Incremental Block Synchronization

Thomas Bakketun, thomas@bitcoin.no
Stein Håvard Ludvigsen, sh@bitcoin.no

Incremental Block Synchronization (IBS) is a method for nodes of the Bitcoin network to faster reach blockchain consensus. No changes in the consensus rules are required.

The mining nodes of Bitcoin will seek to form a small world network, where each node is directly connected to almost all other nodes of the network. Each node is working on extending the blockchain with their own block. Let's call that their candidate block.

In IBS, candidate blocks are built append only. Updates are continuously shared with the network.

IBS is not a block relay method, where a block is transmitted via several hops. Block relay will still be needed occasionally.

Benefits

Near instant block propagation

When a block is found, only the coinbase transaction and block header must be communicated to the other nodes.

Continuous operation

Currently the nodes struggle with huge spikes in bandwidth and computation demand when a new block is found. IBS spreads the work out, potentially removing the spikes completely. A node may build the merkle trees for some or all the other block candidates continuously.

Easy deployment

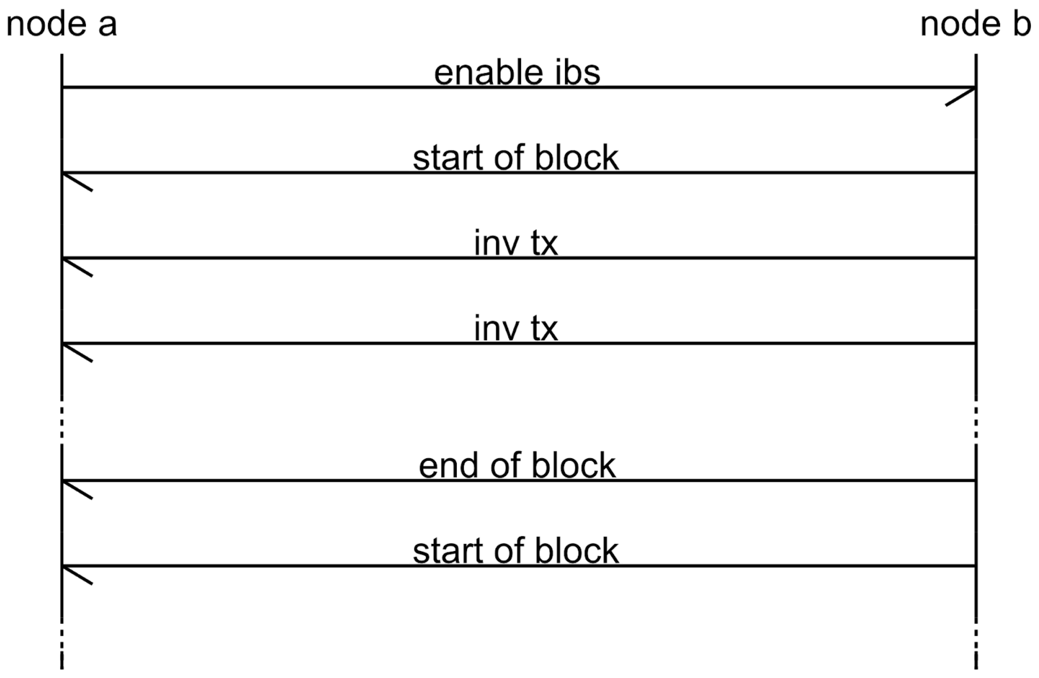
Two nodes can adopt this method and benefit from faster block propagation right from the start. As more nodes adopt IBS, the benefits increase.

Faster transaction confirmation

When mining nodes tells the rest of the network which transactions they are currently trying to append to the blockchain, doing reliable zero confirmation becomes easier. For example, if it can be seen that 90% of the hash rate has added a transaction to their candidate block, that essentially means the transaction has 0.9 confirmations.

Protocol draft

Incremental block synchronization can be implemented with a few additions to the current Bitcoin peer-to-peer protocol.



Node a send the *enable ibs* message to inform b that it wants to receive continuous updates on it's candidate block.

Node b indicates it's support of IBS by sending a *start of block* message. After this point, a transaction inventory message sent from node b indicates that it has added that transaction to it's candidate block.

Node b might postpone the *start of block* message until it starts on a new block.

The payload of *start of block* is the previous block hash.

If node b finds a valid proof-of-work for its candidate block, it sends the *end of block* message. Payload is the transaction count, block header and the coinbase transaction. The receiver will truncate the block according to the transaction count. The transaction count is necessary because the mining hardware will tend to work on a block template that is not fully up-to-date with the candidate block.

Transaction prioritization

A mining node is free to pick which transactions to add to its candidate block. As an example a node could sort transactions into two queues, high pri and low pri. Transactions are then drip fed into the candidate block, at a higher rate from the high pri queue.

Incremental validation

For each peer a node can decide if it will just store the incoming candidate block transaction set or do incremental validation.

Incremental validation allows for extremely fast block propagation. When an end of block message is received, all that is left to do is to validate the provided block header and coinbase transaction and compute a few missing nodes in the merkle tree.

There is no need to make a final decision on this. Each node operator can continuously assess what is the best policy. It’s also possible to use a different policy for each peer.