Masterblocks: Scaling Blockchain by summarizing balances for Dash and Bitcoin Cash
[Research Paper DRAFT]

by Alexey Eromenko "Technologov";
al4321@gmail.com

abstract: this paper shows how it is possible to scale the network by creating multi-genesis blocks, and shrinking the size of the blockchain by removal of old transactions, adding balances for "dumb" (i.e. non-programmable) blockchains such as Dash and Bitcoin Cash. Masterblocks as described here adds a snapshot to cut the length of the historical blockchain relevant for consensus.
This idea could allow for over 3000 tx/sec on desktop class hardware.

1. Basic idea: I think we may be able to reduce a block chain size dramatically, by adding a balance summary (every X blocks) of the previous state, last block of it.
Theoretically it will solve the hard disk space issues, And will reduce network bandwidth requirements for setting up a new Full node. It will not solve new block propagation delays.
New Term: Let's rename our so-called new Summary Block into a Masterblock. (for clarity) It acts very much like a new genesis block, but with all old balances encoded as unspent outputs in the new block.

Applicability: "dumb" (non-programmable blockchains) with large blocks, so Ethereum is incompatible with this proposal. Bitcoin legacy (Core) will not gain much from it.
Bitcoin Cash and Dash (and possibly others) qualify.

Concept:
2017 [block 0]...[block n] --- the last block's balance could transfer into a new Masterblock 2018.
2018 [masterblock 0]

Balances (all utxo) of each address are copied into a new Masterblock as one transaction. Masterblocks are created in a deterministic fashion, so every node can see it and reproduce the correctness of a Masterblock.

Afterwards Mining should start build up on the new Masterblock. Just the history of transactions will be lost.

This idea, if it works, will allow Dash and Bitcoin Cash will beat Visa x 10 transactions easily, with a goal to break 1 billion transactions per day, in a decentralized way, via on-chain scaling

2. Current method of so-called "blockchain pruning", where Full Nodes simply remove old transactions is broken by design, because it disallows new nodes to be added from such truncated nodes. This forces network to keep Full Nodes for the coin to be operational.

Source:
https://news.bitcoin.com/pros-and-cons-on-bitcoin-block-pruning/
https://news.bitcoin.com/pros-and-cons-on-bitcoin-block-pruning/
3. Summary tables: Currently balances are NOT recorded anywhere, only transactions. By summing up all transactions up to block X, Full Nodes can calculate balances.

Basic Summary Table example:

Currently only transactions are written in the block-chain:
Alice had 10.0 BTC, pays to Bob 2.0 BTC, Bob pays to Charlie 0.5 BTC, etc…

With summary table, balances will be written as well, like this:

	Amount	Type
Alice	8.0	Bitcoin Cash
Bob	1.5	Bitcoin Cash
Charlie	0.5	Bitcoin Cash

Extended Summary Table (with colored coins and time locked):

What if a blockchain wanted to add some colored coins, say RedCoin ?
Then every user will have multiple entries in the DB, one for main coin, another for colored coin.
Let's say Charlie has 5 RedCoins and sent one to Bob. And Alice got 500 Greencoins during ICO, but those are Time Locked until Block X.

(address)	Amount	Coin Type	Time Lock
Alice	8.0	Bitcoin Cash	
Bob	1.5	Bitcoin Cash	
Charlie	0.5	Bitcoin Cash	
Alice	500	GreenCoin	TimeLocked until Block X
Bob	1.0	RedCoin	
Charlie	4.0	RedCoin	

*NOTE: important features, like "Coin Type" and "Time lock" may be hard-coded in the database. Some complex features like MultiSig scripts will go to a separate table; Other features may be dropped completely.

4. Masterblock must be generated in a deterministic way, to be reproducible by all miners and all Full Nodes.

5. There could be more types of nodes: Archival node, keeping all history of all the previous balances, and the Full Node, keeping only the current block chain.
In this scenario, only block explorers and scientists will need to keep an Archival Node. It is not be needed for payments and verification at all; not part of the consensus.

6. Downsides:

Are there any? Of course.
a. A bunch of features will need to be dropped for this optimization to work. Metadata and custom scripting may not work, (except though Spicy Table) – Spicy Table is there to solve complicated edge-cases, where summary tables will not work, where scripting may be required, such as MultiSig addresses.
This means that Colored Coins (like Omni and CounterParty) will need to be coded in a different way.
b. MultiSig: because MultiSig addresses require a script, they need to be put in a separate table that will travel across Masterblocks. This will go in a separate table called a “Spicy table” (part of the Masterblock).
c. Dash Masternodes: a separate list of Dash Masternodes needs to be kept, and as long as a Masternode balance keeps it's 1000.0 DASH, the Masternode is considered 'alive'. This will go in a separate table called a “Spicy table” (part of the Masterblock).
d. Application logic and smart contracts: are part of the block in Ethereum, so by deleting old data, app logic will get deleted also. This is why this optimization idea is incompatible with Ethereum.
e. Generating a Masterblock, may cause a delay in the network. (it's better to start generating it in advance, and finish just on time)
f. TIME Locked transactions may become a special type in the Extended Summary Table. (like TimeLocked Bitcoin Cash until block X or until time Y).

Therefore a Masterblock will need to have two new tables: a Summary Table (having balances of ALL standard addresses) and a Spicy table (for special cases).

7. Dust Cleanup:
Let's define 'dust' as unspent transaction outputs (UTXO) below the median fees for the last 10 blocks. (this removes a bunch of bloat from the blockchain)

Those tiny dust amounts can be distributed to the last 10 miners. (via adding to their balance’s summary table)

8. Recommended Application and assumptions:
Create a Masterblock summary every year.

Okay, Let's do some maths: (with a few assumptions)

My models assume 4000x on-chain capacity increase over Bitcoin’s (1MB4EVER policy).
1 GB block each 2.5 min = 4 GB in 10 min (equivalent of a 4 GB block in Bitcoin). This will allow us for 1 billion tx/day (=12,000 tx/sec). In Dash it would equal to 576 blocks/day x 1 GB/block = 576 GB/day. (In Bitcoin it would equal to 144 blocks/day x 4 GB/block = 576 GB/day.)

It's 576 GB/day x 365 (year) = 210 TB-per-year.

Without my idea we will grow into a multi-petabyte territory in 5 years. Will be hard, even with incentivized Masternodes.
With my idea, it would take only 18 Hard Disks (okay 20 HDDs, with RAID6) to keep an entire block chain (that's assuming high-capacity 12 TB HDDs; that both Seagate and Western Digital started producing in 2017). (future HDD capacity is projected around 50 TB in late 2020’s assuming a conservative 15% growth per year; so only 5 or 6 of those will be required, Full node can be run on a high-end desktop PC)

Block propagation ? It takes only 8 seconds on a Google Fiber (1 Gbit/s Internet) -- so I believe it's very much possible and feasible to grow with on-chain transaction scalability. I assume that 1 Gbit/s Internet will become common in a decade. (late 2020's)

The big problem I'm solving is adding new nodes, With my solution, only 210 TB of data needs to be transferred, in worst case scenario. Without my idea, adding new nodes will require transfer of multi-petabytes of data.

I do not recommend to remove old blocks immediately after computing a Masterblock, but rather with delay of several weeks or perhaps even until two Masterblocks are found, so that other hosts can download blocks, and re-compute the Masterblock in a deterministic way, independently.
Additionally, keep the block headers since the genesis block, so that Proof-of-Work isn't being lost, but accumulated.
After two Masterblocks, all contents from ALL blocks prior to those two Masterblocks may be safely deleted. (except Block headers, keeping the PoW)

9. Miner's fee – can be added into it's own balance. (directly to summary table)

10. Security: the idea is to integrate a Masterblock into the chain, so full Proof of Work protection takes place on the whole chain, including the Masterblock. This way Masterblock cannot be modified under the nose of the users.

Security is theoretically slightly weaker than standard Bitcoin security model:
Nodes will have about one month's time to validate the Masterblock, until old blocks get emptied. But in practice it shouldn't matter, as long as we don't experience a global black-out with a one month duration. This is why removing tx history must be delayed, and not immediate on Masterblock creation.

11. CPU: Various research projects (particularly Gigablock, run by Bitcoin Cash researchers), came up with the number of 100 tps. It is what an average Intel Core i5/i7 can process on a single thread. This translates to around 32 MB blocks.
Going higher will require either multi-threading -or- developing new hardware - special ASICs for accelerating signature processing - ECDSA-256.

12. Added bonus idea: use "Snappy" for blockchain compression; Like zip but much faster algorithm. This can cut blockchain size, summary tables, storage costs and network bandwidth requirements by half. (by Gil Klein)

Standard Bitcoin Block header

Bytes	Name	Data Type	Description
4	version	int32_t	The block version number indicates which set of block validation rules to follow. See the list of block versions below.
32	previous block	char[32]	A SHA256(SHA256()) hash in internal byte orderof the previous block's header. This ensures no previous block can

	header hash		be changed without also changing this block's header.
32	merkle roothash	char[32]	A SHA256(SHA256()) hash in internal byte order. The merkle root is derived from the hashes of all transactions included in this block, ensuring that none of those transactions can be modified without modifying the header. See the merkle trees section below.
4	time	uint32_t	The block time is a Unix epoch time when the miner started hashing the header (according to the miner). Must be strictly greater than the median time of the previous 11 blocks. Full nodeswill not accept blocks with headers more than two hours in the future according to their clock.
4	nBits	uint32_t	An encoded version of the target threshold this block's header hash must be less than or equal to. See the nBits format described below.
4	nonce	uint32_t	An arbitrary number miners change to modify the header hash in order to produce a hash less than or equal to the target threshold. If all 32-bit values are tested, the time can be updated or the coinbase transaction can be changed and the merkle root updated.

Source: https://bitcoin.org/en/developer-reference#block-headers

Modified Scalable Masterblock header, with removed transactions:

Bytes	Name	Data Type	Description
4	version	int32_t	The block version number indicates which set of block validation rules to follow. See the list of block versions below.
32	previous block header hash	char[32]	A SHA256(SHA256()) hash in internal byte orderof the previous block's header. This ensures no previous block can be changed without also changing this block's header.
32	Summary hash	char[32]	A SHA256(SHA256()) hash in internal byte order. The hash is a calculated hash on the summary addresses table and a Spicy table. (this replaces the Merkle root, and instead of a transactions summary hash, provides a balances summary hash)

4	time	uint32_t	The block time is a Unix epoch time when the miner started hashing the header (according to the miner). Must be strictly greater than the median time of the previous 11 blocks. Full nodeswill not accept blocks with headers more than two hours in the future according to their clock.
4	nBits	uint32_t	An encoded version of the target threshold this block's header hash must be less than or equal to. See the nBits format described below.
4	nonce	uint32_t	An arbitrary number miners change to modify the header hash in order to produce a hash less than or equal to the target threshold. If all 32-bit values are tested, the time can be updated or the coinbase transaction can be changed and the merkle root updated.
8	sumtable_length	uint64	Summary table may grow huge in the future in several decades, potentially with billions of users.
8	spicytable_length	uint64	Spicy table may grow huge in the future in several decades, potentially with billions of users.
-	sumtable	variable	Summary table (all standard addresses' balances go here)
-	spicytable	variable	Spicy table (special cases)

Masterblock contains NO transactions whatsoever, only balances + a Spicy table keeping special cases, such as MultiSig scripts and Dash Masternode list.

What will a new Blockchain look like?

[Block] → [Block] → [Masterblock] → [Block] → [Block]
...
New normal blocks should treat Masterblock like a Genesis block. A block with balances, but no history.

UPDATE on 09.09.2018, Thanks to Meni Rosenfeld, he proved that my ideas is not new, and a similar idea was presented back in 2012:
https://bitcointalk.org/index.php?topic=74559.15

-06.Jan.2018. By Alexey Eromenko. "technologov"; al4321@gmail.com
Original idea: https://github.com/dashpay/dash/issues/1380

Changelog: [v1.1] 10.01.2018: Added two features: Colored coins and Time locking.

Changelog: [v1.2] 07.02.2018: Added new feature: CPU and ASICs