Ripple Labs Byzantine Consensus Algorithm

Konzer A. Cason

Student: University of Michigan- Mathematics

Author Note

Written under Dr. Cam McLeman – MTH 200: Proofs and Structures

**Abstract**

Cryptocurrency developers at Ripple Labs break down the Ripple (XRP) consensus protocol. The basic concept of a decentralized accounting system is introduced. Basics of consensus algorithms are introduced on past foundations before diving into the specifics of the Ripple Protocol Consensus Algorithm (RPCA). Specifics of agreement, consensus and utility are described as the core network principles. A deeper dive into the Byzantine Generals problem is addressed with bounds introduced for both consensus and agreement.

Keywords: Cryptocurrency, Protocol, Server, Node, Set, Probability, Byzantine Generals, Ripple

**Ripple Labs Byzantine Consensus Algorithm**

Ripple Labs set their sights on the next big cryptocurrency following the blossom and general adoption of well-known Bitcoin. In their whitepaper on the Ripple Protocol Consensus Algorithm (RPCA), Schwartz, Youngs, and Britto dive into the foundations they built upon and the specifics of their network. Following an introduction to consensus systems, a formal framework is introduced to lay the foundations of such systems. The consensus algorithm is running on a network of servers, entities running the ripple server software, which create a node based graphical network of transaction verifiers. Each confirmed transaction is submitted to a community ledger, documenting transfers of XRP, the ripple coin, on the ripple network. Two main ledgers are recorded on the network. The last closed ledger is the last know verified ledger available to all nodes, a system log of historic transactions verified by a consensus of nodes. The open ledger is the current, real-time, log of transactions yet to be, or currently being verified. Each server, or node, references a unique node list (UNL) of other nodes in the network it must double check each transaction against. A server itself becomes a proposer by broadcasting its own verifications to the public, for those nodes who’s UNL includes it to check against.

Within a network, two types of entities are defined. Non-faulty nodes, or good actors, are those servers contributing to the network while following community guidelines and only confirming verified transactions. Faulty nodes, or bad actors, are those servers with intention to manipulate the network, looking to double spend, spend one currency twice simultaneously to double it’s value, or introduce malignant transactions that remove XRP tokens from one account and deposit into another, without consent from both parties. The consensus problem is now set with three conditions: 1 – Each good actor decides in a finite time. 2 – All good actors approve/deny the same sets of transactions. 3 – Each consensus reached is binary, approved or denied. From here the problem is formalized and the RPCA can be defined.

The basis of any consensus algorithm relies on three main metrics, correctness, agreement, and utility. The correctness metric introduces the Byzantine Generals problem. This problem is defined as follows, each node is to report approval/denial for each transaction, any node may be a good/bad actor, how can we be certain a correct consensus is reached even in the presence of bad actors? The previous requirements stand, all actors will respond in finite time, all good actors agree iff all proposers send the same message, and each response is binary. The caveat is that there is a limit on the allowed bad actors, in any case a network with a majority bad actors could delude the good actors. The question now at hand, is at what threshold of bad actors can a network sustain. As an individual node, one will never be certain whether a separate node is a good/bad actor, it is left to the combined network to make this conclusion.

In conclusion,

**References**

Schwartz, D., Youngs., N Britto, A. (2014). Triangular Numbers, Gaussian Integers, and KenKen. *The College Mathematics Journal*, Pages 37 - 42.

**Figures**

Figure 1:

Figure 1.

Figure 2:

*Figure 2*:

Figure 3:

*Figure 3*: