

A formal definition and statistical model checking of the constellation blockchain

Wyatt Meldman-Floch

February 17 2018

Abstract

Introduction

Fundamental Data Structures, Types and Functions

Tier <: Numeric

Numeric type that is used to define ordering among Radials (see below)

BlockData[Tier] <: U

Consider a type *BlockData* which serves as the parent type (under universal type U) of all data going on chain. All transactions, validator requests, meme data etc. All BlockData is equivalent via covariance and thus all BlockData can be compressed into a Block.

Block[U] <: U

Contains compressed form of Block Data. This is the result of consensus. Each unit of *BlockData* must reference the previous round of consensus, this is

satisfied by containing the *Block* hash. It is worth noting that $Block[U] \equiv Block[BlockData[Tier]]$ through covariance. $Block[U]$ is a Fix Point Type ¹

Transaction <: BlockData[0]

Transaction is a subtype of BlockData that can only exist on the bottom tier. It is the building block of our currency.

LeftHand <: RightHand <: Transaction

Note: LeftHand and RightHand are symmetric subtypes of Transaction. For a Transaction to be valid it must be 'signed' by the counterparty. This is satisfied by sending a RightHand transaction that references a LeftHand transaction.

Definition: Consensus Simplex, σ

A set of validators, undergoing cryptographic consensus to produce a *Block*. Formally defined as a Simplex.

$$\sigma \cong collection[node] \tag{1}$$

Definition: Validator n

A node that has 'woken up' by starting a *consensus* process. It now gossips transactions to it's neighbors and is waiting to be selected as a Delegate.

$$n \in \sigma \tag{2}$$

Definition: Delegate

A node who has been selected to perform consensus. Delegates are implicitly chosen, locally on each node via the Generating Function (see below). Once a new *Block* is received, it is passed to the Generator Function, which tells this node if it is a delegate. A delegate is a node $n \in d$ such that

¹<https://jto.github.io/articles/typelevel-fix/>

$$d \subset \sigma \mid g(Block, d) \rightarrow d \quad (3)$$

where g is the generating function below.

Generating Function g

This function is used to determine the next set of delegates from a given set of validators (consensus cluster) by selectively sampling a subset of nodes, based on reputation score and a probability distribution. See GURU for examples selective sampling via probability distribution.

$$g : (Block, \sigma) \rightarrow \phi \mid \phi \subseteq \sigma \quad (4)$$

Consensus Function $c : \sigma_k \rightarrow Block_k$

A function that maps a σ_k to $Block_k$. It follows that this is isomorphic to a catamorphism, with the collection being a simplex's mempool and the result the reduce being a Block.

Node ADT

let n be an abstract data type Node, with three attributes, a *protocol* process a *consensus* process and a *chain* class. The *protocol* process handles transaction signing and essentially all functionality to send and receive payments. The *consensus* process which implements the responsibilities of a validator node, is optional. Nodes can be 'sleepy', turning on the *consensus* process at will. The *chain* class is a local blockchain made up of all transactions/interactions with the chain that this node has made. Each link in the chain is a sub type of BlockData.

Formally we define n as the algebraic data type

```
case object Node {
  val chain: Chain = new Chain()
  val protocol: => Seq[BlockData[T]] = new Protocol()
  val consensus: => Block[U] = new Consensus()
}
```

Network Topology

We define the primitives of our network topology. We come to the conclusion that our network topology can be formally defined as a simplicial complex, specifically an ordered set of a special simplex called a Radial, which we define below. We define constellation's topology as an ordered set R_K of Radials R_t , following ordering defined by Tier $t \in T$

$$R_K : \{R_0 \dots R_T\} \quad (5)$$

Simplex

The clique complex of a graph G is a simplicial complex whose simplices are the cliques of G ². We define a simplex σ as a completely connected graph (clique) of constellation nodes, with which we can perform cryptographic consensus. Given $\sigma = \{n_0 \dots n_i\}$, σ is a simplex iff:

1: σ is a complete graph such that for a set of edges e , corresponding to nodes n ,

$$\forall n \in \sigma, \exists e \in E \mid e \equiv \sigma \setminus \{n\} \quad (6)$$

Corollary: for each n in a simplex σ_K , the simplicial star³ of n , $st_K(n)$, for simplex K is equivalent to the σ_K that is:

$$\forall n \in \sigma_K, st_K(n) \equiv \sigma_K \quad (7)$$

2: There exists a deterministic mapping given by an immutable generating function g

$$\exists g : \sigma \rightarrow d, \mid d \subseteq \sigma \quad (8)$$

3: There exists the notion of a star cluster⁴ Such that star cluster σ forms an independence complex of K .

$$SC(\sigma_k) = \bigcup_k st(n) \in I_G \quad \forall k \in K \quad (9)$$

Where I_G is the set of all independence graphs of K ⁵. It follows from corollary (7) that

²<https://arxiv.org/pdf/1007.0418.pdf>

³Closure, star, and link: [wikipedia/Simplicialcomplex](https://en.wikipedia.org/wiki/Simplicial_complex)

⁴<https://arxiv.org/pdf/1007.0418.pdf>

⁵Definition 2.1 <https://arxiv.org/pdf/1007.0418.pdf>

$$\begin{aligned}
SC(\sigma_k) &= \bigcup_k st(n) \\
&\equiv st(n) \forall n \in k
\end{aligned} \tag{10}$$

Radials: Tiered Hypergraphs

A hypergraph represents an arbitrary set of subsets of it a graph's vertex set, where each subset is called a hyperedge⁶. We define the the Radial abstract data type in terms of a hypergraph and mappings between vertex sets within that hypergraph. Specifically for given Tier $t \in \mathbb{N}$ (see above), we define a Radial ADT R_t as a functor⁷ over the category of simplexes such that

```

case class Sigma(nodes: Node*)

type Radial[T <: Tier] {
  /*
  simplex and starCluster are hyperedges
  */
  val starCluster: Sigma[T-1] \\ we need a stricter definition here
  val simplices: Sigma[T]*
  def hyperPlane[T1, T2]: Sigma[T1] => Sigma[T2]
  def route[T]: hyperPlane[T, T-1]
}

```

The hypergraph for all simplexes σ_k within Tier t is given by the two element set

$$H_{t,k} = \{I_{g_{t-1}} \mid g_{t-1} \subset G_{t-1}, \sigma_k\} \tag{11}$$

where $I_{g_{t-1}}$ is a subset of all simplexes in tier $t-1$ (as we know all simplexes are independent) and $\sigma_k \in R_t.\text{simplices}$. *hyperPlane* is a function that maps between two hyperedges, potentially across tiers. *route* is a function that connects our two hyperedges with a *hyperPlane* between Tier t injectively to an independence complex in $t-1$. Note that $Sigma[T-1]$ and $I_{g_{t-1}}$ are equivalent via homotopy as shown in Definition 2.1 (footnote 5).

In the degenerate case of our definition of R_T , namely when $R_t = \{R_0\}$, *starCluster* is a mempool as defined below, but of transactions which are isomorphic to all subtypes of *BlockData* via covariance.

⁶Pal S. et. al. <http://www.facweb.iitkgp.ernet.in/~spp/geomgraph.pdf>

⁷<https://typelevel.org/cats/typeclasses/functor.html>

Chain Topology

We define the primitives of our DAG chain. We come to the conclusion that our DAG chain is a directed acyclic graph whose direction is given by simplex graph⁸ κ and topological ordering follows the tiered ordering of R_K . The simplex graph $\kappa(G)$ of an undirected graph G is itself a graph, with one node for each clique (a set of mutually adjacent vertices) in G . As all of our simplexes are independent, $\kappa_j \mid j \subseteq t - 1$ for tier t is formed as a disjoint subset of simplexes in $t - 1$. We define κ as a mapping from an undirected graph G to a new graph $\kappa(G)$ whose vertices are cliques and compositions of cliques of G

$$\kappa : \sigma_j \rightarrow \sigma_k \mid j \subset t - 1, \sigma_k \in R_{t-1}.simplices \quad (12)$$

Remark: there is notable symmetry in our definition of κ and *Radial.route* above. We will see a duality formed from this connection through consensus.

Chain Fibering

We formulate notion of chain fibers by defining consensus in terms of a simplex graph κ . A chain fiber is given by

$$f \circ \kappa_j^k : \sigma_k \rightarrow Block_k \quad (13)$$

where κ_j^k is a simplex graph made up of chain fibers $Block_j$ in a preceding tier. The chain fibers from the preceding tier, $Block_j$, become the domain of the consensus function for tier k . We can say equivalently

$$c_k : \{Block \dots Block\}_j \rightarrow Block_k \quad (14)$$

We can formally define the mempool *mem* of simplex σ_k in tier t as

$$mem_k \equiv \{Block \dots Block\}_j \mid j \subset R_t.simplices \quad (15)$$

Consensus, hyperplanes and delegate selection

Delegate selection is performed by the Generating function, a deterministic function. Consensus is a probabilistic function. Both are defined by the hyperplane function defined on radials. Specifically we use the framework above to formally define the Generating function and Consensus

⁸wikipedia/Simplexgraph

The Generating function is an instance of *Radial.hyperPlane* where $T1 = T2 = T$. Formally

$$\begin{aligned} g : (Block, \sigma) &\rightarrow \phi \mid \phi \subseteq \sigma \\ g &\cong Radial.hyperPlane[T, T] \end{aligned} \quad (16)$$

g is implemented isomorphic to

$$\begin{aligned} g &\cong Radial.simplices \\ .fold(\sigma \Rightarrow z.update(\sigma))(z : Histogram) \\ .flatMap(\sigma * \Rightarrow dist(\sigma *)) \end{aligned} \quad (17)$$

where *dist* is the generating function of a probability distribution given by precompiled bytecode and notarized by the TGE Block.

Consensus is an instance of *Radial.hyperPlane* $[T - 1, T]$. Formally

$$\begin{aligned} c_k : \{Block \dots Block\}_j &\rightarrow Block_k \\ &\cong Radial.simplices \\ .filter(\sigma_k \Rightarrow Radial.route[t, j](\sigma_k)) \end{aligned} \quad (18)$$

where *Radial.route* $[t, j]$ is isomorphically defined as

$$Radial.route[t, j](\sigma_k) \cong \exists \kappa_j^k \quad (19)$$

State Transition

We define a DFA for a constellation node. Given the regular definition

$$D = (S, \Sigma, \tau, s_0, F) \quad (20)$$

Our set of possible states is the three element set

$$S = \{online, offline, validating\} \quad (21)$$

Our working alphabet Σ is loosely defined by the RPC interface to our node, which we will define as a receive loop over states in S yielding $\Sigma : RPC[S]$. Our state transition function

$$\tau : S \times \Sigma \rightarrow S \quad (22)$$

Is the *Receive* loop to an akka actor. Finally, our initial state s_0 is offline and our final state

$$F = \{offline\} \quad (23)$$

Transition Validator

A constellation node that wishes to "mine" coins may do so by opting to become a validator. This state transition from online to validator τ_v is defined as follows: given a $z \in \Sigma^*$ that represents the state transition τ_v we have

$$\tau_v = s_{online} \times z \rightarrow s_{validator} \quad (24)$$

Upon state change the *consensus* process, defined by the FSM below, begins.

Consensus FSM

In terms of traditional DFA used above we define our *consensus* process as the following finite automata

$$\begin{aligned} S &= \{Gossiper, Delegate, Offline\} \\ \Sigma &: FSM[S] \\ \tau &: Receive[S] \\ s_0 &= Gossiper \\ F &= \{Offline\} \end{aligned} \quad (25)$$

where *FSM* refers to the akka FSM module⁹.

Gossiper

While a node is a gossip it's goal is to spread awareness of each transaction across it's simplex. All nodes in the Gossiper state are waiting to become delegates. It is patiently awaiting blocks from delegates, as the contents of each new block tell the Gossiper whether or not it is to become a Delegate.

⁹<https://doc.akka.io/docs/akka/2.5.5/scala/fsm.html>

Delegate

The state transition to a Delegate is given as follows. Criteria for a state transition is the existence of this node in the selected delegates as given by the generating function g above.

$$\tau_d : s_{Gossip} \times g(Block, \sigma) \rightarrow s_{Delegate} \quad (26)$$

Proof of Meme

Delegate selection is the cornerstone to minting, we define delegate selection as a selective sampling¹⁰ method given by the generating function g . In order to formally define proof of meme, we need to define a metric for "memeness" and the metric space upon which it can be calculated. First we define the space of hyperplanes between Radials and show that this is a metric space. Every metric space is also a topological space, so it serves us to first define R_K as a topological space. It is trivial to show that $n \in k\forall K$, σ , $Radial.simplices$ satisfied the necessary axioms¹¹ of a topological space and just as trivial to extend this to the union of simplices across Radials. We leave this as an exercise for the reader.

The topological space we are interested in is actually a mixture of $n \in k\forall K$, $\sigma_k \in K$ and $R \in R_K$. Because σ_k is an independence complex for all $k \in K$, we need to use the concept of homotopy to "stitch" these topologies together. Formally we will show that the recursion scheme of a Hylomorphism and it's parameterization via algebra and coalgebra allow us to "stitch" these topologies together under homotopy defined by typesafety across state transitions.

From the beginning we have assumed homotopy equivalence, thus it follows up to isomorphism that if one Radial is a topological space, so is R_K

$$\exists R \in R_K \mid R \cong T \implies \forall R \in R_K, R \cong T \quad (27)$$

where T is a topological space. Since we know that R is a functor over the category of topological spaces it follows that R_K constructs a chain complex under type equivalence¹²

$$C_R := R_0 \xrightarrow{h_0} R_1 \dots R_{n-1} \xrightarrow{h_{n-1}} R_n \quad (28)$$

where h_0 is a hylomorphism.

¹⁰<http://dissertation.laerd.com/purposive-sampling.php>

¹¹<http://mathworld.wolfram.com/TopologicalSpace.html>

¹²

A hylomorphism can be defined in terms of algebras and coalgebras:

```
def hyl[F[_] : Functor, A, B] (f: F[B] => B) (g: A => F[A]): A => B =
  a => f(g(a) map hyl(f)(g))
```

Where g is an f -coalgebra and f is an f -algebra ¹³. It is trivial to note that process of distributed consensus is isomorphic to a hylomorphism.

Hylomorphic Vector Space

Cohomology is obtained if we reverse the arrows in a C_R . We are concerned Cohomology because we do not have the notion of a tensor product in the homology theory of homotopy types. However the cohomology of a wedge product between two spaces is isomorphic to the product of the cohomologies. ¹⁴ and we will make use of this and the fact that a metamorphism symmetrically defined with the same bialgebras as our hylomorphism formally define a vector space.

$$H_n(X \wedge Y) \cong H_n(X) \times H_n(Y) \quad (29)$$

The product is a manifold from the wedge of spaces with isomorphic cohomology groups. As our metamorphism is a bialgebra it follows that a vector space can be formed¹⁵. Naturally, we can create an inner product space of a homomorphic chain complex as follows

$$H_n(X \wedge X) \cong H_n(X) \times H_n(X) = \langle X, X^* \rangle \quad (30)$$

This vector space can be formally defined in terms of the hyperplanes between Radials, and furthermore referred to as "hylomorphic space". A hylomorphic space $H_n(X) \times H_n(X)$ is essentially an inner product space upon which we can define a probability functional¹⁶.

Memetic metrics: dap functional

We define our metric *dap* as an ordinary metric on the space of a hyperplane between Radials.

¹³<http://free.cofree.io/2017/11/13/recursion/>

¹⁴R. Cavallo, Theorem 4.6 <https://www.cs.cmu.edu/~rwh/theses/cavallo.pdf>

¹⁵<http://www.cs.ox.ac.uk/jeremy.gibbons/publications/metamorphisms-scp.pdf>

¹⁶

$$d : Block \rightarrow \quad (31)$$

TODO show how this is isomorphic to a $A \times B$ where A,B are the basis of the vector space.

We can now define a distance metric d that satisfies all the necessary axioms (replace with acceptance criteria).

$$\begin{aligned} d(n1, n2) &\geq 0 \\ d(x, z) &\leq d(x, y) + d(y, z) \\ d(x, y) &= d(y, x) \\ d(x, z) - d(z, y) &= 0 \iff x = y = z \end{aligned} \quad (32)$$

Hylochain: Infinite scalability through homomorphic parachains

We show that the criteria for cross-chain liquidity allows us to recursively define an unbounded blockchain within the homology theory of homotopy types.

Our definition of the hylomorphic space implicitly showed that R_K is a homology class within the homology theory of Homotopy types¹⁷. It follows that R_K forms a chain complex under homotopy. A homology theory of K_i types, where $H_n(-)$ is a functor by ¹⁸, is given by

$$H_n(X) = ||colim_i(X_i, \theta_i)||_0 \quad (33)$$

It follows from our algebraic definition of a hylomorphism that we can construct a homology

$$H_n^h(X) = ||colim_i(F_i, h_i)||_0 \mid h_i : A \rightarrow B = g \circ f \quad (34)$$

where h is a hylomorphism.

It follows that R_K is isomorphic to H_n^h . Proof: ¹⁹

¹⁷<https://arxiv.org/pdf/1706.01540.pdf>

¹⁸R. Grahm Theorem 34 <https://arxiv.org/pdf/1706.01540.pdf>

¹⁹ensure above definition satisfies criteria from R. Grahm

Typesafe Cross-chain Atomic Swaps

Definition of chain liquidity

Two chain complexes are chain equivalent if there exists a homotopic mapping between them. We show how homotopy is constructed for a chain complex.

Proof: we know due to the univalence axiom that homotopy is implicit from type equivalence. We know from R. Graham that a function from $||\text{colim}_i(X_i, \theta_i)||_0 \rightarrow ||\text{colim}_i(X_i, \phi_i)||_0$ where each X_i, Y_i is a set, it suffices to show that $f_i : X_i \rightarrow Y_i$ such that $f_i \circ \phi_i = \theta_i \circ f_{i+1}$.

Thus we show that for two chain complexes with type equivalent functors, if their f-algebras are isomorphic, it follows that ²⁰

$$f_i \circ \phi_i = \theta_i \circ f_{i+1} = \theta_i \circ f_i = \theta_i \circ \phi_i \quad (35)$$

A tensor defines chain mappings²¹ and a path differential of 0 is required for homotopy. We do not have additivity which is required for a tensor, but we can get around this via the Exactness Axiom. We know that the path differential is 0 under type equivalence due to the univalence axiom, thus there exists homotopy.

Liquidity between multiple chains is isomorphic to the existence of a fibration, which is defined as $C : S \rightarrow \text{Type}$ where C is a base type, and S is a pointed set. This is analogous to classical topology where C and S are spaces. Due to the behavior of truncated colimits of sets²² it can be shown that for $||\text{colim}_i(Y_i, \theta_i)||_0 \rightarrow ||\text{colim}_i(Z_i, \zeta_i)||_0$ given a $g_i : Y_i \rightarrow Z_i$ then $||\text{colim}_i(X_i, \theta_i)||_0 \rightarrow ||\text{colim}_i(Z_i, \zeta_i)||_0$ is given by $f \circ g$. If f, g are isomorphic then there exists a homotopy and there exists a fibration. It follows that more complicated cross chain structures can be formed by mapping cones and mapping cylinders.

Scaling through Recursive Parachains

We show that our definition of C_K is unbounded using the formulation of a chain-complex of parachains. The criteria for forming a smooth chain complex are satisfied by C_K which implies that the chain complex is unbounded and our network bandwidth follows the following exponential formula:

²⁰<https://www.seas.upenn.edu/~jean/sheaves-cohomology.pdf>

²¹<http://www.math.uni-frankfurt.de/~johannso/SkriptAll/SkriptTopAlg/Skript-TopChain/algtop4.pdf>

²²R. Graham Remark33

$$\begin{aligned} f(x, n) &\cong \alpha x \mod n \\ U(f, x, t) &= (cf(x, n))^t \end{aligned} \tag{36}$$

where c, α, n are scaling params. TODO, tie these together in our definition of Radial.

Sound Bytes

Consensus

Consensus is the process of forming a *Block*, it can be thought of as a function $f : \sigma \rightarrow \text{Block}$. Consensus clusters are formed in a tiered hierarchy. The top most tier forms the global state of the chain, which is made up of *Blocks* from preceding tiers. Each tier, from top down until the second to last, has responsibility for routing transactions and validating the blocks from consensus clusters.

What is the lifecycle of a transaction

When a transaction is sent, it is sent from a node to a higher tier which 'routes' the transaction to the consensus cluster(s) that host its 'shard', or the shard of the blockchain that host's its public key's history. Each transaction has a left and right half. The initiator of the transaction sends the *LeftHalf* to the network. Its *LeftHalf* is then referenced by the *RightHalf* which is sent by the counterparty.

Why do we double sign?

It preserves the notion of ordering. Scenario: I have 5 dollars, I send to two people. I send 5 to Wyatt and then 5 to Preston. I will need to wait until the top tier finishes consensus because both transactions are effectively 'racing' each other. Double signing allows us to preserve ordering (with high probability) without waiting for the total network to update state. Without this, consumer facing point of sale systems (think grocery store) are not possible; we would need to wait for a global state to reach consensus. Double signing gives users the illusion of instant transaction confirmation.

How is consensus performed

Consensus $f : \sigma \rightarrow Block$ is the act of creating notarized data in the form of *Blocks*. In our case, we are using the HoneyBadgerBFT, which prevents against sybil attacks using encryption ²³. Nodes are rewarded based upon successful completion of consensus, the number of transactions they provide, and metadata about their performance. This is all calculated post facto by proof of meme and rewards are given within a set interval of blocks.

How are delegates selected

Delegates are selected locally, by passing the previous block and current set of validators into our Generating Function. This happens within the Consensus FSM.

How is this secure/fit in with our incentive model

Double spends across asynchronous consensus is prevented by double signing transactions. Consensus clusters are rewarded for processing transactions and sybil attacks are mitigated via encryption in honeybadgerBFT. We also are able theoretically improve typical byzantine fault tolerance over 40% thanks to GURU and our reputation model. Ddos attacks can be mitigated via throttling of accounts with low reputation scores. I propose an incentive for routing where each node that routes a transaction signs the tx, and when a tx is notarized each account that routed the tx is given a reputation increase.

²³ch. 4.3 <https://eprint.iacr.org/2016/199.pdf>