Aakash Saini(2018CSM1001)

March 11, 2019

- **Observations:** Paper proposes a distributed file system that supports scalability and reliability in the case of data-intensive applications. In GFS architecture supports: (i)master that holds complete file system metadata in memory and (ii) several chunk servers, that holds data replicas and responds to client requests. LRU buffer is maintained at the chunk servers. To reduce the client-master interaction, data is divided into the chunks(64MB each) and thus network overhead and metadata size get reduced. GFS client requests to master to find the location of chunks it needed. Master responds to the client by returning the chunk-server address. Chunks are replicated to three chunk-servers and one replica is designated to the master. Client can read from any of these three replicas, it chooses the closest one. Atomicity and correctness is maintained by namespace locking. Regular handshaking is done between Master and chunk-servers which in turn detects the data corruption also. Master performs replica replacement, re-replication, garbage collection,stale replica detection and re-balancing. For recovering the state of master, checkpoints are used at master server.

- **Limitations:**For the proposed system, small files can increase the overhead of metadata which need to be stored in main memory of master. In the paper it is not cleared that what information of chunk-servers has to be kept by master. Chunk-server may get overloaded if a single file is accessed by several clients. GFS client layer may limits the domain of file system. Role of the shadow master is not clear in the proposed work.

- **Conclusions:** GFS signifies the characteristics that are required to support data-intensive applications. GFS provides the fault tolerance by regular monitoring and replicating the important data. Checksumming helps to detect the frequent data-corruption at the disks. GFS is able to achieve high throughput even in the case of several concurrent readers and writers. GFS supports all storage system requirements and is used within Google as an important tool to handle the large amount of data queries.

- **Future work:**As the present work doesn't support fault tolerability at master node, so master node can be fixed and efficient peer-to-peer modifications can be considered to provide fault tolerance.