数据块放置于不同的机架上。

MASTER在可用的数据块备份低于用户设定的数目时需要进行重新复制。这种情况源于多种原因：服务器不可用，数据被破坏，磁盘被破坏，或者备份数目被修改。每个被需要重新复制的数据块的优先级根据以下几项确定：第一是现在的数目距目标的距离，对于能阻塞用户程序的数据块，提高它的优先级。

最后， MASTER按照产生数据块的原则复制数据块，并把它们放到不同的机架内的服务器上。

4、垃圾收集

文件删除后，GFS不会立即回收可用的物理存储。它只是在文件和块级别的常规垃圾收集期间才这样做

当一个文件被应用程序删除之后，MASTER会立即记录下这些变化，但文件所占用的资源却不会被立即收回，而是重新给文件命了一个隐藏的名字，并附上了删除的时间戳。在MASTER定期检查名字空间时，它删除超过三天（可以设定）的隐藏的文件。在此之前，可以以一个新的名字来读文件，还可以以前的名字恢复。当隐藏的文件在名字空间中被删除以后，它在内存中的元数据即被擦除，这就有效地切断了它和所有数据块的联系。

5、 过时副本检测

在数据更新时如果服务器停机了，那么他所保存的数据备份就会过时。对每个数据块，MASTER设置了一个版本号来区别更新过的数据块和过时的数据块。当MASTER授权一个新的lease时，他会增加数据块的版本号并会通知更新数据备份。MASTER和备份都会记录下当前的版本号，如果一个备份当时不可用，那么他的版本号不可能提高，当ChunkServer重新启动并向MASTER报告他的数据块集时，MASTER就会发现过时的数据。

五、容错和诊断

1、 高可用性

1） 快速恢复

不管如何终止服务，MASTER和数据块服务器都会在几秒钟内恢复状态和运行。

2） 数据块备份

每个数据块都会被备份到放到不同机架上的不同服务器上。对不同的名字空间，用户可以设置不同的备份级别。在数据块服务器掉线或是数据被破坏时，MASTER会按照需要来复制数据块。

3） MASTER备份

为确保可靠性，MASTER的状态、操作记录和检查点都在多台机器上进行了备份。

2、 数据完整性

每个数据块服务器都利用校验和来检验存储数据的完整性。原因：每个服务器随时都有发生崩溃的可能性，并且在两个服务器间比较数据块也是不现实的，同时，在两台服务器间拷贝数据并不能保证数据的一致性。

3、 诊断工具

广泛而细致的诊断日志以微小的代价换取了在问题隔离、诊断、性能分析方面起到了重大的作用。GFS服务器用日志来记录显著的事件（例如服务器停机和启动）和远程的应答。远程日志记录机器之间的请求和应答，通过收集不同机器上的日志记录，并对它们进行分析恢复，完整地重现活动的场景，并用此来进行错误分析。

以上就是我对GFS的粗略了解。