Paper Review Title: *The Google file system*, S. Ghemawat, H. Gobioff and S-T. Leung, Proc. of the 19th ACM symposium on Operating systems principles, October 2003, pp. 29-43.
Reviewer: swayanshu shanti Pragnya

1. Summarize the (at most) 3 key main ideas.

The paper is all about the design and implication of a scalable distributed file system for immensely large distributed applications.

1. Design over view of "Google file system" which is a scalable distributed file system
2. System interactions which includes data flow, mutation order and atomic record append.
3. Master in operations which includes the functionalities as-
   Replica placement, garbage collection and stale replica detection
4. Diagnosis and fault tolerance

2. State the main contribution of the paper

The primary contributions are-

1. **System interaction which includes**-

   Mutation order:- Each mutation is performed at all the chunks of replicas

2. Master operations:-

   Name space management and locking. Ex- Read-lock-on/dir1/dir2

   Replica replacement which maximizes the data reliability and availability.

   Garbage collection

3. **Fault Tolerance and Diagnosis**

   High availability: Strategy includes fast recovery and replication

4. Different measurements which includes **microbenchmarks, real world clusters, workload break down**

3. Critique the main contribution

3.a. Rate the significance of the paper on a scale of 5 (breakthrough), 4 (significant contribution), 3 (modest contribution), 2 (incremental contribution), 1 (no contribution or negative contribution). Explain your rating in a sentence or two.

It is a significant contribution so I would rate as 5 because the master operation includes the following attributes-

1. **Name space management and locking**.

Example- Read-lock-on/dir1/dir2

Each absolute file name or directory name has associated with read and write lock.

2. **Replica replacement**:- This is used for 2 purposes which includes data reliability and availability.
3. Creation, replication and balancing will help to replicate the chunks as soon as number of available replicas can be prioritized.
4. Another point is garbage collection and stale replica detection which may become the stale if a chunk server fails and misses mutations.
5. **Fault tolerance** is another primary function and problem which we generally face in file system management but this "Google file system" has high availability and most primarily data integrity with diagnostic tools.

1. ***b. Rate how convincing is the methodology: do the claims and conclusions follow from the experiments? Are the assumptions realistic? Are the experiments well designed? Are there different experiments that would be more convincing? Are there other alternatives the authors should have considered? (And, of course, is the paper free of methodological errors.)***

All the methodologies are convincing which includes the following majors,

1. Redundancy enabled reliability which helps distributed storage systems from system crash or network failure.

2. Designing the file system using GFS client for metadata operation is easy to exhibit.

3. System interactions and workflow from master to secondary Replica appends the duplicate records.

4. Data flow is decoupled using TCP pipeline which is well explained.

5. Serialized atomic record appends concurrent records.

3. c. What is the most important limitation of the approach?

Though it's a brilliant paper but still have few limitations like,
1. Map Reduce functionality
2. While appending large files, it takes so much time so time consumption is a draw back
3. Requires high throughput.
4. Tuning of applications for Google file systems
5. Handling with chunk server failures.

4. Rate the writing in the paper on a scale of 5 (great) to 1 (muddled), and justify your ranking. Did you have to re-read sections? Were algorithms clearly explained? Did the paper have a logical flow?

Writing-5

I would say its really a good paper as the major issue of handling big distributed files can be solved by this architecture.

The paper had a structural flow and proper explanation of methodology like starting from the paper objective, recovery methodology, limitations, future work for improvising the performance and correctness are well explained.

As the authors clearly explained the design, implementation and method behind each protocol, it's easy to understand the logic & architecture.

5. Answer one of the following three questions (whichever is most relevant for this paper):

1. What lessons should system researchers and builders take away from this work?2. What is the lasting impact of this work? 3. What (if any) questions does this work leave open?

By answering the question 1 these are the following points which researchers can work further,

1. Improvising the chunk servers

2. Increasing large scale file processing

3. Upgrading the network stack by increasing the throughput

4. Optimization of huge files with sequential read.

5. Operational issues like production systems can be controlled by fast recovery.