Simple Coherency Model

HDFS applications need a write-once-read-many access model for files. A file once created, written, and closed need not be changed except for appends and truncates. Appending the content to the end of the files is supported but cannot be updated at arbitrary point. This assumption simplifies data coherency issues and enables high throughput data access. A MapReduce application or a web crawler application fits perfectly with this model.

简单一致性模型：

HDFS应用程序需要一次只读一次读取文件的访问模型。 除了追加和截断之外，无需更改创建，写入和关闭的文件。 支持将内容附加到文件末尾，但无法在任意点更新。 此假设简化了数据一致性问题，并实现了高吞吐量数据访问。 MapReduce应用程序或Web爬网程序应用程序完全适合此模型。

要把问题说清楚前，先要把“内部一致性”和“外部一致性”这两个名词定义清楚。

“内部一致性”搞数据库的人很少这么说，一般就直接说一致性，更准确的说是“Consistency in ACID”（“事务 ACID 属性中的一致性”）。需要跟分布式领域的 Consistency 做一下澄清。自从云计算蓬勃发展之后，集群环境下的数据库服务越来越平民化，数据库和分布式原本两个有交集但交集不多的领域开始水乳交融，也就出现了名词打架的现象。打的最厉害的荣耀王者就是“Consistency”（一致性）。事务的 ACID 属性中有一个 C 是 Consistency，表示事务的执行一定保证数据库中数据的约束不被破坏。而分布式领域提及的 Consistency 表示系统的正确性模型，著名的也是臭名昭著的 CAP 理论中的 C 就是这个范畴的。CAP 理论的 Consistency 严格指 Linearizability[1] 这个一致性模型，与 ACID 中的 Consistency 没有半毛钱关系。

CAP理论：

CAP理论主要是针对分布式存储系统的，C是指Consistency一致性，A是指Availability可用性，P是指Partition tolerance分区容忍性。CAP定理认为分布式系统中这三个特性最多只能同时满足两个特性。

 一致性(Consistency): 指在分布式系统中的所有数据备份，在同一时刻是否同样的值。（等同于所有节点访问同一份最新的数据副本）

 可用性(Availability): 在集群中一部分节点故障后，集群整体是否还能响应客户端的读写请求。（对数据更新具备高可用性）

 分区容忍性(Partition tolerance): 即当节点之间无法正常通信时，就产生了分区，而分区产生后，依然能够保证服务可用，那么我们就说系统是分区容忍的。显然如果节点越多，且备份越多，分区容忍度就越高（因为即便是其中一个或多个节点挂了，仍然有其它节点和备份可用）。

为什么说三个特性无法全部保证呢？

首先，假如我们要保证分区容忍性，必然要做多个副本节点，而这必然会带来一致性的问题，即保证多个节点的数据是相同的，但是，要让多个节点数据相同，就必须要花时间去复制数据，这还是能够正常通信的情况下，那么在数据复制的过程中为了保持一致性，就不能对外提供服务，所以这段时间就无法满足可用性的问题。

参考资料：

[1] <https://www.zhihu.com/question/56073588/answer/253106572>

[2] <https://www.jianshu.com/p/2c30d1fe5c4e>