Fault Tolerance in Big Data Processing using Heartbeat Messages and Data Replication

T.Cowsalya[1] N.Gomathi[2] R.Arunkumar[3]
[1,2,3]Assistant Professor
[1,2,3]Department of Computer Science and Engineering
[1,2,3]SVS College of Engineering, Coimbatore, Tamil Nadu, India, Pincode-642109

Abstract— Big data is a popular term used to define the exponential evolution and availability of data, includes both structured and unstructured data. The volatile progression of demands on big data processing imposes heavy burden on computation, communication and storage in geographically distributed data centers. Hence it is necessary to minimize the cost of big data processing, which also includes fault tolerance cost. Big Data processing involves two types of faults: node failure and data loss. Both the faults can be recovered using heartbeat messages. Here heartbeat messages acts as an acknowledgement messages between two servers. This paper depicts about the study of node failure and recovery, data replication and heartbeat messages.

Key words: Big data, Fault Tolerance, Heartbeat Messages, Node Recovery, Data Replication

I. INTRODUCTION

Big data is a slogan, used to define a gigantic measurement of both structured and Unstructured data that is so large and difficult to process using traditional database architecture. Due to its explosive growth the volatile progression of demands on big data processing imposes heavy burden on computation, communication and storage in geographically distributed data centers. The incoming large data set is broken up into multiple chunks and each individual multiple chunks are placed in different data canters with the help of volley system. The Volley System [2] makes use of logs to submit the jobs to the data center. Cloud users make use of volley system foe automatic data placement.

A. Geo-Distributed Data Center:

The data centers distributed at multiple geographical regions are known as geographically distributed data centers [1].For example Google has 13 datacenters over 8 countries and 4 continents.

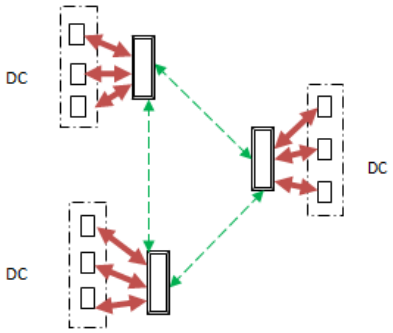

Fig. 1: Data Center Topology

II. FAULT TOLERANCE

The challenge of big data includes analysis, capture, search, sharing, storage, transfer, visualization and privacy violations. Among these challenges fault tolerance is one of the main challenge in big data. There are possibly two faults that can occur while processing big data. First the data chunk may loss while transferring the data to multiple data center. Second the server may fail or slows down.

A. Heartbeat Messages;

The solution for the above two problems are heartbeat messages. Here heartbeat message is a message sent from an inventor to the endpoint to identify if and when the inventor fails or is no longer available. Heartbeat messages are non-stop on a periodic recurring basis from the inventor's startup until the inventor's shutdown. When the receiver identifies lack of heartbeat messages during an anticipated arrival period, the destination may determine that the inventor has failed, shutdown, or is generally no longer available.

The developmentrelays to fault recovery in multiprocessor system, where the processors constantly monitor heartbeat messages from the other processor is capable of taking autonomous recovery action in response to a failure to receive heartbeat messages, advantageously without the overall guidance of an executive processor.

B. Data Loss Prevention:

When transmitting the jobs to multiple data centers, there may be chance of data loss. Data loss may occur due to network link failure. The links in networks may vary on transmissionrates according to their unique features. For example the distances and optical fiber facilities between multiple

data centers. Due to capacity constraints, all tasks are not placedonto the same server, on which theconsistentdata exist in. It is unavoidable when certain data must bedownloaded from a remote server. In this case, routingplan matters on the transmission cost.

C. Hadoop Architecture:

Hadoop is a software framework used for processing big data in parallel. It consists of two important components called Hadoop Distributed File System and MapReduce

1) Hadoop Distributed File System:

The Hadoop Distributed File System (HDFS)[3] is a distributed, highly fault-tolerant file system designed to run on low-cost commodity hardware. HDFS provides high-throughput access to application data and it is suitable for applications with large data sets. HDFS consists of two nodes called name node and data node.Name node manages file system namespace operations like opening, closing, and renaming files and directories. A name node also maps data blocks to data nodes, which handle read and write requests from HDFS clients. Data nodes also create, delete, and replicate data blocks according to instructions from the governing name node. Name Node and Data Node send messages to prove their identity.

2) Mapreduce:

MapReduce [12] is a programming model and its associated implementation for processing and generating large data sets with parallel, distributed algorithm on a cluster. MapReduce also consists of two nodes called job tracker and task tracker. The JobTracker talks to the NameNode to determine the location of the data. The Task Tracker node executes the assigned tasks in the data nodes.

III. FAILURES

One of the major benefits of using Hadoop is its ability to handle these failures and allow our job to complete.

A. Task Failure:

When the user code in the map or reduce tasks throws runtime exception, then the child task fails. Another failure mode is the sudden exit of child JVM. In this case the task tracker notices the process has exited and marks the attempt as failed. A task may also be killed, which is different from failing.

B. Task Tracker Failure:

If a task tracker fails by crashing or running very slowly, it will stop sending heartbeat messages to the job tracker. The job tracker notice that the task trackers has stopped sending heartbeat and remove it from its pool of task trackers to schedule tasks on.

C. Job Tracker Failure:

Failure of job tracker is the most serious failure mode. It is a single point of failure. This failure mode has low chance of occurring, since the chance of particular machine failing is low.

D. Name Node Failure:

The name node was a single point of failure, so if it failed that meant your cluster became unstable. Even the secondary name node doesn't help in this case since it is only used for checkpoints, not as a backup for the name node. If the name node fails someone like an administrator would have to restart the name node.

E. Data Node Failure;

A compute node can fail for any variety of reasons, for example broken node hardware, a broken network, software bugs, or inadequate hardware resources

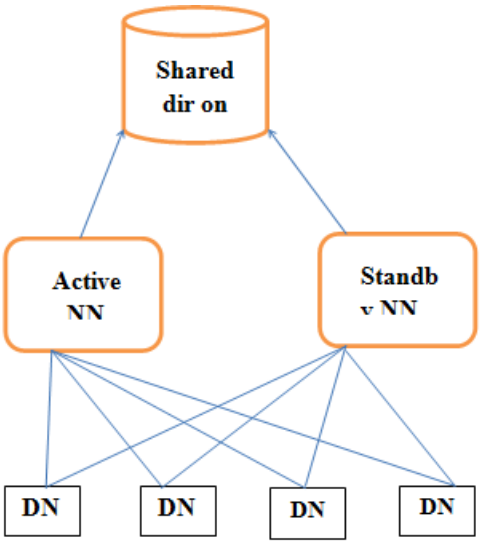

Fig. 2: Name node Schema

When a compute node fails, all jobs running on that node fail. Even though the running jobs running on other nodes that weren't communicating with jobs on the failed node will continue to run without a problem.

IV. SOLUTION

A. Data Replication:

An application can specify the number of replicas of a file at the time it is created, and this number can be changed any time after that. [6] The name node makes all decisions concerning block replication. HDFS uses an intelligent replica placement model for reliability and performance. Optimizing replica placement makes HDFS unique from most other distributed file systems, and is facilitated by a rack-aware replica placement policy that uses network bandwidth efficiently.

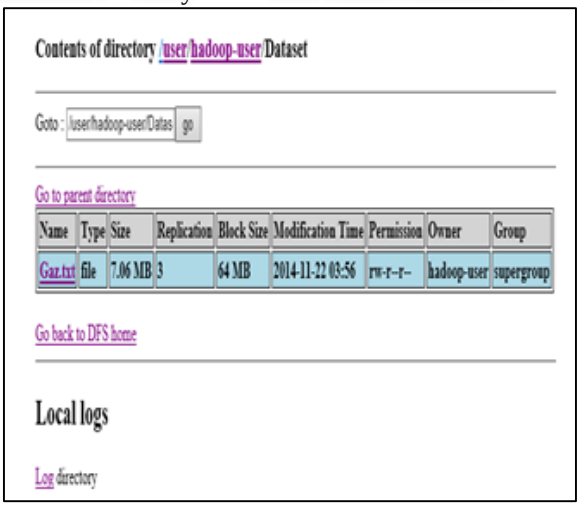

Fig. 3: Data Replication in DFS

The name node makes all decisions regarding replications of blocks. It periodically receives a heartbeat and a block report from each of the data nodes in the cluster. Receipt of a heartbeat implies that the data node is functioning properly. A block report contains a list of all blocks on a data node.

B. Replica Selection:

To minimize global bandwidth consumption and read latency, HDFS tries to satisfy a read request from a replica that is closest to the reader. If there exists a replica on the same rack as the reader node, then the replica is preferred to satisfy the read request. If HDFS clusters spans multiple data centers, then a replica that is resident in the local data center is preferred over any remote replica.

V. CONCLUSION

Thus the failure in big data processing has been studied. Data replication and heartbeat messages are used as a fault tolerant mechanism. In future practical setups can be executed and the computation and communication cost can be computed. Result can be compared with the cost of data processing in non- failure node

REFERENCES

[1] Data Center Locations," http://www.google.com/about/data centers/inside/locations/index.html.
[2] S. Agarwal, J. Dunagan, N. Jain, S. Saroiu, A. Wolman, and H. Bhogan, "Volley: Automated Data Placement for Geo-DistributedCloud Services," in The 7th USENIX Symposium on Networked Systems Design and Implementation (NSDI), 2010, pp. 17–32.
[3] L. Rao, X. Liu, L. Xie, and W. Liu, "Minimizing Electricity Cost: Optimization of Distributed Internet Data Centers in a Multi-Electricity-Market Environment," in Proceedings of the 29th International Conference on Computer Communications (INFOCOM). IEEE, 2010, pp. 1–9.
[4] A. Qureshi, R. Weber, H. Balakrishnan, J. Guttag, and B. Maggs, "Cutting the Electric Bill for Internet-scale Systems," in Proceedings of the ACM Special Interest Group on Data Communication (SIGCOMM). ACM, 2009, pp. 123–134.
[5] R. Urgaonkar, B. Urgaonkar, M. J. Neely, and A. Sivasubramaniam, "Optimal Power Cost Management Using Stored Energyin Data Centers," in Proceedings of International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS).ACM, 2011, pp. 221–232.
[6] X. Fan, W.-D.Weber, and L. A. Barroso, "Power Provisioning for A Warehouse-sized Computer," in Proceedings of the 34th Annual International Symposium on Computer Architecture (ISCA). ACM, 2007, pp. 13–23.
[7] S. Govindan, A. Sivasubramaniam, and B. Urgaonkar, "Benefits and Limitations of Tapping Into Stored Energy for Datacenters,"in Proceedings of the 38th Annual International Symposium on Computer Architecture (ISCA). ACM, 2011, pp. 341–352
[8] P. X. Gao, A. R. Curtis, B. Wong, and S. Keshav, "It's Not Easy Being Green," in Proceedings of the ACM Special Interest Group on Data Communication (SIGCOMM). ACM, 2012, pp. 211–222.
[9] S. A. Yazd, S. Venkatesan, and N. Mittal, "Boosting energy efficiency with mirrored data block replication policy and energyscheduler," SIGOPS Oper. Syst. Rev., vol. 47, no. 2, pp. 33–40, 2013.
[10] J. Cohen, B. Dolan, M. Dunlap, J. M. Hellerstein, and C. Welton, "Mad skills: new analysis practices for big data," Proc. VLDBEndow., vol. 2, no. 2, pp. 1481–1492, 2009.
[11] R. Kaushik and K. Nahrstedt, "T*: A data-centric cooling energy costs reduction approach for Big Data analytics cloud," in 2012 International Conference for High Performance Computing, Networking, Storage and Analysis (SC), 2012, pp. 1–11.
[12] MapReduce: Simpli_ed Data Processing on Large Clusters, Jeffrey Dean and Sanjay Ghemawat, jeff@google.com, sanjay@google.com, Google, Inc.

