I believe I thought of a way to parallelize operations dynamically on a blockchain while maintaining both decentralization and security. Looking on the internet, I could not find any documentation of this design in particular so I just wanted to share it in case it might actually help with scaling. In contrast to sharding, this design provides a much higher adaptability to the current utilization of the network, however, does not improve the storage consumption on each node like sharding does. After doing some reading, I see no reason why this could not be implemented on top of sharding for increasing flexibility and transaction throughput. Please let me know your thoughts!

Basic design

Block-link:

In the initial state of the system there exists only one chain. Each block is connected to its predecessor through a digital signature on the contents of the block itself and a pointer in the preceding block, stipulating who the signer of the next block should be. Thus, every block will have only one prescribed creator.

Validation:

Subsequent to its creation, a block will be passed to a set of randomly (, unambiguously, and unpredictably) chosen validators which will receive the block for verification in a tree-like order (l: layers, b: branches

). Each Validator will check the block for its correctness, afterwards, if correct, sign it (including the previous signatures), and send the signature to the block creator, the other validators in the same layer, and also, together with the block itself and the remaining b^l-1

signatures, to the next b

validators whose address will be determined based on the signature. Should only a single validator spot a mistake in the block, they can report it and every address which has already signed off will be punished for malicious behavior. If someone owned 99% of all validators/block creators of a system with 126 validators per block (l = 6, b = 2

), their overall chance of manipulating the chain would only be around 28%. With 90% ownership, the chance is already down to 0.00015% and at 50% it is practically 0 (5.9*10^(-37)%).

Overclocking:

Prescribing a block creator in advance and using only a small number of validators offers a fairly exciting new way of running a blockchain. The main chain described here is still very limited in terms of transaction throughput. However, in a scenario where the blockchain reaches its limit, it now can essentially split itself into two (overclock). If a certain condition is met, indicating a sufficient utilization of limits, a block producer has to include two addresses into one block. The first one will once again point to the next block creator, yet, the second one will point to the block creator after the next one. Each of the two chains will take turns so that chain A includes the block with the block number n

, B includes n+1

, A n+2

and so forth. The chain has two ‘heads’ now, however, still practically functions as one chain and has an average block time (bt

) of only 1/2 of what it used to be. Hence, the maximum possible transaction-rate just doubled. If the supply of incoming transactions falls below a certain level, the latest chain will simply end (the last block will not include a pointer) and new blocks will only be attached to the initial chain. Assuming s

is the number of times the chain has been overclocked/split, the overall block time will be bt/(s+1)

. Since the process of overclocking is only constrained to the number of validators and block creators in the network, a network as big as the Ethereum network could potentially be overclocked thousands of times.

[

Blockchain_overclocked

791×241 4.4 KB

](https://ethresear.ch/uploads/default/original/2X/3/3bdedbb64fb1d88a2bed33afaf08da67bfb7f345.png)

Specifications

There are plenty of challenges coming along with this concept. Below I addressed a few of these issues and also provided some ideas for how to deal with them.

- Transaction distribution among block creators

(protect block creators from accidentally including transactions multiple times -> double spending)

- When to split or merge a chain
- Punishment for malicious behavior
- Block withholding attacks

(a block creator or validator withholding a block)

Transaction distribution among block creators

A question remaining is how transactions will be distributed among block creators in order to prevent transactions from accidentally being processed multiple times and double spending.

One possible solution is to assign addresses to one specific head of the chain (like in sharding). All heads will now operate as if they were a separate blockchain each and only accept transactions from their assigned addresses. After a split, both heads would get an equal amount of addresses from the chain they are rooted in to have the greatest possible chance of a somewhat efficient distribution.

When to split or merge chains

While the process of overclocking/splitting a chain can greatly enhance the efficiency of a blockchain; it is also important to split and later merge at the right time. A mechanism for automating this process could look as follows:

All we need are two conditions, one for accelerating and another for decelerating the chain. For accelerating the chain the requirement could be to have at least b

blocks among the latest a

blocks, filling a minimum of x%

of the block size. Accordingly, should at least b

of the most recent a

blocks cover less than y%

of the block-size, the next block to be mined will not be allowed to include an address pointer.

Punishment for malicious behavior

A quite familiar way of disincentivizing rule breaking when creating and validating a block is confiscating previously staked tokens in the event of misbehavior. By staking, an address qualifies for being a block creator/validator and risks losing this stake when validating or creating a malicious block.

Block withholding attack

This is probably the biggest concern in a system operating as described above. Since there is one stipulated block creator from which all validators are derived, the continuing of the whole chain is dependent on them. If some block creator or validator decided to not create or validate a block, there would be nothing the network could do about it except for forking away from the chain.

To resolve this issue, we need to add a mechanism, limiting the time available for producing a single block. The basic idea here is to make block production open to anyone with a big enough stake but still significantly thwart everyone apart from the chosen block creator. A block created by a non-chosen address must include the hash of the previous block (n+1

) and a nonce into the new block since the hash of this block is required to be smaller than some value x

. In order to make up for the varying time, each node will take for solving the block, and spare energy resources the target block time here should be noticeably larger than the native block time of the chain. Whenever the aforementioned block-type is included in the chain, the creator will receive the entire stake from the address which was initially chosen to create the block.

[

Blockchain_overclocked_bwa

791×241 6.04 KB

](https://ethresear.ch/uploads/default/original/2X/6/6eb2d78efaf0e945dcc9994d63ec1e9d8dee2e75.png)

Should a validator withhold their signature, the block creator is on the hook since he is the one getting punished for not submitting a valid, verified block. Therefore, should the block creator suspect a withholding attack is being performed by a validator, there are two things they can do:

1. Change the block data so that their digital signature will generate a different validator tree. (Should not be allowed - significantly lowers security!)
2. Submit an ‘accusing transaction’ containing their digital signature on the block data and the address(es) of the validator(s), refusing to submit their approval. The validator accused of withholding their signature has two options now:
3. They can submit their digital signature together with the block data so that the production of the block can proceed
4. Since validators withholding their signature could also just mean that they found the block to be invalid, they can also submit a fault-proof, showing that the block signed by the block creator is not valid.

If the accused fail(s) to submit a valid response, they will lose all their staked tokens.

1. They can submit their digital signature together with the block data so that the production of the block can proceed
2. Since validators withholding their signature could also just mean that they found the block to be invalid, they can also submit a fault-proof, showing that the block signed by the block creator is not valid.