Analysis of Raft Consensus Algorithm

T. Srajan Kumar
Department of CSE,
JNTUH/Vignan Institute of Management and Technology for Women, Hyderabad, India
teralasrajankumar@gmail.com

V. Indrani
Department of CSE,
JNTUH/Vignan Institute of Management and Technology for Women, Hyderabad, India

D. Swaroopa
Department of CSE,
JNTUH/Vignan Institute of Management and Technology for Women, Hyderabad, India

ABSTRACT
Raft Consensus is an algorithm designed as an update to paxos. It was proposed in a way such that it is more understandable than paxos by means of separation of states, but it also formally proven protected and carries some additional features. Raft approach for distributed consensus by a leader in which cluster has one and only elected leader which is fully responsible for managing log value on the other servers of the cluster. It means that the leader has privilege to decide on new entries placement and establishment of data flow between it and the other servers without consulting. Raft provides a universal way to share nodes across a cluster of computing systems, ensuring that every node in the cluster set upon the same series of transaction.

Keywords
Consensus, Data Communication, Distributed System, Paxos

1. INTRODUCTION
Raft is based upon consensus algorithm that is designed and developed to make easy to understand and its equivalent to paxos in fault-trace and performance. It is also formally proven safe and offers some additional features in cluster of nodes[1][2].

1.1 Data Communication
Data Communication is the process of transformation of data using communication technologies .Scanty technologies used in data communications are DCE [Data Communication Equipment] used at sending node and DTE [Data Terminal Equipment] used at the receiving node. Main agenda is to transfer the data and maintenance of the data during the process but here the actual information is not generated during the process[3][4].

1.2 Cloud
A network of remote servers hosted on the internet and used to store, manage, and process data in place of local servers or personal computers[1].

1.3 Paxos
Paxos is a group of protocols for synchronizing the unreliable machines. It is used for solving consensus in a network of unreliable processors[1].

1.4 Consensus
It is a general agreement among a group of participants on their results. Any number of nodes in the cluster environment can be a leader so it has some degree of set value. Consensus means several servers approves on same information[10].

Limitations
Some types of paxos algorithm exist that address this bottle neck. As it is a strictly single leader protocol. Too much traffic can drown the system

1.5 DISTRIBUTED SYSTEM
It consists of independent computers that are connected through a distributed middleware. The connected system helps in sharing different resources and services capabilities to provide users with single and multilevel coherent network systems[5][6].

Advantages
- Here it consists of multiple servers when one server failed it runs through other servers.
- As to make them easily understand they are breakdown into subprograms which can work on relatively independent.

2. RAFT CONSENSUS ALGORITHM
Raft consensus algorithm works in broadly 2 stages:

2.1 Leader Election
As a leader as authority to maintain the clusters, the heartbeat of leader is send to follower nodes .It will consider when there is time legitimate while waiting for a response in a way of heartbeats from a leader. The node changes the state in to candidate state and issues request to Remote Procedure Call[9].
It undergoes in three ways:

- By receiving the high number of vote values from the cluster nodes, the candidate node will becomes the leader. At the time goes, other servers of the new Leader get initiates by receiving the heartbeats from their leader[9].
- The candidate who participate in the leader election and didn't receive the high number of votes in the election returns to the follower state[9].
- If the other candidate's nodes receive the votes minor than the leader then they retain the candidate status through the Remote Procedure Call as rejected to the remaining cluster nodes[9].

2.2 Log Replication

The scope of client is restrict making only write requests for better understanding of beginner level audience. The log's of the leader is reproduce or exact copy to other nodes (Followers) immediately after these logs are filled with request from the clients[11]. Typically, a log access contains the following:

- The command value specified by the client to execute
- Identify the position of entry in the log of the node.
- The entry time of the command.

The current leader entries are synchronized with their logs to all other services by the leader node. Until the client replicates to the user. The client request for their entries to their leaders.

The several numbers of servers in the cluster environment successfully copies the new entries in their log's place, it is considered to be committed state. After the entry is committed state, then the leader will executes the entry and responds back with the result from the client. It should be noted that these entries are executed in the process they are received in order. So this state is called as entry committed algorithm[10].

Advantages

- The procedure of leader election is to gain the several numbers of votes within maximum of 2 terms.
- The remote procedure calls RPC to process the votes and synchronies up the cluster environment using Append Entries. So, the load does not fall on the leader node in the cluster environment.
- It is made to promote and to overcome the time and complexity from the paxos algorithm and other analogous protocol.

3. ANALYSIS OF RAFT ALGORITHM

To make the decision final it includes multiple servers

- Server has a state machine and log to get the result. And the consensus is originated from clone state machines.
- The several state machines process the same series of commands and thus produce the same series of result set[8].
- The consensus protocol failures bear countenance:

  i. Validity
  ii. Agreement
  iii. Termination
  iv. Integrity

Consensus has a multiple server system

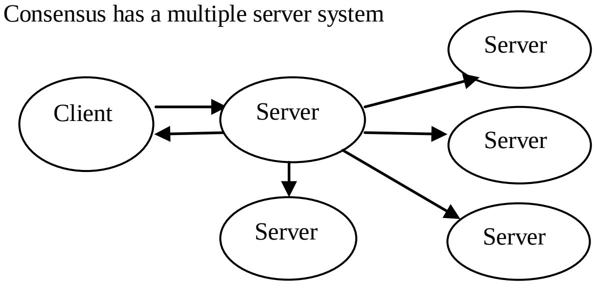

Fig (a) :Multiple Server System in Consensus

a) In the above fig. Multiple servers preserve similar data and interaction between the client and the system.
b) It utilizes the terms following as:
i. Server ii. Client

The above system shown below as in the following way:

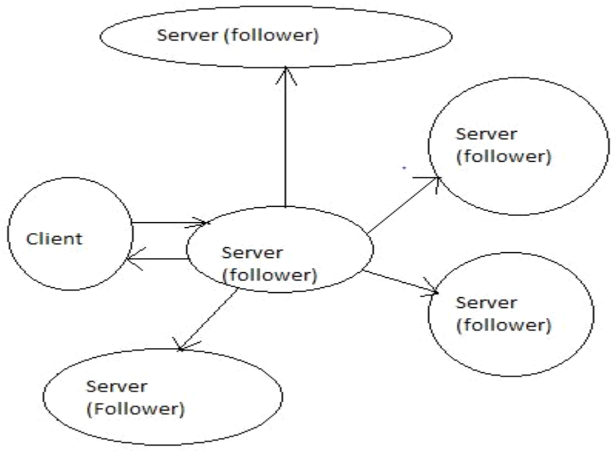

Fig (b) :.Multiple Server System Using Raft Visual

3.1 Consistency

The data cannot be varied or missed after the processing is done in the leader or follower's

3.2 Availability

It responds to every request made by the client in order to get the response.

3.3 Partition Tolerance

If the one of the server fails also it remains to be active by the other servers.

4. CONCLUSION

Paxos role in consensus in a network of unreliable processors where as Raft consensus algorithm approach for distributed consensus by a leader in which cluster has one and only elected leader which is fully responsible for managing log value on the other servers of the cluster.

REFERENCES

[1] Bolosky, W. J., Bradshaw, D., Haagens, R. B., Kusters, N. P., and LI, P .Paxos Replicated State Machines as the Basis of a High-Performance Data Store. In Proc. NSDI'11, USENIX Conference on Networked Systems Design and Implementation (2011), USENIX, pp. 141–154.

[2] Burrows, M. The Chubby lock service for loosely coupled distributed systems. In Proc. OSDI'06, Symposium on Operating Systems Design and Implementation (2006), USENIX, pp. 335–350.

[3] Camargos, L. J., schmidt, R. M., and pedone, F.Multicoordinated Paxos.

[4] In Proc. PODC'07, ACM Symposium on Principles of Distributed Computing (2007), ACM, pp. 316–317. Chandra, T. D., Griesemer, R., and Redstone, J. Poxos made live: an engineering perspective. In Proc. PODC'07, ACM Symposium on Principles of Distributed Computing (2007), ACM, pp. 398–407.

[5] Chang, F., Dean, J., Ghemawat, S., Hsieh, W. C., Wallach, D. A.,B, M., Chandra, T., Fikes, A., and Gruber, R. E. Big table: a Distributed Storage System for Structured Data. In

Proc. OSDI'06, USENIX Symposium on Operating Systems Design and Implementation (2006), USENIX, pp. 205–218.

[6] Corbett, J. C., dean, J., Epstein, M., Fikes, A., Frost, C., Furman, J. J., Ghemawat, S., Gubarev, A., Heiser, C., Hochschild, P., Hsieh, W., KKanthak, S., Kogan, E., Li, H., Lloyd, A., Melnik, S., Mwaura, D., Nagle, D., Quinlan, S., Rao, R., Rolig, L., Saito, Y., Szymaniak, M., Taylor, C., Wang, R., and Woodford, D. Spanner: Google's Globally-distributed database. In Proc. OSDI'12, USENIX Conference on Operating Systems Design and Implementation (2012), USENIX, pp. 251–264.

[7] Ghemawat, S., Gobioff, H., and Leung, S.-T. The Google file system. In Proc. SOSP'03, ACM Symposium on Operating Systems Principles (2003), ACM, pp. 29–43.

[8] Gray, C., and Cheriton, D. Leases: An efficient fault tolerant mechanism for distributed file cache consistency. In Proceedings of the 12th ACM symposium on Operating Systems Principles (1989), pp. 202–210.

[9] Lamport, L. Time, clocks, and the ordering of events in a distributed system. Commununications of the ACM 21, 7 (July 1978), 558–565.

[10] Lampson, B. W. How to build a highly available system using consensus. In Distributed Algorithms, O. Baboaglu and K. Marzullo, Eds. Springer-Verlag, 1996, pp. 1–17.

[11] Lampson, B. W. The ABCD's of Paxos. In Proc. PODC'01, ACM Symposium on Principles of Distributed Computing (2001), ACM, pp.13–13.