Assignment 2 - Understanding Consensus

Max Liu

January 23, 2019

Signature Validation

We need to ensure that the user trying to spend UTXOs is in fact the owner of the UTXOs. This is done by using the public key that is given by the UTXO and using that key to verify the signature (created by the users secret key that they input as part of the transaction) and the hash of the transaction. Essentially we are checking that the public key of the UTXO and the given secret key are in fact a pair. Implementation of signature validation consisted mostly of ensuring that the different keys and hashes were converted back to their original forms after being base58 encoded to improve readability.

Proof of Work

Our goal with the poof of work is to be able to 1) prevent sybil attacks and 2) prevent constant forking. We want the proof of work to be hard enough so that the system is self healing and so that the honest nodes are solving the proof of work faster than the malicious nodes.

In this implementation, I want to be able to scale the difficulty of the proof of work as the number of nodes, and as a result the network's total compute power, increases. I also want each node to solve a unique proof of work. The goal of this is so that as the number of nodes increases, the time it takes for the network to mine a block remains relatively the same; furthermore, this makes it harder for nodes to skip ahead and try proofs that have not been tested yet (because each POW is unique to the node) and allows nodes to take credit for proofs. To do this, I increase the difficulty of the POW by 1 every time the number of nodes double, I also add in the nodes verification key into the hash for the POW.

A benefit to this solution is that it helps to alleviate forking issues that arise when more nodes join the network because the time it takes for blocks to be added will roughly stay the same. Furthermore, we can now attribute a solved POW to a specific node because the verification key is embedded in the proof. A downside is that this does not account for powerful computers that can solve POW's faster than all of the other nodes. A solution to this would be to dynamically change the difficulty of the POW not in regards to the number of nodes but rather to the time it takes for the network to mine a certain number of blocks and then adjust the difficulty depending on the target amount of the time (similar to how Bitcoin adjusts its difficulty level).

Consensus

The goal of this consensus problem is essentially to find a chain that contains the longest prefix, with at least one additional chain, and the longest chain, given that its prefix is the longest or there is no longer prefix. The challenge of this problem is how to evaluate all of the different chains given by the nodes and find the longest prefix with the longest chain. To solve this problem, I implemented a solution where each chain tries to prove that it is the longest prefix with the longest chain by comparing itself to every other chain and keeping track of the longest prefix match that it has. Once it has found a prefix length that is bigger than

the current longest prefix, it will update the length of the longest prefix as well as the length of the longest chain for the given prefix length. If there are prefix length ties, the longer chain is selected, furthermore, longer chains have no advantage if they do not have the longest prefix. Finally, if two chains have the same prefix length and same chain length, the incumbent is chosen.

It is also important to note that it is irrelevant how many other chains share the same prefix, we care only about the length of the prefix.