Laurentia Pool

A peer driven pooled hash model

Con Kolivas, Ryan Ellis

laurentiapool.org

Abstract.

Small, low overhead, regional specific mining pools promote dispersion of hashrate enhancing decentralization of a pooled mining network while maximizing return to the users. With localized peer driven pools, committed to a specific hash to blockfind cadence, and utilizing direct coinbase payments is the most efficient execution to achieve the best of both results: increase of revenue to self and local economy, and adding to the decentralization of pooled mining. Utilizing a peer-based model and a trustless scoring scheme provides users direct control over their newly proven work, and allows the pool operators the ability to focus on user UI/UX down to the individual. Limiting pool space allows the group to easily predict and adjust to changes in the mining network and environment with predetermined commitments, incentivizing peer collaboration in achieving targets and UI/UX development while not requiring it.

Introduction.

The current pooled environment is dominated by pools based in a singular region and modeled after profit driven enterprise , of which most to all retain a user's funds for a payout at a specified threshold increasing liability to pool and risk to the miner. High fee rates are also a considerable amount of economic stimulus that is injected into the host country, far away from the miner themselves. An open pool system does allow freedom but also allows for variances in revenue as miners move or shift hash in accordance to their will and preference, likely disrupting earnings across multiple pools for multiple parties. Certain scoring systems can be manipulated or even hash moved to other SHA-256 networks by pool hosts, with and without a miner's consent. Requiring far more trust from the miner than necessary.

With a limited user base on a single localized server, latency is reduced by not having to relay across multiple nodes to submit shares to the master pool. Increasing performance, reducing chance of orphaned blocks, and eliminating the need for excessive development or maintenance costs in these areas. Utilizing lean code and operations, fee rates can be minimal, adding considerable gain to gross profits for a miner, and translating directly to net for an established one.

Scoring.

Modified ckpool.org SPLNS code by Con Kolivas specifically for Laurentia Pool. *This section consists of excerpts as taken from ckpool.org:*

SPLNS stands for Score Per Last N Shares. Score refers to the fact that share value is weighted by the difficulty of the share found. Last N Shares refers to the fact that the score is a rolling score based on N shares where N means 5 x the current difficulty. The rolling average is weighted according to when the shares were found - the more recent shares are the more they are worth.

Hop proof - the system cannot be gamed to earn more by hopping on and off during lucky blocks. Short "ramp up time" compared to PPLNS - rewards more rapidly rise to stable levels when you first start hashing.

Block finder rewards - a large share weight is attached to block finds (but is applied to the next block reward since user rewards are included in the existing unsolved block reward.) The sooner the next block is found, the higher the block finder reward is.

Malicious & faulty miner disincentives - since shares are rewarded according to the difficulty of the share found, malicious miners that withhold block solves, or faulty hardware that doesn't produce high diff shares will have substantially less reward than on any other pay scheme or pool.

SPLNS calculation is done on the fly and updated every minute based as a product of HERP DERP.
Herp stands for Hash Extracted Rate Product - where each share is worth sqrt(MIN(share diff, network_diff) / work_diff) * work_diff / 2
Derp stands for Difficulty Extrapolated Reward Payment - where the reward equals the user's herp divided by the pool's herp i.e. it is the expected reward should a block be found now.
The pool's Herp is simply added until it reaches 5 * network difficulty. After that it is biased every minute by scaling existing herp down to add the latest minute's herp and all users' herp is adjusted by the same scale the pool’s was.

Accurate statistics by the minute - include accurate estimates of hash rate, herp and derp - which translates into an accurate estimate of payout should a block be solved.
Luck estimates - as the difficulty of each share is calculated into the user's herp, it is compared to an equivalent 'last N share' calculation to determine the miner's overall luck. This is done on a per worker and per user basis and displayed in the stats. The larger the miner and the longer they mine, the closer to 1 their luck will be.

Coinbase generation - Block solve reward is distributed directly from the block to each user, meaning each user gets a 'mined' transaction directly into their wallet as soon as the block is solved so there is no wait to get paid and no pool wallet storing user's rewards. Rewards will be considered 'immature' by bitcoin rules so will be unspendable until 100 network confirmations have passed (approximately within 17 hours).

Blocks are always as full as network congestion demands - never mine empty or light blocks and yet is still extremely fast at getting new work out to miners on block changes to minimize wasted work and decrease orphan risk. Rapid propagation of blocks thanks to high speed low latency connection to further decrease orphan risk.

Shares all transaction fees with miners - transaction fees account for a significant portion of mining reward now and blocks full of transactions the extra rewards can be substantial.

Network.

Separate physically located nodes from the same pool could just as easily find different blocks to other nodes on the same "pool" as completely different pools can - this means nodes of the same "pool" are just as likely to orphan blocks from other nodes of the same pool as a completely different pool. Having miners from remotely connected nodes to the same master pool just means there is actually more potential conflict over what a new block is than that imposed by mining to a relatively distant single pool node.

What the pool policy is about which node is the "master" node ultimately determines which block gets propagated in a conflict situation between different physical nodes on the same pool. For example, if a pool's master node is in the USA, and it has a node in DE, and both nodes find a block at the same time, then even though miners in DE have a local node, the master node will override their block with the USA node. This is actually _less_ likely to happen if the DE miners were to just mine directly to the USA node if the USA node takes priority. Alternatively, if the pool is designed such that the mined block takes precedent regardless of where it originates, then both the DE and USA nodes will get into a fight across the internet and the best connected one will win out. Again, this means there's no guarantee that mining locally on the same pool will save you from having your block orphaned.

Therefore, the most important measure is not latency to the node you're mining to, but the quality of the connectivity of the node responsible for block propagation in both latency, priority with bitcoin nodes, and speed of block processing. A network of miners across the globe with separate mining nodes in different locations is not as powerful as a relatively local cluster of the same number of miners mining to the one physical location. The main determinant of risk of orphaning blocks is speed of pool processing of potential blocks, block propagation, and the quality and speed of connectivity to other bitcoin nodes. The Great Firewall of China presents its own challenges with potentially great latency at moving blocks from outside the GFW inside and vice versa. However, whilst there is a lot of mining hashrate clustered within the GFW meaning a lot of blocks originate within the GFW, the block that succeeds in an orphan block race is determined by the block that is seen first by the greatest number of validating bitcoin nodes. If every node outside the GFW sees one block and every node inside the GFW sees another, given the larger number of nodes outside the GFW, the one outside will likely succeed. Mining on a node inside the GFW when you are outside it then makes no sense at all given the extra latency of trying to get through the GFW.

Structure.

A limited size pool of 100 potential users with an average minimum hashrate target per user through contract. An extremely reduced fee to cover operational overhead and UX enhancement. Direct visibility to work on chain, and collaboration with operators for a personal UI/UX experience. A fair scoring scheme paid out directly from coinbase with every blockfind. Predictability through data and math for a specific cadence to blockfind, with statistics that update every minute. “Gaming”, hop proof, and disincentivizes poor or malicious equipment use with a score scheme that reward for every hash.

Incentive.

By focusing on the user and their location coupled by lean efficient operation the pool is able to drive revenue gain to the miner and incentivize collaboration with peers. While the miner is in a trustful scenario through commitments, contractual and inherent, the trustless payout scheme and high margin mining incentivize participation to the pool without any underlining ethos. Requiring no further action outside of proving work to the network.

Conclusion.

The pool model minimizes trust through a fair direct payout scheme, enhances the pooled mining environment through dispersion of network hash globally. And with a pool cap by design, cannot centralize the network itself. Will increase revenue while offering predictability of reward cadence. Reduces trust given to the pool operator by the miner and spurs collaboration with peers, without requiring it. It is reliant on a miner’s commitment and is only trustful in that regard as the miner can fully verify the chain on which they are working directly from the network. Utilizing simplicity and efficiency in code and operations.