Technical Paper

PHANTOM METCHAIN(GHOSTDAG)

A Scalable Generalization of Nakamoto Consensus

Abstract:

The foundation for distributed ledgers based on blockchains was created by Satoshi Nakamoto in 2008. An open, anonymous network of nodes, or miners, which work together to maintain a public ledger of transactions, is the basis of this system. The ledger is a chain of blocks called a blockchain, and each block contains a collection of fresh transactions gathered from users. The blockchain created by Satoshi has a significant scalability issue.

All trustworthy nodes must be aware of each other's blocks relatively soon after they are created in order for Satoshi's longest chain rule, more often referred to as the Bitcoin protocol, to function securely. In order to achieve this, the system's throughput is artificially suppressed such that each block propagates completely before the creation of the next.

To distinguish between blocks mined correctly by honest nodes and those produced by non-cooperating nodes who opted to stray from the mining protocol, PHANTOM solves an optimization problem over the blockDAG. By making this distinction, PHANTOM offers a solid total order on the blockDAG that is ultimately accepted by all trustworthy nodes. In order to avoid the exorbitant computation required to implement PHANTOM, we created the effective greedy algorithm GHOSTDAG, which perfectly encapsulates the design of PHANTOM.

We offer a formal demonstration of GHOSTDAG's security, showing that its block ordering is irreversible up to an exponentially inconsequential factor with interoperability. We go through GHOSTDAG's characteristics and how it stacks up against other DAG-based protocols. GHOST- DAG under actual world circumstances. We analyze confirmation times gleaned from watching the main network.

Introduction:

Blocks must spread quickly to all network miners in order for the Bitcoin system to be secure. The need that each block to carry a proof-of-work slows down block generation itself. Block propagation must be quicker than the normal time it takes the network as a whole to construct the next block for the Bitcoin protocol to be secure. The protocol for Bitcoin restricts block creation to taking place only once every 10 minutes in order to guarantee this property. Additionally, the block size is constrained to allow for quick transmission.

In this paper we present PHANTOM METCHAIN, a protocol that generalizes Nakamoto's longest-chain protocol. In Bitcoin, blocks reference a single predecessor in the chain, hence forming a tree, in contrast, PHANTOM blocks reference multiple predecessors, thus forming a Directed Acyclic Graph, a blockDAG. Each block can thus include several hash references to predecessors. PHANTOM then provides a total ordering over all blocks and transactions and outputs a consistent set of accepted transactions. Unlike the Bitcoin protocol, where blocks that are not on the main chain are discarded, PHANTOM incorporates all blocks in the blockDAG into the ledger, but all keeping in count the attack 51%, PHANTOM METCHAIN has an added split hash rate algorithm.

In rough terms, PHANTOM METCHAIN consists of a four-step procedure. This method, which is the core of the protocol, is used to exclude blocks made by misbehaving nodes and recognizes a set of well-connected blocks using the blockDAG's structure (later referred to as blue blocks) Blocks that are either withheld by their creator for a while for reference only older blocks from the DAG will almost certainly not be included in the set of blue blocks.

Our method encourages blocks inside the chosen cluster and penalizes those outside of it in order to convert the DAG's naturally occurring partial order into a full topological one (i.e., an order that respects the topology). Transactions within a block are arranged according to the order in which they appear in the block as a result of the order over blocks. We go over each transaction in this order iteratively and approve any that are compatible (in terms of the underlying consistency idea) with the previous transactions.

-Hashrate split algorithm which splits the hashrate to 3 block types.

-Mini MetBlock (Interval of 10 Seconds) 6 blocks per minute.

-Mega Metblock (Interval of 2 Minutes)

-MET Metblock (Interval of 10 Minutes)

Mini Metblocks generated are reconfirmed after every 2 minutes to form a Mega Metblock which helps METCHAIN to attain more security over other Proof-Of-Work chains and helps to avoid 51% attacks. As the mining hash rate will be automatically split as per the block time to avoid any false blockchain from being created. Same is followed by MET Metblock.

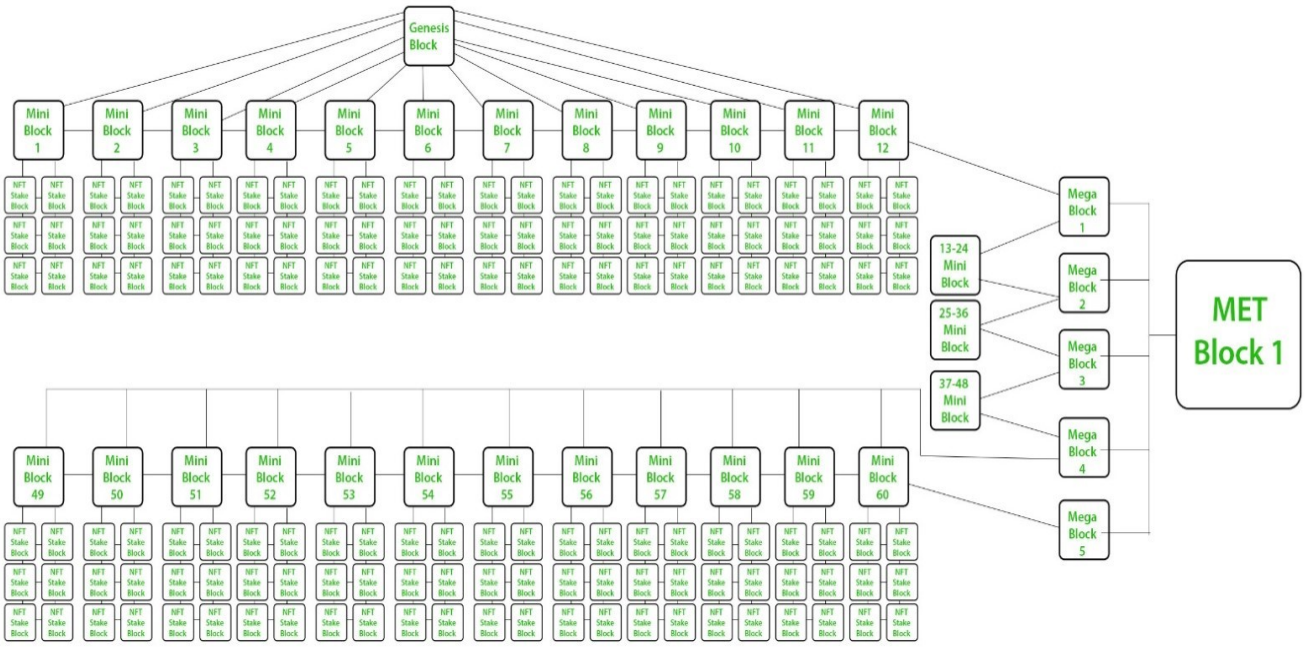

Figure 1: shows Metchain' block topology.

The PHANTOM protocol assumes a malicious mining coalition is not the majority, i.e. an attacker has <50% mining power. This puts the asymptotic maximum-security threshold of the system at 1/2, similar to that of GHOST and SPECTRE. The PHANTOM protocol differs from SPECTRE in that it enforces strict ordering over blocks and thus transactions in the system. PHANTOM is usable for smart contract systems in this light but only at the cost of forcing a proper ordering over blocks. The tradeoff can be quantified in the time that it takes nodes in the PHANTOM protocol to reach consensus, which although scalable is not as fast as a protocol like SPECTRE which can run without these guarantees.

The mining protocol utilized by PHANTOM follows a similar Proof of Work system as Bitcoin, where the computational puzzles are finding hashes under a target difficulty. At each step in an individual node's mining process, it examines its view of the blockDAG network and does the following:

- Finds the set of all blocks with 0 in-degree, denoted B. Computes hashes until it finds a hash h<D.
- Creates a block b with hash h, includes B in the header (directed edges to those blocks), and broadcasts b."

Purely topological methods are used by PHANTOM to reach consensus. In contrast to SPECTRE's voting system, the PHANTOM protocol selects a "correct blockchain" within the blockDAG as it assembles the total number of legitimate blocks. Recursively locating this chain forces the overall ordering of the transactions contained within it. Using a greedy approximation algorithm to solve an optimization problem that will be described below, PHANTOM adds these blocks.

Finding the maximum m-cluster subDAG, as shown in Figure 1, is the first step in the process of identifying the finest, honest blocks. The official issue is described below. Due to the fact that it includes several trade-offs and the fact that the real network delay is unknown, the task of choosing the best parameter m also poses an intriguing problem. To begin with, the parameter has a direct relationship with the anticipated propagation delay of the complete network. The fact that this delay is bounded but not explicitly known is due to the partially synchronous model that we work under.

The Phantom Protocol:

In PHANTOM, a miner refers to all blocks in tips(G), where G is the DAG that the miner observes locally at the time the new block is created, rather than extending a single chain. The miner should publish its new block as soon as it is ready. Together, these two guidelines make up PHANTOM's DAG mining protocol.

The aforementioned DAG mining algorithm specifically indicates that both blocks are added to the blockDAG and used as references by all (honest) miners even if two blocks contain conflicting transactions. The main issue is therefore how to restore the blockDAG's consistency. In our system, this is accomplished by making sure all 12 mini blocks which are set 10 seconds apart after reaching the mature difficulty point. Each mini block supports layer 2 block chain which creates a staking block every 1.67 Seconds reducing the transaction time and also increasing the security. These staking blocks point to Mini Block consisting of block reward of 0.3 MET and these then point to the subsequent Mega Block which is created every 2 minutes with block reward of 3 MET and also points to the same previous hash. This is then further verified while creating a MET block. All the previously created Mega and Mini blocks in 10 minutes should point to the previous MET Block and Subsequent MEGA block as well. Approving those who came before them. By achieving consensus on the sequence of blocks, PHANTOM ensures that there is also agreement on the set of allowed transactions.

In essence, Bitcoin may be viewed as an ordering protocol as well, with the longest chain of blocks' worth of transactions coming before those with the shortest chain. However, the protocol underlying Bitcoin is only known to be secure under low block rates.

Unfortunately, the security analysis of existing blockchains is not as general as ours (e.g., their attacker does not take advantage of providing consulting information to different honest parties), while the analysis of METCHAIN does not carry to the setting of GHOST. This is because the GHOST rule is a natural, albeit radical, reformulation of how each miner determines the main chain. In GHOST, miners adopt blocks in the structure of a tree. Note that in both Bitcoin and GHOST one can consider parties collecting all mined blocks in a tree data structure. However, while in Bitcoin the miners would choose the most difficult chain as the main chain, in GHOST, they will determine the chain by greedily following the heaviest observed subtree. This means that for the same subtree, a Bitcoin miner and a GHOST miner may choose a completely different main chain. Furthermore, it means that the difficulty of the main chain of honest parties does not necessarily increase monotonically (it may decrease at times) and thus a fundamental argument (namely that blockchain monotonically increases) that made the analysis of our possible, does not hold anymore.

Results:

In contrast to METCHAIN, we propose a new analysis framework for blockchain protocols that focuses on trees of blocks. Due to this system, it can debate random variables on the participant-created block trees. We may describe ideas like a node being d-dominant in our framework, which indicates that the block corresponding to that node would be favored over other sibling nodes by a margin of d based on a particular weight measure. In fact, by demonstrating that Bitcoin and GHOST adhere to the same rule but just for a different weight measure, allows us to unify the description of both.

We then offer the first formal security proof of the GHOST rule for blockchain systems using our framework. In particular, it is demonstrated that GHOST is a reliable transaction ledger that meets liveness and persistence requirements. The new methodology, which is referred to as the fresh block lemma, which condenses the features of the resilient transaction ledger into a single lemma, allows us to obtain this result.

We use the abstraction suggested in METCHAIN for our model. Particularly, synchronous communication is assumed in their environment, known as the q-bounded environment, and each party is permitted q queries to a random oracle. The network includes an anonymous message diffusion mechanism that ensures that each round's messages are delivered by all sincere parties. At the start of the following round, all messages are delivered. It should be noted that the diffusion method is unreliable, meaning that the attacker may deliver messages to only some of the network participants. Additionally, the enemy rushes and adapts. Rushing in this situation allows him to view every honest player's message before choosing his own course of action. Furthermore, they have total control over the sequence in which messages are sent to each player. In terms of computational power, the model assumes that all honest parties have the same amount, whereas the adversary's computational power is inversely correlated with the number of participants it controls.

Since there are n parties altogether, it is presumed that the adversary has power over t of them (honest parties are unaware of any of these conditions). Finding a hash value smaller than a difficulty parameter D results in the discovery of a new block. The likelihood that a single hashing query will result in a solution is given by the formula $p=D^2$, where p is the hash length. The adversary's total hashing power is equal to pqt, the honest players' total hashing power is f = pqt, and the sum of all three is pqt. The following list includes a number of definitions that will be used frequently.

1. A round is referred to as: Successful if in this round at least one honest participant computes a solution. If precisely one honest participant computes a solution in this round, it will be considered singularly successful.
2. In execution, blocks are called:

   -honest, if mined by an honest party.

   -adversarial, if mined by the adversary.

1. Some chain notations: By C dk we denote the chain that results by dropping the last k blocks of C. We will say that a chain C 0 extends another chain C if a non-empty prex of C 0 is a su-x of C.

In METCHAIN, a lower bound to the probabilities of two events, that a round is successful or that is uniquely successful (dented above), was established and denoted by γu = α−α 2. While this bound is sufficient for the setting of small f, here we will need to use a better lower bound to the probability of those events, denoted by γ, and with a value approximately αe−α

THE GHOST BACKBONE PROTOCOL:

The term "Backbone Protocol" was first used to investigate the characteristics of the fundamental Bitcoin protocol in METCHAIN. At this level of abstraction, we are only concerned with the characteristics of the blockchain, not the information contained in the blocks themselves. The fundamental tenet of the Bitcoin Backbone is that sincere participants receive fresh chains from the network at the beginning of each round and choose the longest valid chain to mine. At the conclusion of the round, if they discover a new block (by finding a little hash), they broadcast their chain.

The GHOST protocol can also be expressed at the same level of abstraction. The GHOST Backbone protocol, as described above, is founded on the idea that blocks that do not become part of the main chain should nevertheless be taken into consideration when deciding which chain to use. Players maintain a tree of all mined blocks they have received in order to accomplish this, and then they choose which chain to mine using the greedy heaviest observed subtree (GHOST) rule.

Every round, Miners add valid1 blocks sent by other miners to their tree, updating it. Similar to Bitcoin, a block must be a legitimate child of another tree block in order to be added to the tree. As long as the blocks are valid, the opponent is free to add them anywhere in the tree. miners attempt to add one or more blocks to the chains they select, just like in Bitcoin. Finally, a tree of blocks is saved and updated during each round in the main function. A miner then broadcasts any changes to his tree to all other miners on the network.

Layer 2 Staking:

When the block is mined and supplied to the other miners' tree, it also verifies the block for the next few seconds which is achieved through layer 2 authentication. Metchain is one of the first blockchains which supports Layer 1 Mining and Layer 2 Staking. A unique concept of NFT consensus providing annual yield percents (APY) returns based on rarity and the vesting period. Within the process of NFT staking, The NFT and the staked MET coin amount is locked with the transaction. The balance of MET coins is then transferred to the Metchain Coinbase for the vesting period. When the vesting time is completed the NFT is unlocked and staking rewards plus agreed APY are sent to the respective user's wallet. NFT locking and unlocking can only be achieved through Byzantine Fault Tolerance System (BFT). If the nodes do not agree on the unlocking period and staking time and staking user wallet and block hash where they staked, then the nodes will come to force BFT’s to check the altered blocks. Any block alteration will suspend the node for 3 days unless it’s unbanned manually. A node banned through BFT’s will be broadcast to all nodes, so it remains banned. This increases the security of the user staked coins and user wallets.

Layer 2 blocks have their own consensus system where the nodes communicate with each other and decide which staked NFT becomes the validator for each block till the next MET Block is generated.

(a) = Mini block

(b) = Mega Block

© = MET Block

When © block is created, all the nodes confirm and add it. At this point, each node is randomly selected to supply the selected staked NFT validators for all blocks that will be created between every (a) block. There are 6 NFT staked blocks which are sub-seconds apart and is decided by all the nodes then only the next (a) block can be submitted or accepted. Any block generated before that will be counted as orphan.

Any transaction in orphan block is still counted as valid no matter what the block state is but the orphan block must be validated as orphan on all nodes. And the transaction should have been broadcast throughout the node system.

Conclusion of NFT staking:

15 NFT Staked blocks every (a) Mini Block.

12 (a) Mini Block in every Mega Block.

5 (b) Mega Block in ever MET Block.

1 Met Block consists of 900 NFT staked blocks which must be same and verified through all the nodes. If these 900 NFT staked blocks do not match BFT's will be forced into action to prevent any attack.

Proof-of-Stake (PoS) is a revolutionary consensus algorithm that is gaining popularity in the blockchain space. In contrast to traditional Proof-of-Work (PoW) algorithms, PoS relies on validators who are chosen to create new blocks based on the number of tokens they "stake" or lock up as collateral. As the blockchain industry evolves, new and improved versions of PoS algorithms are emerging, promising greater efficiency, scalability, and environmental sustainability.

The Drawbacks of Proof-of-Work:

While PoW has been the backbone of several successful blockchain networks, it comes with some significant drawbacks. The most prominent issue is its high energy consumption. Miners must compete to solve complex mathematical puzzles, leading to enormous computational power consumption, which has raised concerns about its impact on the environment.

Additionally, PoW is susceptible to centralization, as it often rewards those with access to the most powerful and expensive mining equipment. This concentration of power can compromise the decentralization and security of the network. This is resolved by adding layer 2 PoS to the blockchain.

Metchain has enhanced the security by adding another layer for the consensus, which includes the NFT. Only the NFT holders will be considered validators and allowed to stake. Layer 2 blockchain PoS above layer 1 blockchain PoW will also reduce the transaction time to sub-seconds and create the fastest transactions possible with low transaction fees of 0.00025%.

The Advantages of New Metchain Proof-of-Stake Algorithm:

1. Energy Efficiency: One of the most significant advantages of new PoS algorithms is their energy efficiency. Since they do not rely on energy-intensive mining processes, the energy consumption is drastically reduced compared to PoW. This makes PoS a more sustainable and eco-friendly option for blockchain networks, aligning with global efforts to combat climate change.

2. Scalability: PoS algorithms offer improved scalability, enabling networks to process a higher number of transactions per second. With reduced energy requirements, validators can process transactions more quickly and efficiently, contributing to a smoother user experience.

3. Decentralization: New Metchain Proof-of-Stake Algorithm addresses the centralization concerns of PoW. Validators are chosen based on their stake, incentivizing token holders to participate in securing the network. This democratic approach encourages a broader distribution of power, ensuring the network's resilience and security.

4. Security: PoS algorithms enhance network security by penalizing malicious behavior. Validators are required to lock up their tokens as collateral, which they stand to lose if they act against the network's best interests. This economic incentive promotes honest behavior and discourages attacks.

Conclusion:

The evolution of blockchain technology has led to the emergence of new and improved PoS algorithms, marking a significant step towards a more efficient, scalable, and sustainable future for the industry. As these algorithms continue to gain traction, blockchain networks are becoming more environmentally friendly, secure, and accessible. With the implementation of PoS, the blockchain space is poised to further revolutionize various sectors, from finance and supply chain management and beyond. Embracing the potential of PoS algorithms, the future of blockchain technology looks promising, as it strives for a decentralized, interconnected, and eco-conscious global ecosystem.

Metchain V3 Wallet:

Metchain wallets are unique in many ways. Metchain wallet creation uses a bip39 mnemonic seed to generate the wallet but in such a way that it becomes difficult for user to anticipate the seed. Each wallet has its own timestamp included in the wallet to generate on wallet address locally. Keeping the wallet secure unless the timestamp is same the same wallet address won't be created nor can be recovered even with the same seed unless recovered using private key.

Private key generation also uses multiple processes. Each seed being 1 creates 12 wallet addresses which are then again combined to a single wallet address making it more secure than ever. The signature verification for the transaction also follows the same steps. 12 + 1 signature must match before the transaction is accepted by any node to be processed. Transaction signature and approval uses Elliptic Curve Digital Signature Algorithm (ECDSA) for all wallet signatures merged as one. If the transaction gets approved by 12 and rejected by 1 it will reject the transaction as the signature must match the wallet address and its 12 wallets in single wallet.

Metchain Anti-Asic:

Metchain being unique uses an Anti-Asic Mining sub-algorithm. Which includes multiple hashing algorithms. When a certain criterion is met by the integrated mining algorithm it automatically modifies the sub-algorithm. This results in Kheavyhash Asics being unable to provide valid shares. They might be able to connect or stay connected to the blockchain, but all the shares will start getting rejected. Keeping it safe for GPU miners.

If the system detects unusual traffic on the node for hashes the algorithm is modified automatically for all nodes after reaching consensus. All nodes agree to 1 modification in the algorithm, if any node has not come to agreement and has miners connected all the blocks submitted from that node will be counted or marked as orphan blocks but transactions will still be accepted.

Algorithm 1:

The GHOST backbone protocol, parameterized by the input contribution function I(·) and the reading function R(·). xC is the vector of inputs of all block in chain C.

1: T GenesisBlock . T is a tree. 2: state <- ε

3: round <- 0

4: while True do

5: Tnew <- update(T , blocks found in Receive()) 6: C <-˜ GHOST(Tnew)

7: hstate, xi <- I(state, C˜, round, Input(), Receive()) 8: Cnew <- pow(x, C˜)

9: if C 6 ˜ = Cnew or T 6= Tnew then

10: T <- update(Tnew, head(Cnew)) 11: Broadcast(head(Cnew))

12: end if

13: round <- round + 1

14: if Input() contains Read then 15: write R(xC) to Output()

16: end if

17: end while

Algorithm 1 Explained:

For completeness' sake, we now go over the remaining steps in the GHOST backbone protocol. Function update (see Algorithm 4) refers to how the block tree is updated. Function pow (see Algorithm 3), which has to do with block mining, is the same as the one described in the Bitcoin Backbone.

Algorithm 2:

The GHOST backbone protocol, parameterized by the input contribution function I(·) and the reading function R(·). xC is the vector of inputs of all block in chain C.

1: T GenesisBlock . T is a tree. 2: state <- ε

3: round <- 0

4: while True do

5: Tnew <- update(T , blocks found in Receive()) 6: C <-˜ GHOST(Tnew)

7: hstate, xi <- I(state, C˜, round, Input(), Receive()) 8: Cnew <- pow(x, C˜)

9: if C 6 ˜ = Cnew or T 6= Tnew then

10: T <- update(Tnew, head(Cnew)) 11: Broadcast(head(Cnew))

12: end if

13: round <- round + 1

14: if Input() contains Read then 15: write R(xC) to Output()

16: end if

17: end while

Algorithm 2 Explained:

For completeness' sake, we now go over the remaining steps in the GHOST backbone protocol. Function update (see Algorithm 4) refers to how the block tree is updated. Function pow (see Algorithm 3), which has to do with block mining, is the same as the one described in the Bitcoin Backbone.

Algorithm 3:

The proof of work function, parameterized by q, D and hash functions H(·), G(·). The input is (x, C).

1: function pow(x, C)

2: if C = ε then . Determine proof of work instance 3: s <- 0

4: else

5: hs 0 , x0 , ctr0 i <- head(C) 6: s <- H(ctr0 , G(s 0 , x0))

7: end if

8: ctr <- 1 9: B <- ε

10: h <- G(s, x)

11: while (ctr ≤ q) do

12: if (H(ctr, h) < D) then 13: B <- hs, x, ctri

14: break 15: end if

16: ctr <- ctr + 1

17: end while

18: C <- CB . Extend chain

19: return C

20: end function

Algorithm 3 Explained:

Two essential security characteristics of the Bitcoin backbone protocol; the common prefix and the chain quality property were taken into account in [above and algorithms 2 and 3]. If they remove a few blocks from the tail, the common prefix property assures that two trustworthy participants have the same understanding of the blockchain. On the other hand, the chain quality feature makes sure that chains from honorable players don't include long stretches of hostile blocks. These characteristics are defined as predicates over the random variable created by adding the viewpoints of all par- ties, which is represented by the notation view H(.)Π,A,Z (k, q, z).

Algorithm 4:

The tree update function, parameterized by q, D and hash functions H(·), G(·). The inputs are a block tree T and an array of blocks.

1: function update(T,B) 2: foreach hs, x, ctri in T

3: foreach hs 0 , x0 , ctr0 i in B

4: if ((s 0 = H(ctr, G(s, x))) (H(ctr0 , G(x 0 , ctr0)) < D)) then

5: childrenT (hs, x, ctri) = childrenT (hs, x, ctri) U hs 0 , x0 , ctr0 i . Add to the tree.

6: end if

7: return T

8: end function

Algorithm 4 Explained:

To make the concepts of persistence and liveness relevant to the manner in which parties confirm transactions, we slightly modify them. A transaction is ‘stable' in Bitcoin, for instance, if it is at least k blocks deep in the chain and has the parameter k. On the other hand, for a transaction to be deemed "stable" in GHOST, the subtree rooted at the block containing the transaction must be at least k in size.

From this point on, whenever we discuss the longevity or persistence of Bitcoin or GHOST, we'll be referring to the parameterized versions with the corresponding definitions of stability that we just discussed.