Hi All!

I am Cheng, founder of Alephium project. We found a novel sharding protocol which supports native cross-shard transactions, i.e. no two-phase commit is needed for cross-shard txs. We call it blockflow. It's a general sharding algorithm, but the idea could be applied to Ethereum.

We briefly describe the ideas here. In blockflow, we first shard addresses into G groups, then we distribute all transactions into GxG shards based on the input address and output address. Let's say that shard (i, j) consists of transactions from group i to group j. For group i, it only needs to download transactions from shards (j, i) and (i, j), so 2G-1 shards instead of GxG shards in total. This contributes to scalability. Transactions from group i to group j would be submitted directly to shards (i, j), which is probably the first approach that avoids two-phase commit.

We use a specific data structure + finality algorithm to reach consensus for all shards. Thanks to our data structure, the algorithm suffers from 51% attack instead of 1% attack. The cost of using this sharding algorithm is that it requires additional storage for the new data structure that stores shards dependencies, the cost is around 100B-200B per block. We don't need a super node, but "full node" would consist of G node with one for each group to form a complete ledger.

We also have some innovative features for scaling smart contracts. We decompose smart contracts into a token part and a data part. Then, we provide a scripting language for token level programming. It's a practical tradeoff as we don't want to have a built-in VM language for the data part.

More details and proofs are available in our white paper. I'd welcome any questions and discussions! Or give us feedback via Twitter @alephium

.