Suppose that you have a PoS chain, where there is some validator set V, and where blocks are subdivided into epochs of length EpochLength

; you want every validator to vote once in every epoch as this is important for the consensus cycle and to minimize any gains from RNG manipulation. There is a source of random entropy in the chain; suppose that the random entropy updates once per epoch. At the start of every epoch, we randomly shuffle V

and partition it into slices S_1 ... S_{EpochLength}

, each of size SSize = \frac{V}{EpochLength}

, and we split up activity into consecutive time slots T_1 ... T_{EpochLength}

.

There are two types of messages: blocks and attestations. A block is a data structure that contains a pointer to a parent block, as well as a set of messages, and a set of signatures (“attestations”) of the parent block or another earlier block. The block must include at least \frac{SSize}{2}

attestations of its parent, though it can contain more. A block must specify what slot number it is produced in; if a block is in slot i

, immediate attesters of that block must come from the set S_i

, and its signer must be the first member of S_i

.

[

%5BB_i%5D%20-%3E%20%5BS_i%20attester%202%5D%2C%5BB_i%5D%20-
%3E%20%5BS_i%20attester%203%5D%2C%5BB_i%5D%20-
%3E%20%5BS_i%20attester%204%5D%2C%5BB_i%5D%20-
%3E%20%5BS_i%20attester%205%5D%2C%5BB_i%5D%20-
%3E%20%5BS_i%20attester%206%5D%2C%5BS_i%20attester%202%5D%20-
%3E%20%5BB_i%2B1%5D%2C%5BS_i%20attester%203%5D%20-
%3E%20%5BB_i%2B1%5D%2C%5BS_i%20attester%204%5D%20-
%3E%20%5BB_i%2B1%5D%2C%5BS_i%20attester%205%5D%20-
%3E%20%5BB_i%2B1%5D%2C%5BS_i%20attester%206%5D%20-
%3E%20%5BB_i%2B2%5D%2C%5BB_i%2B1%5D%20-
%3E%20%5BS_i%2B1%20attester%202%20%5D%2C%5BB_i%2B1%5D%20-
%3E%20%5BS_i%2B1%20attester%203%20%5D%2C%5BB_i%2B1%5D%20-
%3E%20%5BS_i%2B1%20attester%204%20%5D%2C%5BB_i%2B1%5D%20-
%3E%20%5BS_i%2B1%20attester%205%20%5D%2C%5BB_i%2B1%5D%20-
%3E%20%5BS_i%2B1%20attester%206%20%5D%2C%5BS_i%2B1%20attester%202%20%5D%20-
%3E%20%5BB_i%2B2%5D%2C%5BS_i%2B1%20attester%203%20%5D%20-
%3E%20%5BB_i%2B2%5D%2C%5BS_i%2B1%20attester%204%20%5D%20-
%3E%20%5BB_i%2B2%5D%2C%5BS_i%2B1%20attester%205%20%5D%20-
%3E%20%5BB_i%2B2%5D%2C%5BS_i%2B1%20attester%206%20%5D%20-%3E%20%5BB_i%2B2%5D

834×788

](https://yuml.me/diagram/scruffy/class/%5BB_i%5D%20-%3E%20%5BS_i%20attester%202%5D%2C%5BB_i%5D%20-
%3E%20%5BS_i%20attester%203%5D%2C%5BB_i%5D%20-
%3E%20%5BS_i%20attester%204%5D%2C%5BB_i%5D%20-
%3E%20%5BS_i%20attester%205%5D%2C%5BB_i%5D%20-
%3E%20%5BS_i%20attester%206%5D%2C%5BS_i%20attester%202%5D%20-
%3E%20%5BB_i%2B1%5D%2C%5BS_i%20attester%203%5D%20-
%3E%20%5BB_i%2B1%5D%2C%5BS_i%20attester%204%5D%20-
%3E%20%5BB_i%2B1%5D%2C%5BS_i%20attester%205%5D%20-
%3E%20%5BB_i%2B1%5D%2C%5BS_i%20attester%206%5D%20-
%3E%20%5BB_i%2B2%5D%2C%5BB_i%2B1%5D%20-
%3E%20%5BS_i%2B1%20attester%202%20%5D%2C%5BB_i%2B1%5D%20-
%3E%20%5BS_i%2B1%20attester%203%20%5D%2C%5BB_i%2B1%5D%20-
%3E%20%5BS_i%2B1%20attester%204%20%5D%2C%5BB_i%2B1%5D%20-
%3E%20%5BS_i%2B1%20attester%205%20%5D%2C%5BB_i%2B1%5D%20-
%3E%20%5BS_i%2B1%20attester%206%20%5D%2C%5BS_i%2B1%20attester%202%20%5D%20-
%3E%20%5BB_i%2B2%5D%2C%5BS_i%2B1%20attester%203%20%5D%20-
%3E%20%5BB_i%2B2%5D%2C%5BS_i%2B1%20attester%204%20%5D%20-
%3E%20%5BB_i%2B2%5D%2C%5BS_i%2B1%20attester%205%20%5D%20-
%3E%20%5BB_i%2B2%5D%2C%5BS_i%2B1%20attester%206%20%5D%20-%3E%20%5BB_i%2B2%5D)

Notice that a block at step i+2

can include not just attestations of the parent block created at step i+1

, but also attestations of blocks earlier in history, as well as attestations of blocks outside of the same chain.

If a block is produced by the first member of S_i

during timeslot i

that is on the head, the other attesters are expected to co-sign this block. If a block does not get produced on the head, then the other attesters can sign a message asserting to what they consider the current head to be. Hence, there can actually be two

ways that an attestation of a block with height i

gets included in a block with height j > i+1

:

1. The creator(s) of the block(s) with heights i < h < j

fail to include it

1. The creator(s) of the block(s) with heights i < h < j

fail to produce blocks, or produce blocks that are not on the head, in which case attesters can create attestations at height h

of blocks of height i < h

Attestations of a block's parent that come in slots later in the first slot when such an attestation could appear (ie. case (2)) can count toward the \frac{SSize}{2}

attestation minimum (which is what allows the chain to progress even when <\frac{1}{2}

of validators are online), but attestations of blocks other than a block's parent do not.

Note that a signer and an attester are in symmetric positions in the block structure but not at network layer: because the signer is the sole mandatory

attester, in general the signer signs a block immediately after making it and broadcasts it with the signature, and other attesters then sign after them. This allows the total aggregate signature of a block to be only one signature in size and require only one pairing to compute, at least in the best case where everyone agrees what they are signing.

Fork choice rule

The simplest fork choice rule to use is simple height counting. The ideal fork choice rule to use isGHOST, which works as follows:

1. Start at the genesis (or most recent finalized block)
2. Let the "current block" be the block the algorithm is looking at at the moment (ie. the genesis or most recent finalized block initially)
3. If the current block has zero children, then exit.
4. If the current block has only one child, set the current block to that child, and go
5. If the current block has more than one child, set the current block to the child that has more valid most-recent signatures from a validator signing on either that block or some descendant of that block
6. If the current block has zero children, then exit.
7. If the current block has only one child, set the current block to that child, and go
8. If the current block has more than one child, set the current block to the child that has more valid most-recent signatures from a validator signing on either that block or some descendant of that block
9. Repeat (2) until exit, and return the current block as the head.

For example, consider the following diagram, where A, B… M are the most recent signatures, and the blocks they are located in are the blocks those signatures are attesting to:

[

%5BLast%20finalized%20block%20%7Bbg%3Ayellow%7D%5D%20-%3E%20%5BA%5D%2C%5BA%5D%20-%3E%20%5BB%5D%2C%5BA%5D%20-%3E%20%5BC%5D%2C%5BB%5D%20-%3E%20%5BD%20E%5D%2C%5BD%20E%5D%20-%3E%20%5BF%5D%2C%5BD%20E%5D%20-%3E%20%5BG%20H%5D%2C%5BG%20H%5D%20-%3E%20%5BM%5D%2C%5BG%20H%5D%20-%3E%20%5BI%20J%20%7Bbg%3Agreen%7D%5D%2C%5BF%5D%20-%3E%20%5BK%5D%2C%5BK%5D%20-%3E%20%5BL%5D

258×1088

](https://yuml.me/diagram/scruffy/class/%5BLast%20finalized%20block%20%7Bbg%3Ayellow%7D%5D%20-%3E%20%5BA%5D%2C%5BA%5D%20-%3E%20%5BB%5D%2C%5BA%5D%20-%3E%20%5BC%5D%2C%5BB%5D%20-%3E%20%5BD%20E%5D%2C%5BD%20E%5D%20-%3E%20%5BF%5D%2C%5BD%20E%5D%20-%3E%20%5BG%20H%5D%2C%5BG%20H%5D%20-%3E%20%5BM%5D%2C%5BG%20H%5D%20-%3E%20%5BI%20J%20%7Bbg%3Agreen%7D%5D%2C%5BF%5D%20-%3E%20%5BK%5D%2C%5BK%5D%20-%3E%20%5BL%5D)

The head is the green block, because:

- A is the only child of the last finalized block
- B and C are the children of A. B has far more signatures backing it or its descendants than C.
- (D E) is the only child of B.
- F and (G H) are the children of (D E). (G H) has 5 signatures backing it or its descendants, F has 3, so (G H) wins.
- (I J) and M are the children of (G H). (I J) has 2 signatures backing it or its descendants, M has one, so (I J) wins.

GHOST preserves the property that the chain that has the most signatures supporting it is the winning chain, which is an important criterion to make reversion, censorship and other attacks maximally difficult.