Tl;dr:

We can exploit the fact that messages in the beacon chain are created in constant time intervals to gossip more efficiently (and/or faster).

General Idea

There are two general gossip strategies:

- Push

: Nodes just send rumors they learned about to their peers

- Pull

: Nodes ask their peers for new rumors and they send them only then

Pushing is extremely efficient in the beginning of the lifetime of the rumor as almost all messages inform new nodes. But at the end most (but not all) nodes are already informed, so most of the transmissions are wasted.

On the other hand, pulling is very inefficient at the beginning: Only the creator of the rumor is informed and finding them by sending requests without knowledge who they are takes many rounds. But once most of the nodes have been informed, it gets extremely efficient, as only uninformed nodes keep sending requests and those are successful with probability quickly approaching 1.

So ideally one would like to combine the two strategies, switching from push to pull once half of the nodes are informed. But usually this is difficult because identifying the point in time at which to switch is difficult.

However, in the beacon chain the situation is different as blocks are supposed to be created at predefined points in time. And as attestations are triggered by new blocks, the same holds true for most (if not all) of the payload of beacon chain blocks. So here we can switch from pushing to pulling simply based on the time that has passed since the expected rumor creation time.

The ideal switching time of course depends on the propagation time in the network, so it's not known a priori. But it can easily be optimized by counting how often pulling is successful. Ideally it should be 50% for the first try. If the success rate is higher, the node should start pulling earlier, if it's lower it should keep pushing for longer. Alternatively, nodes can estimate the propagation time directly by comparing message creation and arrival time.

Note that in many protocols the gained efficiency can be traded for shorter propagation times: If we spend less resources per peer, we can afford to open connections to additional peers, such that more nodes get informed per unit time.

Simple Exemplary Protocol

There are two message types called PUSH and PULL. Each rumor has a time of birth t_0

, known also to nodes who haven't received it yet. Each node N

keeps track of an estimate of the time T_N

it takes to inform half of the network. N

is supposed to do the following:

On receiving PULL:

- If it knows about the requested rumor, respond with a PUSH containing the rumor.
- Otherwise, respond with an empty PUSH.

On receiving unrequested PUSH at time t

:

- If t < t_0 + T_N

: PUSH it to all peers.

- Otherwise, do nothing.

On receiving a PUSH as a response to a PULL:

- If it was successful, do nothing.
- Otherwise, consider waiting for a short time and send another PULL to a random peer.

At time t_0 + T_N

:

- If the rumor has been received, decrease T_N

.

- Otherwise, increase T_N

and send PULL to a random peer.

References:

Inspired by: http://zoo.cs.yale.edu/classes/cs426/2012/bib/karp00randomized.pdf