Phase: Hybrid Consensus Blockchain Protocol For Real Time Collaboration

David Vennik
david@cybriq.systems
Varna, Bulgaria, April 2022

Abstract. Distributed systems data replication protocols have trade-offs between the three elements of the CAP theorem, and by running them in parallel and integrating them in accordance with their temporal characteristics, Phase aims to create a blockchain system that can completely replace cloud systems and beat the CAP theorem.

The missing part, fast real-time message routing, required for interactive applications such as games and collaborative working environments, is provided by changing the neglected transaction propagation system into a fast message queue with causality resolution, called a Promiscuous Ergodictic Sensor Network (PESN).

Phase combines the PESN with a Cosmos Proof of Stake IBC enabled federated fast finality blockchain and a Nakamoto Consensus based Proof of Work blockchain that acts as token issuance regulator and highly available backup to support the weak availability that the Tendermint consensus has in rare conditions.

Introduction

The final area in which Blockchain protocols are inferior to centralised cloud based server systems is one of the most important. Platforms like Facebook and Google have real-time interactivity systems as part of their offerings.

With the arrival of Cosmos Inter Blockchain Communication protocol, the bridges between Cosmos, Polkadot and Ethereum now all going live, the blockchain networks are becoming increasingly unified, but they cannot completely replace the centralised systems without interactive low latency synchronisation protocols.

The existing non-consensus peer to peer transaction distribution protocols as used in most blockchains do not attempt to do any useful thing to transactions except to collect them and spread them around. In Phase, this part of the process is changed to become a fast, concurrent, low latency, high availability, partition resistant consensus phase that eliminates the discretion of nodes in filtering transactions based on any other criteria than that they do not cause a partition in the network.

It uses the principles of ergodicity and distance optimisation via Bethe lattices to rapidly define a network topology and assigns applications to branches of the lattice to match their timing requirements. The data is accessible to applications at this level, and is checked again by a Cosmos IBC enabled chain that receives the combined global consensus from the low level, performs standard checks for double spend and validity, forms it into blocks, and then these blocks are aggregated by a Nakamoto Consensus chain which then determines the issuance of tokens for the chain and acts as a backup for fast recovery in the event of a consensus failure in the Tendermint Consensus.

Phase brings the notion of an interlocking multiple consensus distributed system, that overlaps different consensus algorithms to achieve a unified system working together to eliminate the weaknesses of each other.

Blockchains as Concurrent Application Platforms

The reality of the modern programmable blockchain as exemplified by the first of its kind, Ethereum, is that each individual application is a domain unto itself. By this, I mean that the applications largely don’t affect each other’s state except in as far as they share the common use of the primary token of the blockchain.

The very first open public blockchain protocol, that runs the Bitcoin network, simply does not run multiple applications, though its scripting engine could perform some amount of computation and has been successfully used in a similar way to the Ethereum Virtual Machine, though far more limited by the much longer block time.

The Tendermint Consensus extended to become the Cosmos blockchain protocol, with the use of the Inter Blockchain Communication protocol, provides the beginnings of a means to making each application run on a distinct but connected blockchain, in parallel and concurrently.

Both systems achieve the concurrent operation of a heterogeneous, connected network of networked applications, but make severe tradeoffs in their design that force compromises that number one victim of this compromise is real time interactivity within a given application.

Ethereum and the general family of serial global state machine blockchains sacrifice it by forcing all applications to follow one clock, and Cosmos sacrifices the unity of the system via its interconnection protocol, with a relatively expensive protocol primarily designed to allow tokens to transit across chains.

Ethereum simplifies the development environment down to a severely constrained lowest common denominator, whereas Cosmos gives great freedom of implementation but raises the time cost of the convergence of the involved chains in a cross chain transaction.

By changing the lowest, fastest parts of the protocol to be shaped according to the timing requirements of of the application, and building a network consensus map and a set of rules for allocating nodes to applications, the connectivity can be seamless as in the EVM model and extensible and concurrent like the Cosmos model.

Thus, in conclusion, the correct geometry for such a network is a radial tree structure in which various applications live on branches of the map and merge their data after converging for the application itself, thus minimising latency while keeping the network state unified, and connected to the base token ledger that secures the network and provides it with value.

Leaderless Network Topology Discovery

In the Kafka [1] distributed message queue network protocol, a complex network of nodes passing and logging messages around allows the creation of topics, a domain of interest, and users of the network can then send and receive messages to these subject areas on a publish subscribe basis with very low latency of distribution.

In order to do this, there has to be assigned leaders for each topic, to form a hub around which the messages are received, logged and delivered.

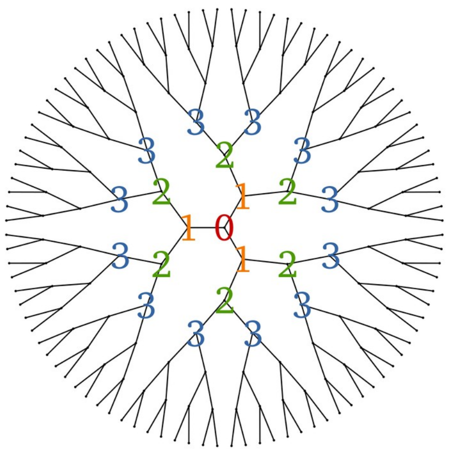
In an open public blockchain network, the step of assigning leaders to each subject area has to be done by some kind decentralised protocol that determines the leader without any node getting privilege to define this.

For the base level of the protocol, such as the common token on a blockchain network, it is possible to create a secure distribution pathway for messages by prescribing network neighbours via a Kademlia Distributed Hash Table as with the Kadcast [3] broadcast protocol - but this both increases the baseline latency via a latency topology ignorant random distribution and is only suited to the purpose of creating a robust global message distribution path, for which purpose it was created.

In the paper “Topology design for fast convergence of network consensus algorithms” [2], is described, in rather complex algebra, a mechanism by which multiple - mutually trusted - nodes can rapidly discover an optimum routing path to keep a cluster of mobile devices in communication with each other. It is hard to decipher, but the best I can make of this paper is that nodes dynamically adjust their topology in accordance with changes in the latency of their peers so as to create a central focus on the node that is in the centre of a star topology.

None of these specifically addresses the problem domain that we are concerned with in this paper, namely, the maintenance of a system involving multiple, concurrent applications that are linked together in a global token ledger that forms the currency of the network, as well as not trusting each other.

However, they all together hint at the notion of a way in which a network path topology can be discovered that achieves minimum message distribution time while allowing parallel applications to also achieve this minimum timing independently while still tying them to one or multiple levels of common data sets in order to achieve partition resistance and consistency as quickly as possible.



A Bethe Lattice [4] as seen on the right, is an optimised graph that connects a number of neighbours. It provides the geometry that we are looking for in creating a concurrent, parallel network wherein distinct branches of the network can accept minimal latency messages and form a consensus prior to propagation towards the centre, which in a blockchain network would be the same thing as the Global State.

In order to enable the creation of such a graph, and for this graph to dynamically adjust to the coming and going of nodes, as is inevitable and unavoidable on a large scale network with potentially adversarial node operators, a protocol must be created that combines redundancy with path minimization, and selects leaders indiscriminately and dynamically.

Each line on the graph has a length that represents the latency of the connection between two nodes, and each branch represents a star network and the tip of the branch in the centre is the leader of the network segment, in the way that Kafka’s architecture defines it.

Thus, such a protocol must balance this target of optimisation with creating sufficient redundancy that an adversary cannot easily poison the network, and partition it, or take control of the ledger and write arbitrary data.

The difference in the topology is that in our use case, we do not want so many branches. We would want to use Bethe lattices of higher orders, and further, in the protocol that operates on a branch connected to a node that connects to the centre, global consensus, the nodes would gossip their subjective message queues to each other completely.

Promiscuous Ergodictic Sensor Network Consensus

One of the key advantages of the Ethereum Virtual Machine model is that the runtime environment is singular and already in operation at any given time.

Cosmos’ model of sovereign chains connected by the IBC creates the problem of the necessity of further token ledgers and the latency of two chains synchronising in order to share state between the chains.

Phase takes a different approach and aims to eliminate this friction as found in Cosmos model, and the single processing thread as found in the EVM model, by making the communication between chains occur at the lowest possible level.

A sensor network is a distributed system made up of independent nodes who observe events external to the network and collaborate to determine a consensus between them of the sequence and concurrency of events observed.

The network is promiscuous, because it mixes between the various applications running on the network, which includes Cosmos zones backed up by Nakamoto Consensus archival and token issuance control network.

Dynamic Exchange Topology Instead of Reserve Currency

Unlike the “reserve currency” model of standard blockchains, with one shared currency that acts as intermediary between the tokens created for applications, all application tokens can instead form direct exchange paths between each other in a completely connected graph, focusing their timing targets to synchronise those that are most heavily in use, dynamically adjusting to the changing conditions of the marketplace as the momentum of sentiment drives activity from one place to another as opportunities for profit shift from one place to another.

For this, therefore, each distinct currency will be fundamentally controlled by the Nakamoto Consensus archival and minting chains, whose primary task is to regulate and distribute the base currency for an application. This further improves security as the more heavily a network is utilised, the more incentive there is to run these open public nodes to mine a given application currency, and thus strengthening the security of the faster, but less secure proof of stake networks they derive their data set from.

This is possible due to the promiscuity of the base network, which allows applications to have both independent ledgers, as well as data on those ledgers which comes from a consensus between two distinct ledgers, as is required to implement an exchange.

In addition to this low level order bock shared consensus between applications, at the level of the global consensus, in the centre of the Bethe Lattice that the PESN forms, there is a Cosmos Zone which then also can bridge between also multiple Phase blockchains as well as Cosmos zones and the aforementioned bridges to Ethereum and Polkadot at a lower level of latency.

It is intended thereby, that it becomes possible to take Cosmos chains and add the Nakamoto Consensus backbone and PESN message queue to bridge the chains right down to the real time event level, changing how fast the exchange between all of the Cosmos networks can hypothetically become, while forming a fast and dynamic bridge to Phase.

Further, to secure the PESN nodes, they have a bonding requirement in the form of staking to Tendermint/Cosmos validator nodes, for which they must share an operator, though the infrastructure must by its structural form be separately operated. In this way, a further extension of the rules of slashing of stake of validators covers the behaviour of PESN nodes, who mutually monitor each other’s behaviour, providing logs to the collective of each given network zone for an application of the telemetry and identity and path of transactions that can prove the mischief and give some data that can help eventually track down the cause, be it malice or error.

PESNC Protocol

The central element of the PESNC is a process of gossip based, randomised sharing of recent events and a protocol for merging these and sharing the merged versions progressively until nodes stop finding differences between each other’s logs.

The protocol functions like this:

1. PESN nodes are connected to, primarily, the central Nakamoto/Tendermint consensus backbone but also chatter with neighbouring nodes with which there is a pathway of exchange between them at the Cosmos IBC level
2. Events of interest to the application, both exchange events and transactions of the application’s currency, are received strictly directly from users whose clients

know which nodes to send them to for the most immediate response.

3. Each node when it receives a new event (transaction) gossips it immediately to its nearest neighbours, who then append this event to their log of recent events in a consensus application epoch. The epoch is bounded in its size by the latency requirements of the application, which can be as low as 100ms.
4. Once the nodes observe that an epoch has passed, they then gossip their version of the epoch's sequence of events, to all of their peers. For this reason, the size of the application sensor network is confined to a group of peers whose connection latency permits the reaching of the application required latency target for convergence.
5. Each node then combines the peer copies of event sequence together, seeking to place each event closest to the beginning of the epoch where two events differ in sequence,
6. Once all of these events are zipped together and older versions eliminated, the merged versions are then gossiped again, and when events are found to partition between two parts of the sensor network (cell), they are designated as concurrent events and encoded into a Directed Acyclic Graph and passed to the leader (central node in the cluster by latency) and passed to the Cosmos Zone where the validators then perform the vote to finalise the transaction set.

This process proceeds also between the nodes of different networks as application token exchange books are synchronised between ledgers, for this shared data, which is part of the transaction data being certified within each PESN cell or application.

PESNC Synchrony

Not only synchrony is required, trust is required also in order to bypass checking requirements for a sensor network to function at minimum latency. Trust is thereby protected as mentioned previously, by bonding with the Tendermint validator that operates a PESN node, and in addition to this, nodes must have a very tight time differential between them, and falling out of sync should be a slashable offence, as nodes must be able to keep in rhythm with each other at the level of the application latency requirements, which will start somewhere around 100ms.

To achieve this, the node those latency distance is optimal between the operating PESN nodes

Bibliography

Not so much a bibliography but a listing of the materials that form the basis of Phase.

1. Kafka Whitepaper - principles of a low latency complex message distribution network protocol.
2. Topology design for fast convergence of network consensus algorithms - a protocol for leaderless discovery of network topology.

3. Kadcast - a Kademlia Distributed Hash Table based message propagation protocol for partition resistant (unreliable) message broadcast.
4. Bethe Lattice - A geometry that enables a minimum number of paths from edge to centre as found in the physics of magnetism.