|  |
| --- |
| Google file system  Paper Report |
| Student names Linna Hu  Jiashuo Sun  Hongyu Huang  Stony brook  AMS 560 |



**Review of GFS**

# Summary

With the number of users growing, there is an urgent need of progressive file system for data-intensive applications. This paper mainly talks about how to design and implement the Google File system (GFS) to fulfill Google’s specific situation.

1. This system provides fault tolerance by constant monitoring, replicating crucial data, and fast and automatic recovery.

2. Because the Multi-GB files are the common and should be managed efficiently, the parameters and chunk size should be reconsidered.

3. Files are mostly mutated by appending new data. Appending gives the performance optimization and atomicity guarantees.

GFS maintains file system metadata by only one single master, also can controls system-wide activities by HeartBeat monitoring and can reduce network overhead and the size of the metadata. This system uses large chunk size to make that happen.

**The contributions**

**Snapshot:** ①Quickly creating branch copies of huge data. ②checkpointing the current state.

**Record append:** ①To enable users to append record on the same file more efficiently. ②To make primary chunk get 2 replicas and try their operation in order.

**Fault tolerance** : ①Using shadow master and master replication to ensure the master’s reliability. ②Having chunk replicas recovery by master detecting the chunkserver offline or data. ③Chunk-servers use checksumming to detect corruption of stored data

## The comments

GFS enables the system to run efficiently against the problem of large-size files and normal components faults. Although a single master controls the overall system, it only transmits a small amount of metadata to avoid network congestion and decrease efficiency. Authorized by master, the primary replica is responsible for program running and secondary replicas operation supervision, so as to achieve concurrently access by hundreds of clients.

**The flash point:** Using central server mode, no data caching, high aggregating throughput to many concurrent readers and writers performing a variety of tasks, separating the control flow and data flow.

**The limitation**: Based on the large-size files, GFS could not handle with too many small files perfectly because of the *hot spots*.