This is a pretty minor tweak, but it could give faster finality in certain cases. Commandment II states

a validator must not vote within the span of its other votes.

Note that this is stricter than necessary. For the safety proof to work, a validator must be prohibited from casting a finalization

vote within the span of another vote; it's not necessary to prohibit any

vote within the span of another vote. (A "finalization vote" is a vote from some checkpoint c

to a direct child of c

.)

Here's a combination of votes which would be prohibited by the current Commandment II, but permitted by the more lenient variant. The gray nodes are justified, and the green arrows represent votes.

For this pattern of voting to be rational, we also need to tweak the fork choice rule. It currently states

FOLLOW THE CHAIN CONTAINING THE JUSTIFIED CHECKPOINT OF THE GREATEST HEIGHT

If we interpret this 100% literally, it's not quite optimal. Block creators want to maximize the weight of their fork after their own block is added, not before. So if a block creator sees a supermajority for 1 -> 5b

, they should follow 5b

and make it justified, even though 5a

had the greater weight before.

Consider a voter who has already voted 2a -> 4a

. It's time to vote for \text{h} = 5

, and they observe that 1 -> 5b

is close to a supermajority. (This wouldn't happen if all validators had perfect information, but most of the network might not be aware of 2a

's justified status.)

With the previous rule, this validator's vote has no use. They can either abstain, or vote 2a -> 5a

which is doomed to fail. With the more lenient rule, they can vote 1 -> 5b

to increase the chance that 5b

becomes justified. If it does, that's good for the network, since it creates an opportunity to finalize a checkpoint next time.

I'm thinking of using this Casper variant in a new protocol, so please let me know if anything doesn't seem right

.