DEPLOYING BOBTAIL AS A SOFT FORK

1. Introduction

Bobtail is an algorithm for lowering the variance of mining in proof-of-work blockchains. In PoW systems currently, a block is mined whenever a single PoW result is discovered that is less than a target. In Bobtail, the mean of k such values must be less than a target. In this document, we describe the changes that are required to deploy Bobtail as a soft fork. The soft fork requires one backwards-compatible change to the mining process and a few backwards-compatible changes to p2p network announcements.

The goal of a soft fork is that all blocks produced by miners following Bobtail can be validated by miners that have not upgraded. There are costs to introducing Bobtail as a soft fork instead of a hard fork. First, the amount of data stored on the blockchain is slightly higher. Second, k must be incremented at most 1 each adjustment period, whereas a hard fork could raise k to a higher value in one block without an issue. Third, even with an incremental increase to k, a small spike in average block time will occur initially, though it can be mitigated with a smoothing algorithm.

2. Voting

The soft fork begins with a phase where miners signal that they are willing to mine according to Bobtail rules. The signal is expressed as an addendum to block announcements. Addenda are not stored in the chain. Once a sufficient percentage of mining power has signaled positively, the second phase begins, which we describe below. Each vote states the largest value of k the miner is willing to support. The second phase adopts the the largest value of k supported by a sufficient amount of mining power.

3. Adjusting the Mining Process

In the soft fork, just as in the current system, miners hash a header comprised of an unchanging version, prior, and difficulty, while iterating over nonces, the Merkle root, and the timestamp. Let h be the hash of a block header:

$$h = H(\text{version, prior, root, difficulty, time, nonce})$$

In current systems, a block is mined whenever $h < t$ for a target t. For miners running Bobtail, a *proof* is discovered whenever $h < kt$. A block is mined only once the mean k such proofs is below the target.

4. Announcements

4.1. **Proof Announcements.** If a newly discovered proof is smaller than at least one of the k-lowest proofs previously announced for the given prior (and $h < kt$), the miner announces the full header over the network.

To validate a proof announcement, a peer requires two more values: a *support* and an address for depositing rewards. To express these values and tie them securely to the proof, the announcement includes

(1) A *null data transaction*[1] that states the high order 4-bytes of a *support* (sufficient for matching) and a 20-byte address for depositing rewards; i.e.,
`OP_RETURN 24 0x<support><address>`.
(2) The values required to prove that the data transaction is represented by the Merkle Tree root stated in the header. These Merkle proofs can be expressed with 352 bytes for today's blocks (growing to 512 bytes for a block with 65,535 transactions).

Upgraded miners will consider the announcement valid if it meets the kt bound and the Merkle tree contains the transaction.

4.2. **Block Announcements.** When the mean of k proofs are below a target t, a block B_1 will be mined. This block is announced by the miner of the first order proof (i.e., the proof that hashes to the smallest value). The INV of each of the $k-1$ associated proofs is listed as an addendum to the block announcement; alternatively, the INV of each proof could be expressed as a Graphene message. Any missing proofs are requested by the neighboring peer.

Miners that have not upgraded will validate the block. The reason is that the the header of the Bobtail block is the proof with the lowest hash value, i.e., the first order statistic. The first order statistic will always be less than the difficulty stated in the header. The reason is that the mean of a set of k values must be greater than or equal to the lowest value in the set.

Upgraded miners follow stricter rules and they accept newly announced blocks only if the following three criteria hold.

(1) All the existing rules for valid blocks hold true.
(2) The mean of the k proofs are below the difficulty.
(3) Each proof is valid according to the rules above.
(4) Let B_0 be the prior block of B_1, and assume B_0 is valid. For B_1 to be considered valid, it's coinbase must disburses appropriate funds to the miners that

[1] See `https://bitcoin.org/en/glossary/null-data-transaction`.

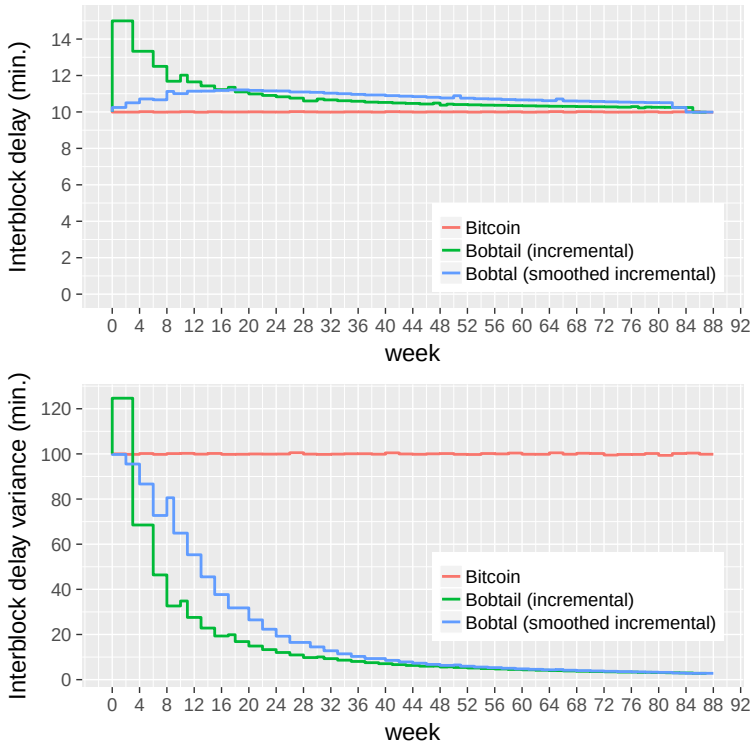

FIGURE 1. Soft fork of Bobtail

authored the proofs of B_0. The reward amounts are discussed in a separate document.

(5) The block includes a null data transaction that states, for each proof used by B_0 (other than the first order statistic), its nonce, Merkle root, time, its support, and all Merkle proofs.

With these values in place, all information needed to validate B_0 are written to the blockchain historically, though one block later. Of course, the values for B_1 will appear in B_2. None of these items will cause miners that have not upgraded to consider the block invalid.

The transaction will contain about $(k-1)(80+377)$ bytes of data for Bitcoin Core's 1MB blocks. For example, when $k = 10$, the transaction will be 4.0KiB (overhead of 0.4%); and $k = 40$, the transaction will be 17.4KiB (overhead of 1.7%).

5. Adjustments to Difficulty Calculations

After the soft fork begins, k must be incremented by not more than 1 at each difficulty adjustment period. For example, if the miners immediately switch to mining based on the mean of $k = 40$ proofs against a target t selected when $k = 1$, the first Bobtail blocks will be delayed by hours or possibly days until the difficulty adjusts several times; in fact, the adjustments themselves will be delayed.

These problems are not present if k is increment by one each period. Figure 1 shows, in comparison to the current system, two methods of raising k, from 1 to 40, that we have investigated using a blockchain simulation.

- The straight line (in red) on the plot shows the average block time for Bitcoin now (where $k = 1$). As expected, the mean inter-block time is 10 minutes with a variance of 100 minutes.
- A spiked line (in green) shows the effect of incrementing k by 1 each adjustment period. An initial spike slows average block time to 15 minutes and difficulty is adjusted at 3 and 6 weeks in, instead of at 2, 4, and 6. After those adjustments, the adjustments are not delayed.
- The more moderate line (in blue) shows a smoothing strategy where the mean is weighted, at first, towards the first order proof. This strategy keeps the inter-block time average to 11 minutes or under while still reducing variance.

In the smoothed method, k is treated as the desired end value, and k' represents an intermediate value used in Bobtail. At each adjustment period, k' is incremented by 1. A weighted mean of k' proofs are used to determine if a block is mined: the first order statistic's weight is $\frac{k-(k'-1)}{k}$ and all other proofs are weighted as $1/k$. Though we know of no attack, we have not analyzed if such a weighting introduces any vulnerabilities to the protocol.