

Turbo-Geth - optimising Ethereum client(s)

Alexey Akhunov

Supported by a grant from Ethereum Foundation



Ledgerwatch

Outline

Authenticated data structures (trees)

Persistence of state in geth and turbo-geth

Alternatives - sparse merkle trees, self-balancing trees

Block/tx processing architectures: geth and turbo-geth

Latest performance data

Thoughts on light clients and fast sync

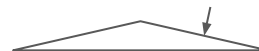
Future experiments (in-memory state, better organised history)

Status

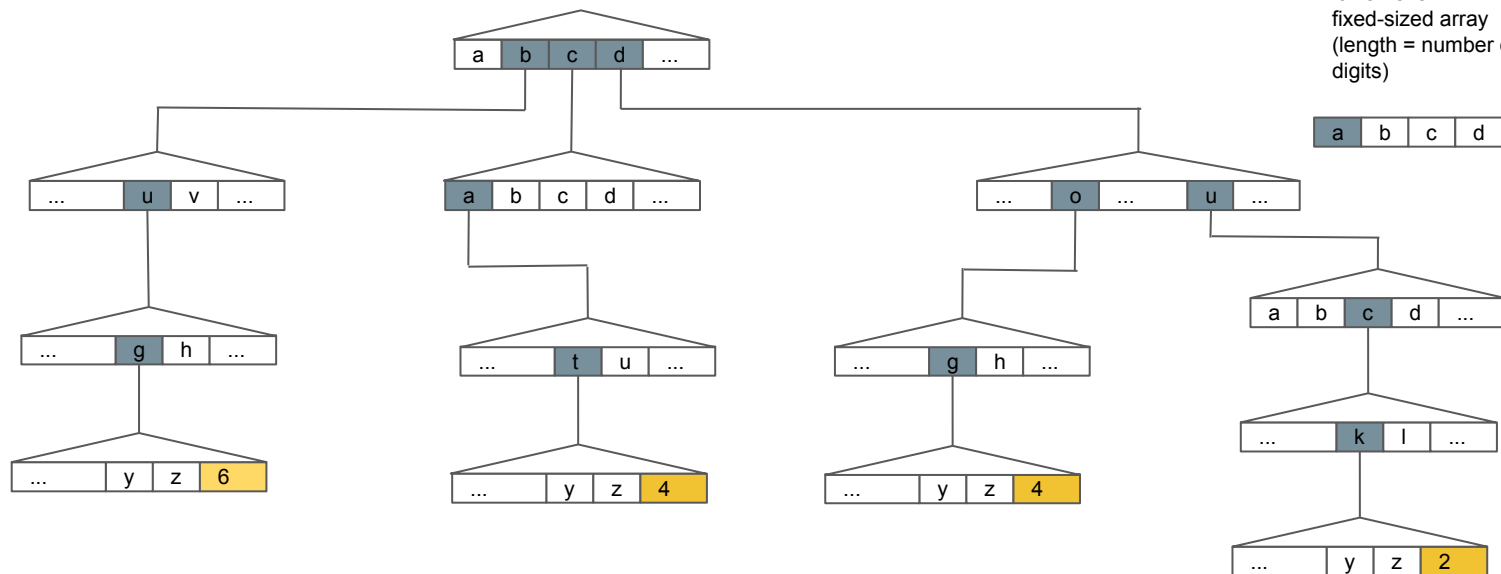
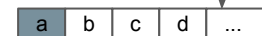
Merkle radix tree (trie)

{ 'cat':4, 'duck':2, 'dog':4, 'bug':6 }

hash of concatenation:

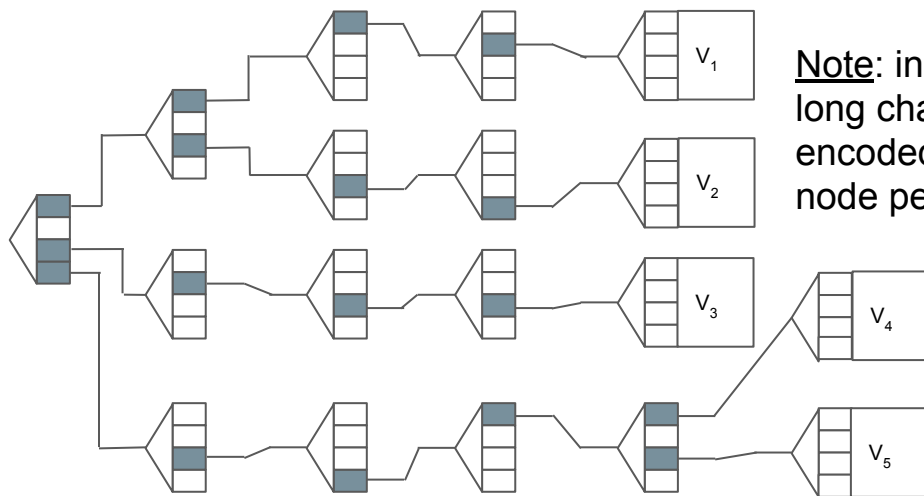


does not actually store
digits, but hashes from
lower level in
fixed-sized array
(length = number of
digits)



Merkle radix tree (trie)

Merkle radix tree supports compact encoding of modifications, short proofs of membership and non-memberships. However, if the keys are relatively long, representation tends to be quite inefficient:

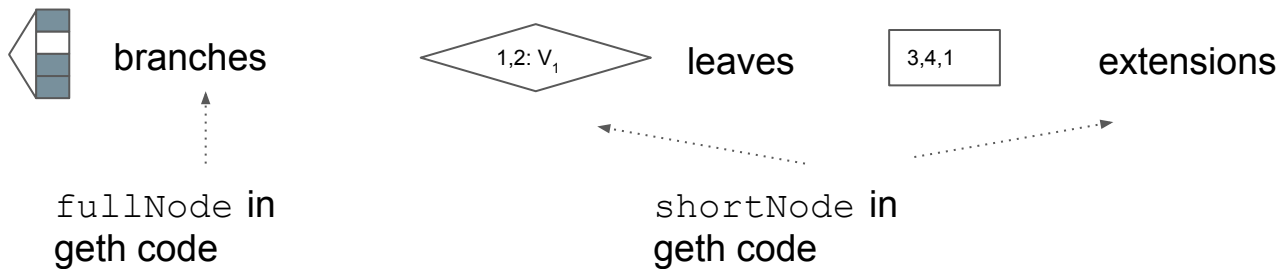


Note: inefficiency comes from the fact that long chains of digits for the keys are encoded by the chains of tree nodes, 1 node per digit

Merkle Patricia tree (trie)

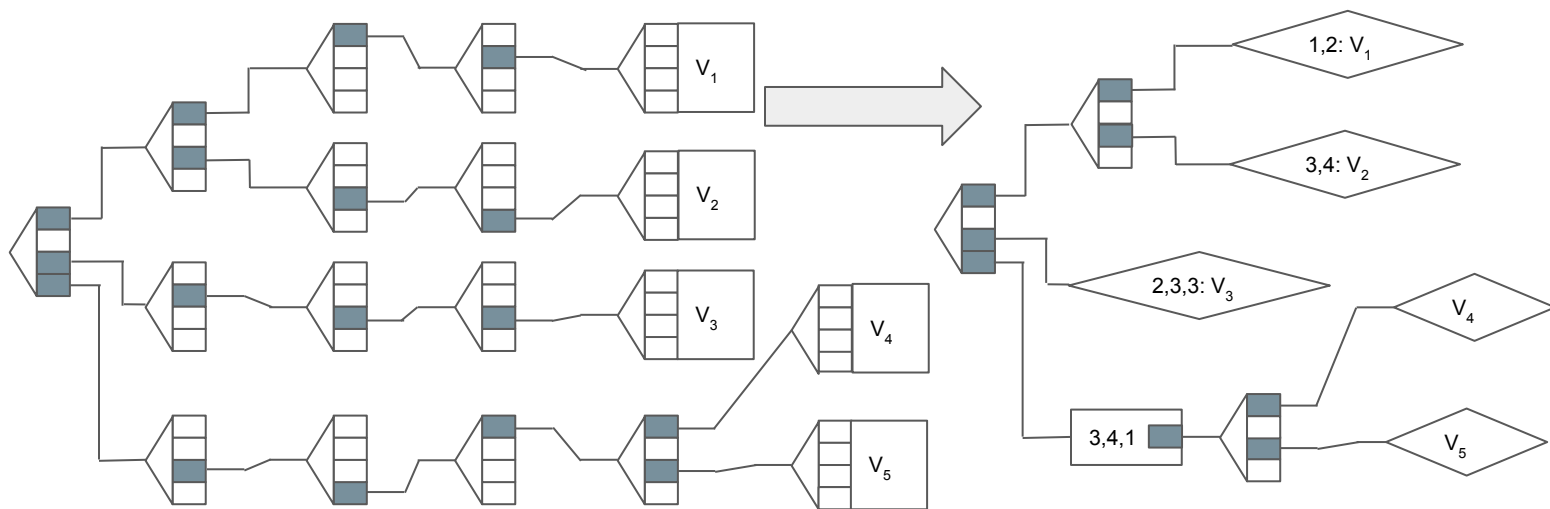
PATRICIA stands for “PRACTICAL ALGORITHM TO RETRIEVE INFORMATION CODED IN ALPHANUMERIC” and was invented in 1968 independently by Donald Morrison and Gernot Gwehenberger.

Patricia tree in Ethereum’s version addresses the inefficiency of radix tree by having 3 types of nodes:



Merkle radix tree (trie)

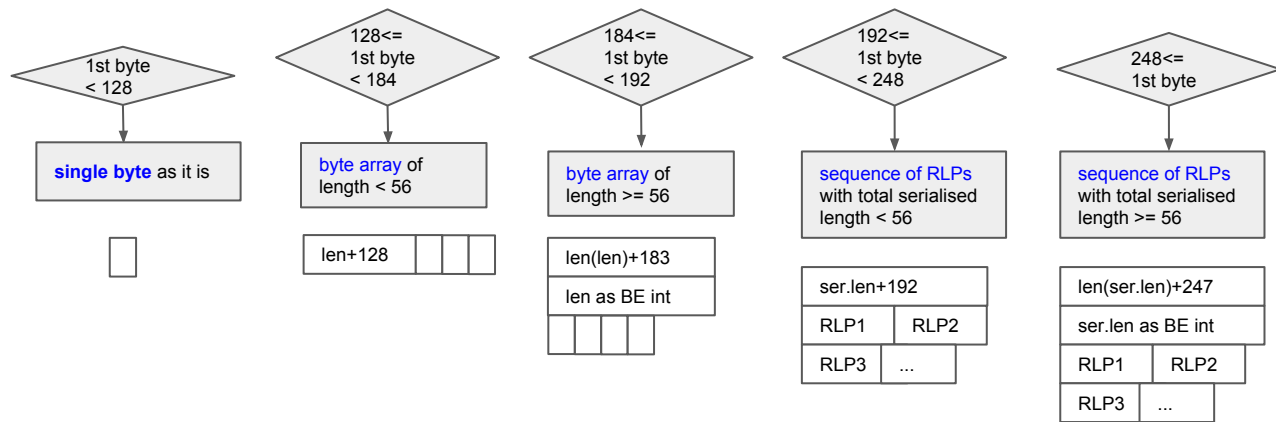
If we transform the radix tree from earlier to Patricia tree, it will look like this:



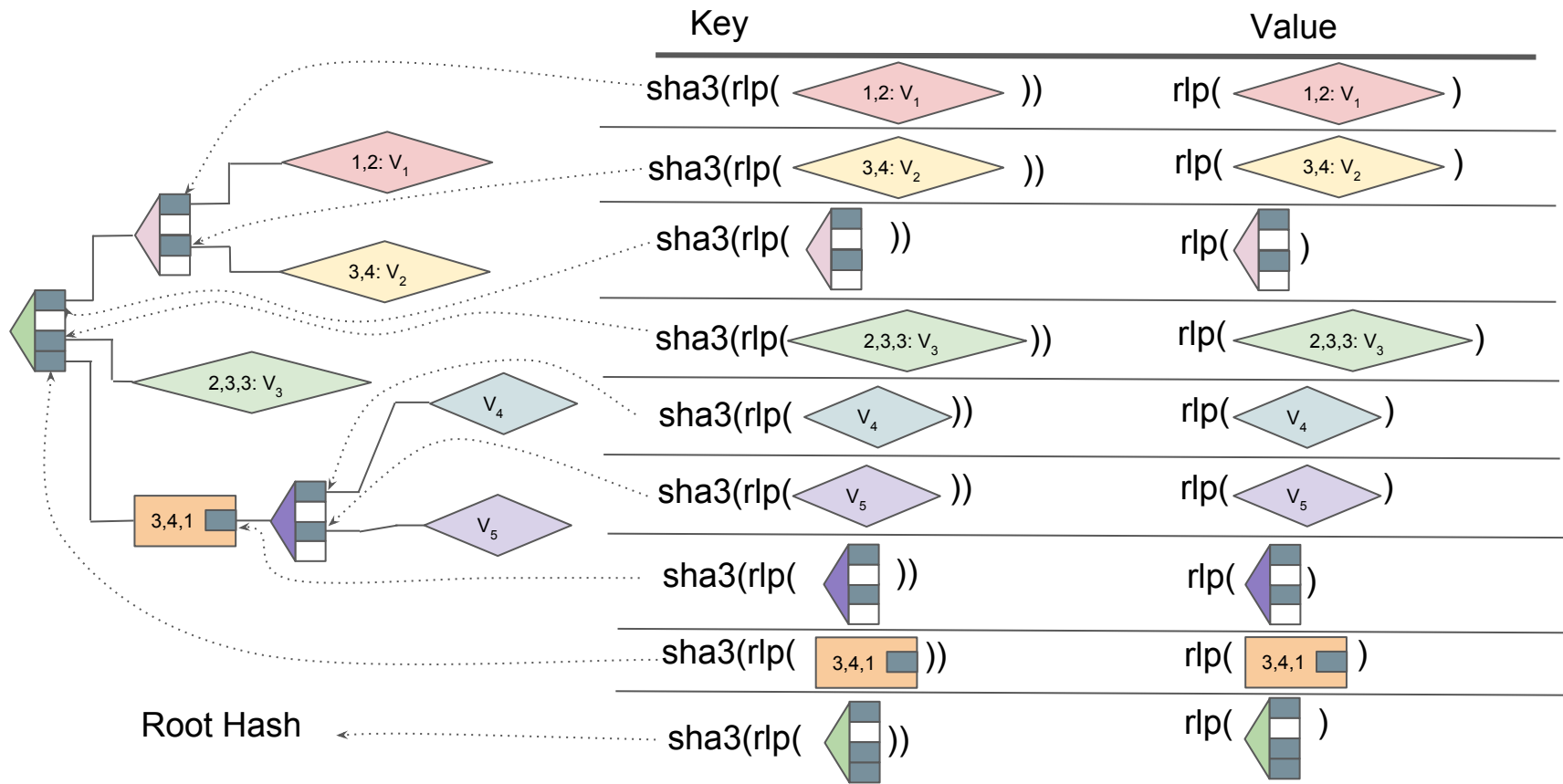
Ethereum: radix 16 (0,1,2,...,b,c,d,e,f), all keys 256 bit == 32 bytes == 64 nibbles

RLP - Recursive Length Prefix encoding

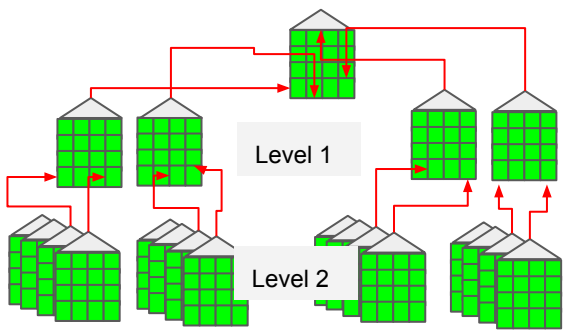
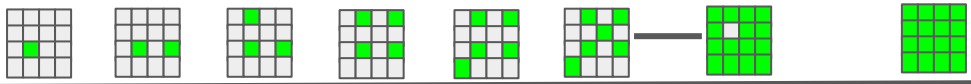
RLP is able to encode two types of structures: byte arrays of arbitrary size and sequences of other RLP-encoded structures. Interpretation of RLP stream:



Persistence of Patricia tree in geth

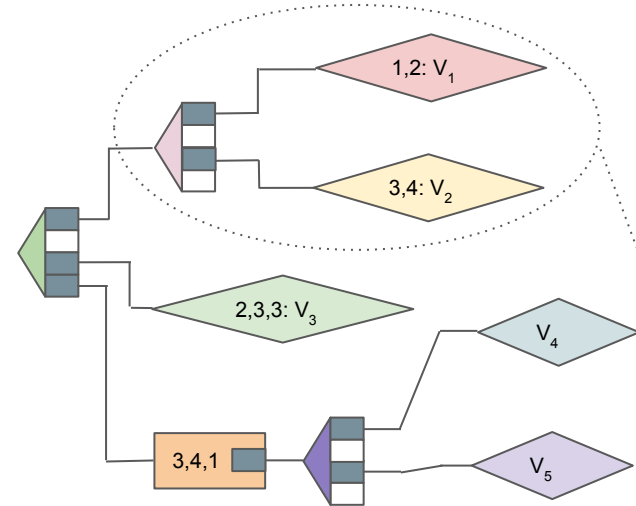


Block 5'032'091



							$1 = 16^0$
							$16 = 16^1$
							$256 = 16^2$
Level 3							$4'096 = 16^3$
Level 4							$65'536 = 16^4$
Level 5				7	41	1'023'387	25'141
Level 6	5.7m	4.4m	2.1m	0.7m	117k	39'123	
Level 7	18m	1.1m	31k	628	11		
Level 8	2.3m	72k	86				
Level 9	144k	1'675					
Level 9, 10, 11, 12	3'484	67					

Persistence of ~~Patricia tree~~ in turbo-geth



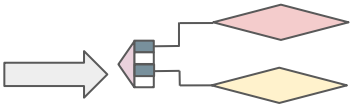
DEPTH DOES NOT MATTER

Key	Value
1,1,1,2	V ₁
1,3,3,4	V ₂
3,2,3,3	V ₃
4,3,4,1,1	V ₄
4,3,4,1,3	V ₅

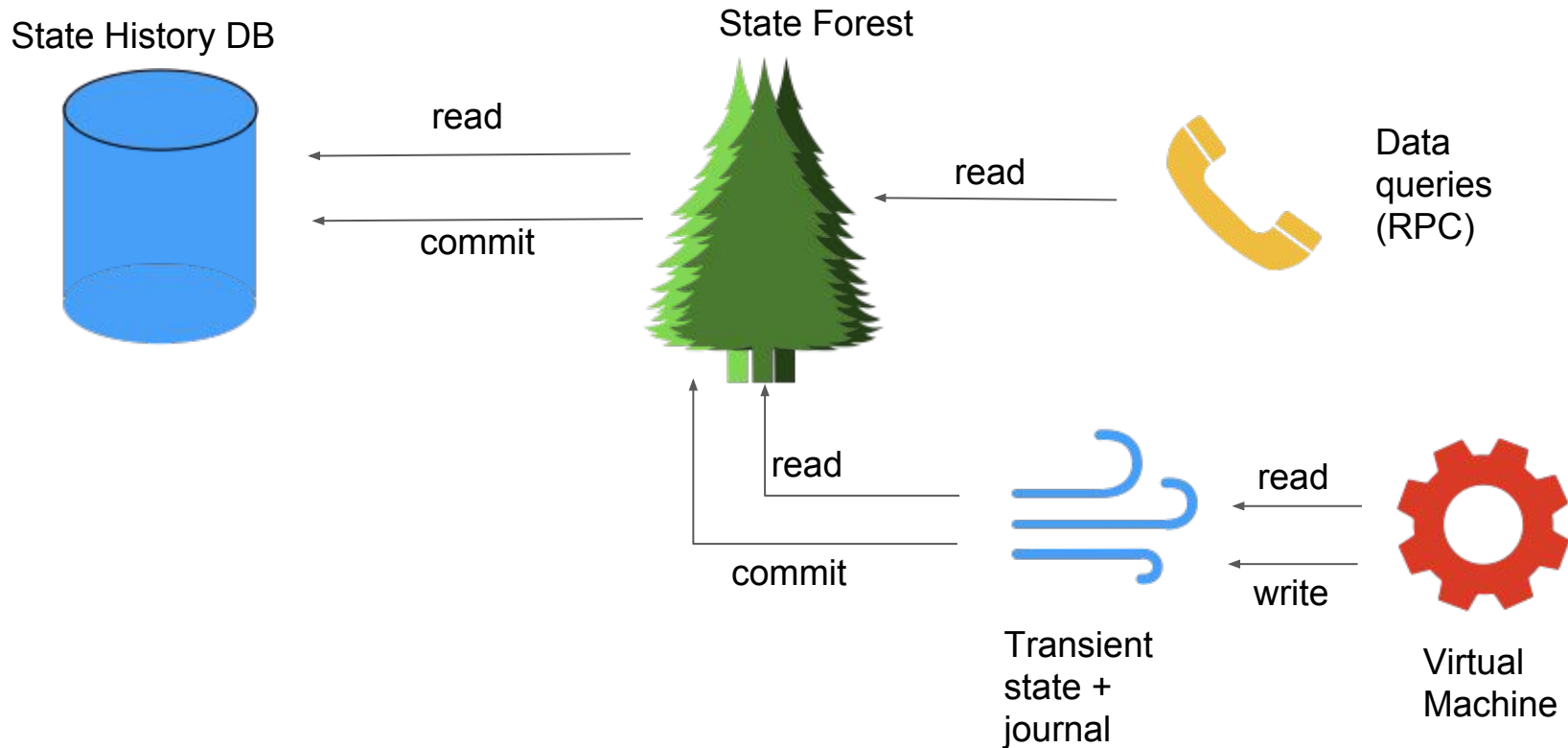
Goes here because it is sorted

Range query:
1, *, *, *, >=-blockNr

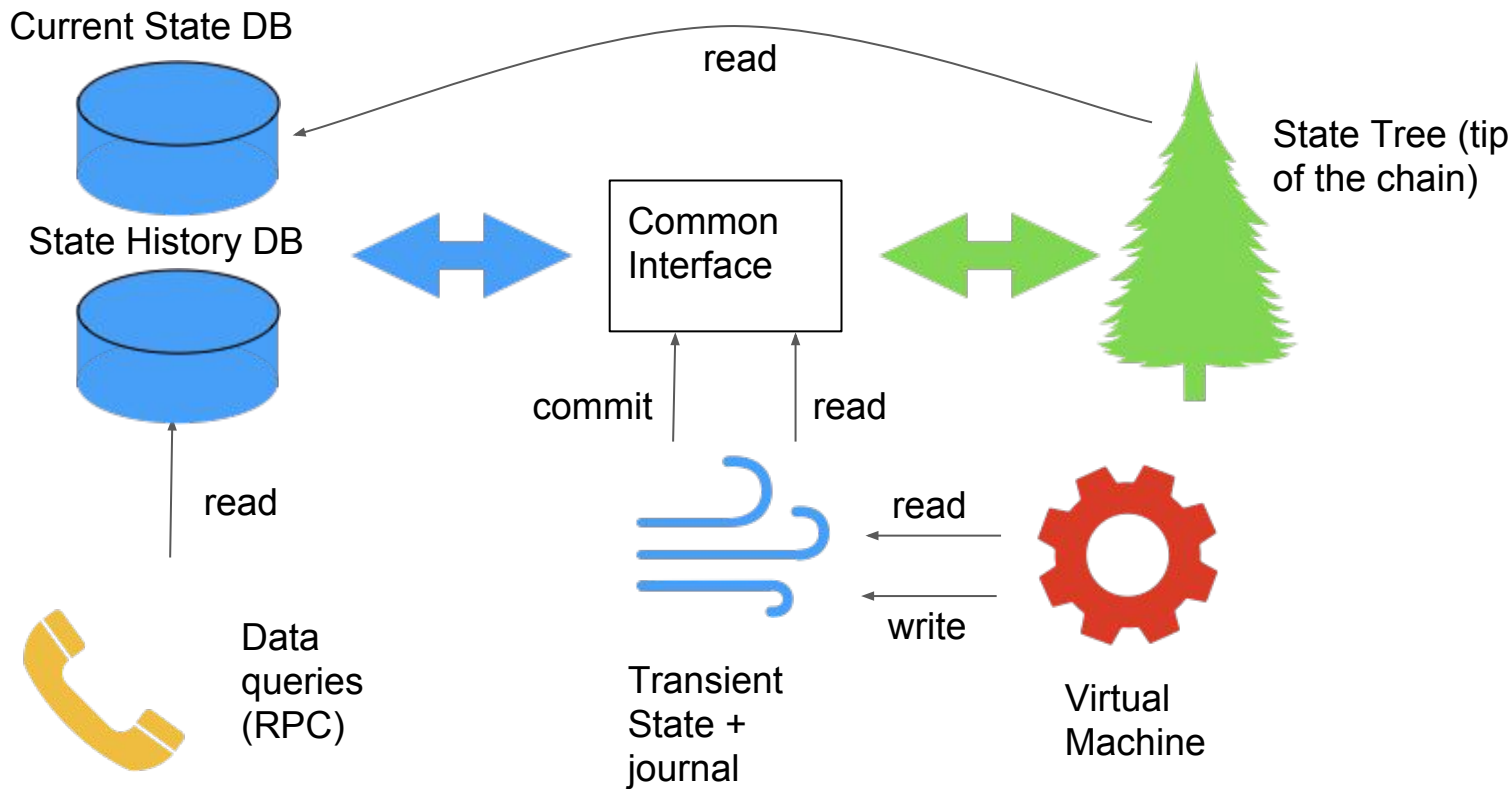
1,1,1,2,-blockNr V₁
1,3,3,4,-blockNr V₂



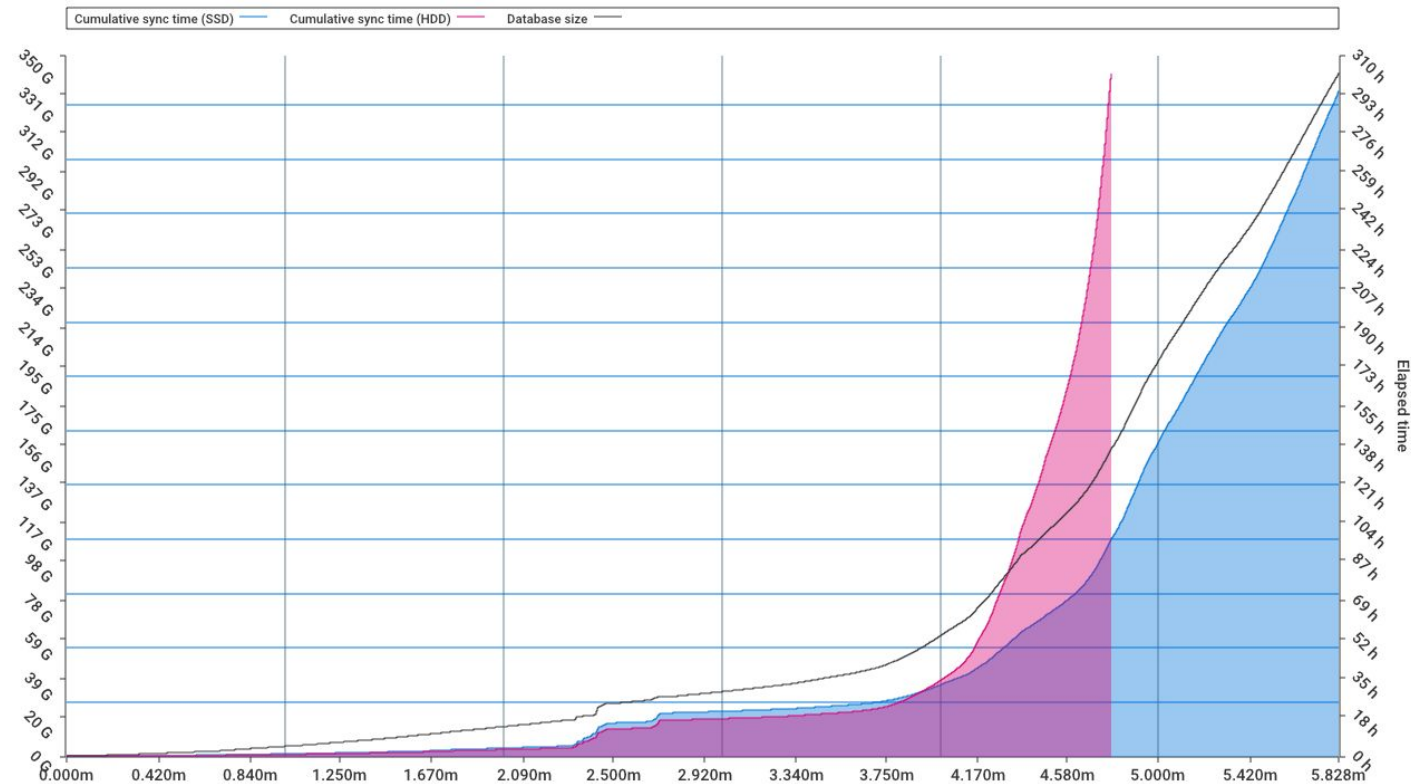
State and Virtual Machine in geth



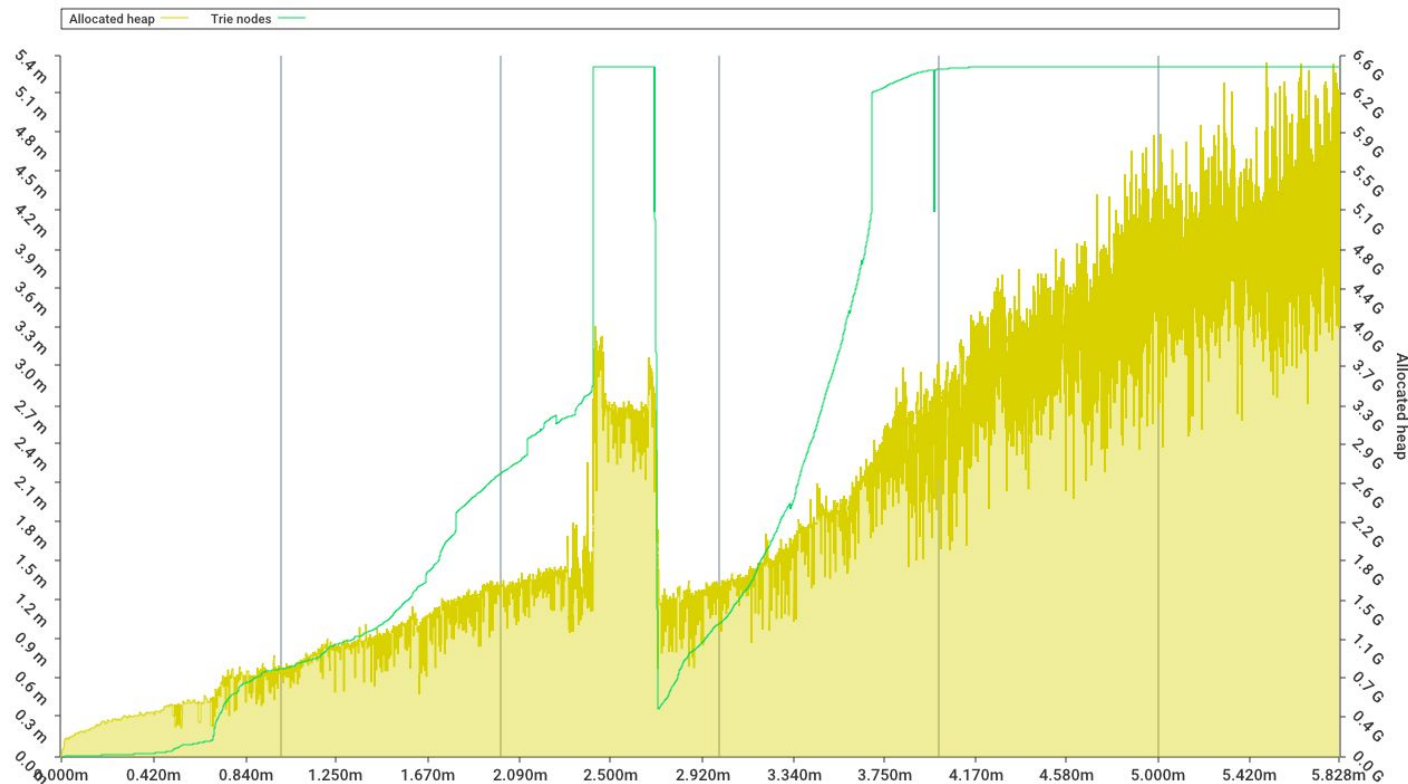
State and Virtual Machine in turbo-geth



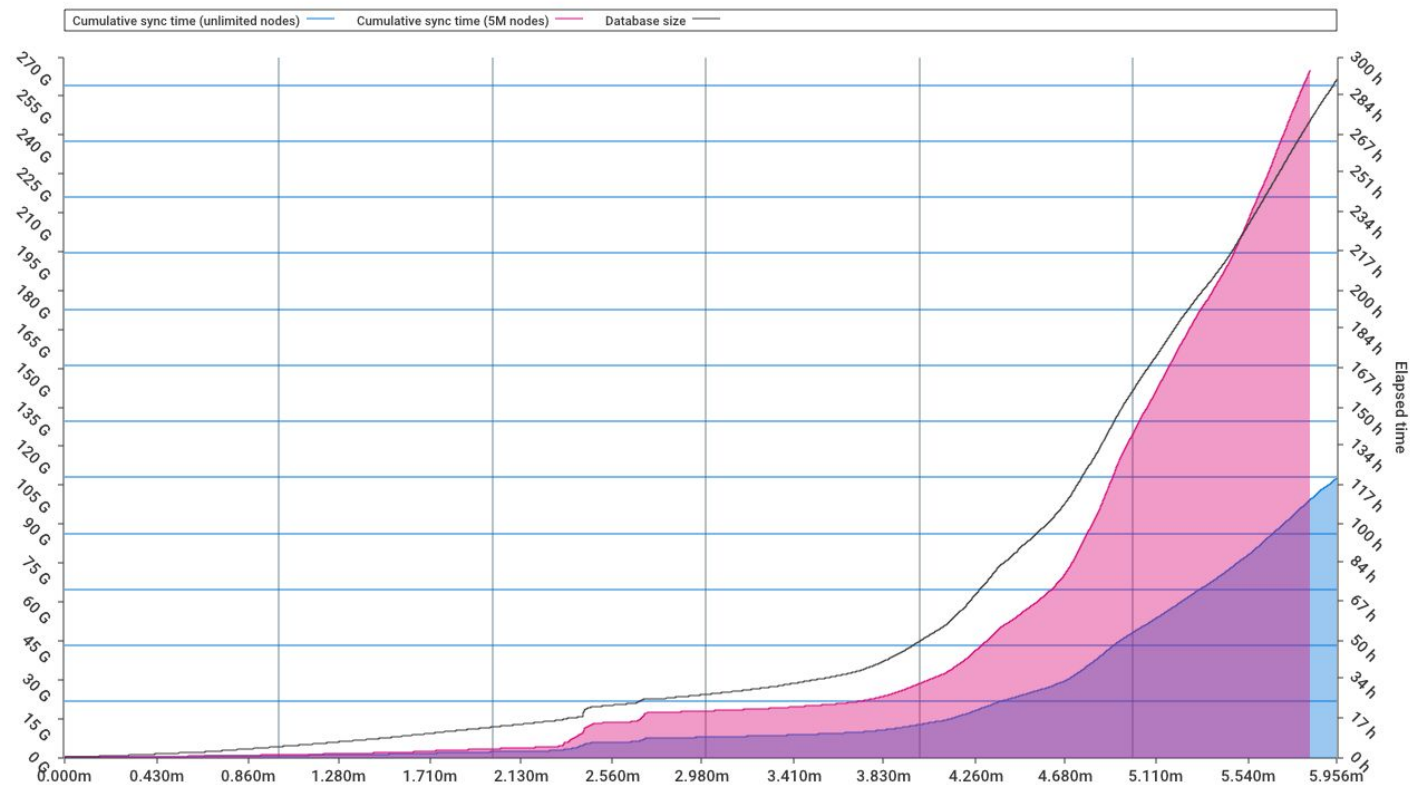
Full sync time SSD (blue) vs HDD (purple)



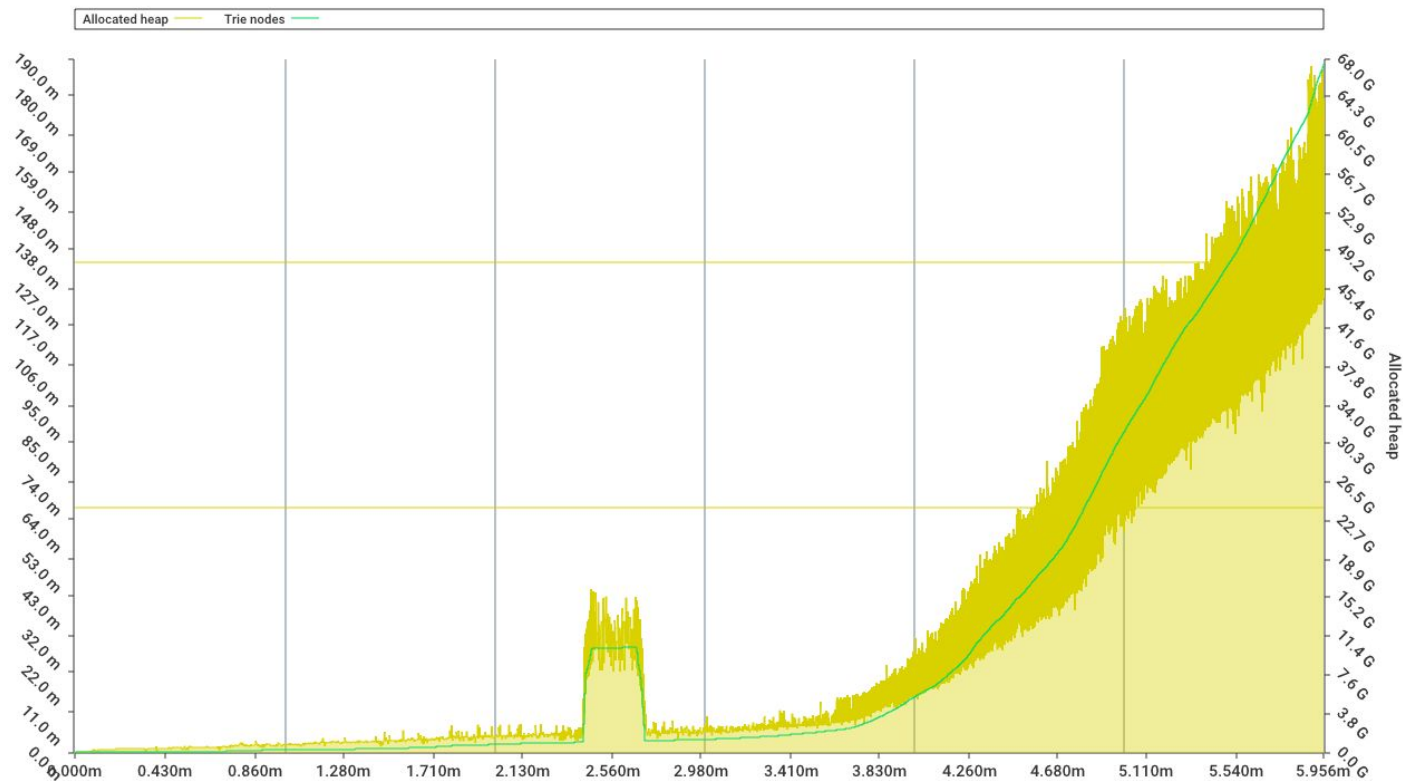
Full sync space (limit 5m trie nodes)



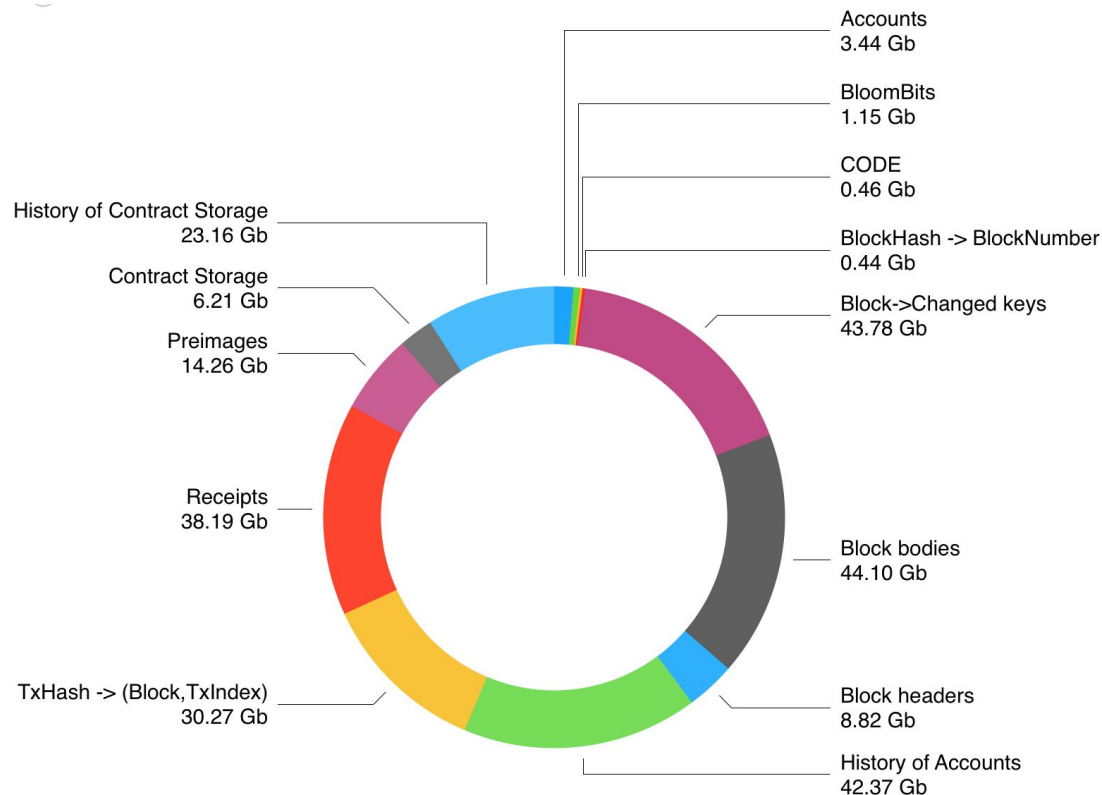
Full sync time 5m nodes (purple) vs unlimited (blue)



Full sync space (unlimited nodes)



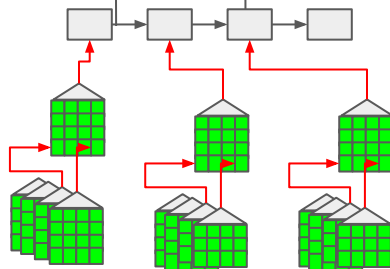
Disk usage (block 5.83m) - total 260 Gb



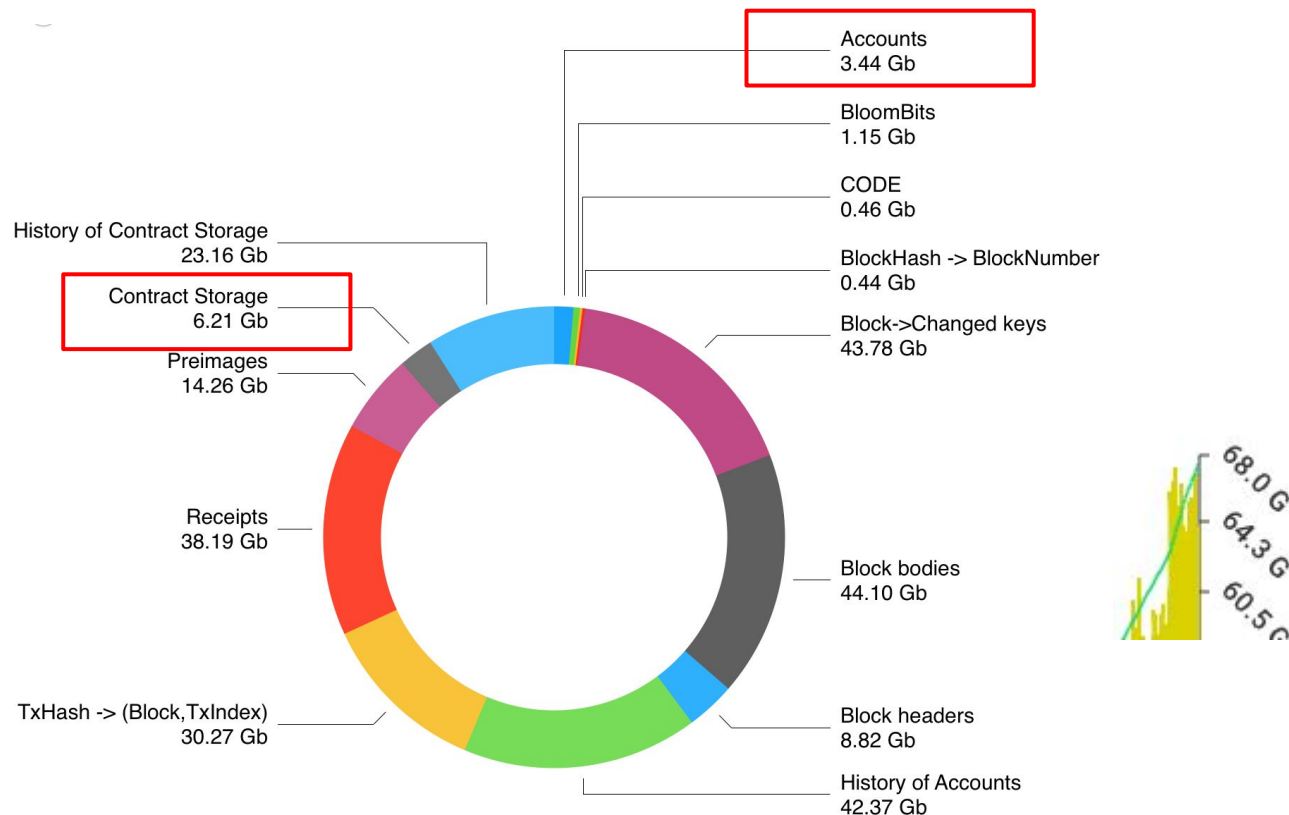
Light clients?

	eth/63	les/2	
Status	✓	✓	Handshake - negotiate version, network_id, genesis
[Get]Block(Headers Bodies)	✓	✓	Get/Send headers/blocks by number or by hash
[Get]NodeData	✓	✓	Get/Send nodes of the patricia tree by hash
[Get]Receipts	✓	✓	Get/Send receipts by transaction hash
NewBlock[Hashes]	✓		Announce new block/block hash
Announce		✓	Announce new chain head
[Get]Proofs		✓	Get/Send merkle proof for given part of trie and block hash
[Get]ContractCode		✓	Get/Send code of given contract at block hash
SendTx	✓	✓	Add new transaction to the pool and relay
[Get]HelperTreeProofs		✓	Get/Send merkle proof of block hash/bloom filters

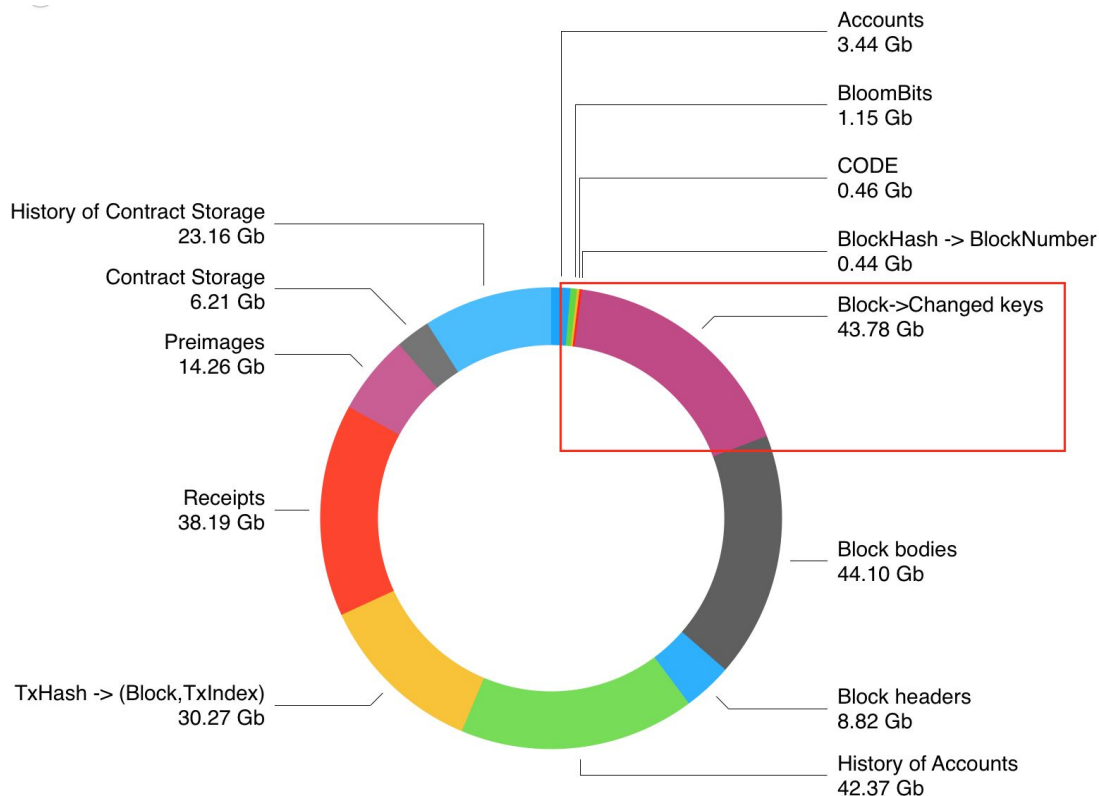
Require materialised
Patricia tree



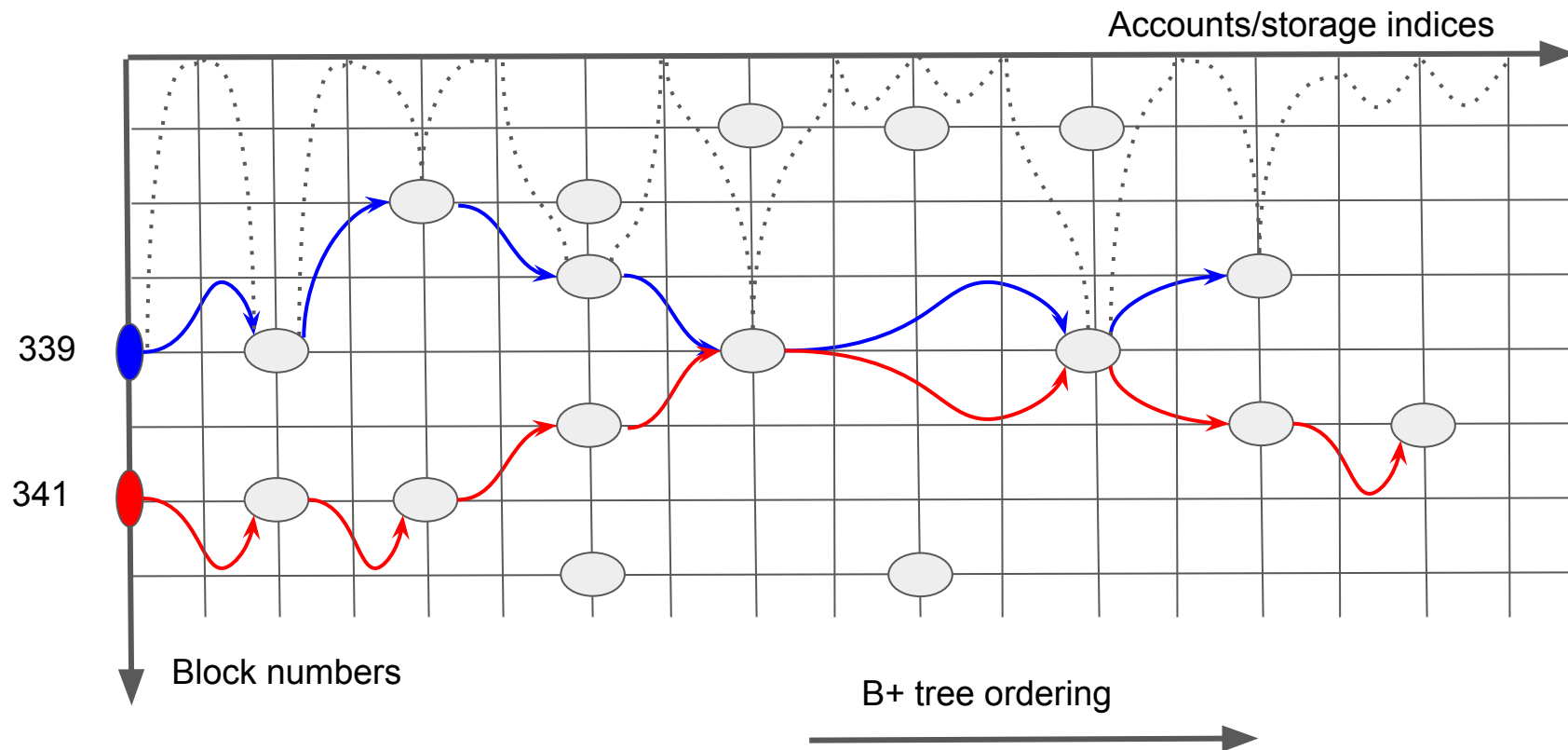
Future experiments - current state in memory



Future experiments - better organised history



Future experiments - better organised history



Some literature on MVBT

An Asymptotically Optimal Multiversion B-Tree

1996

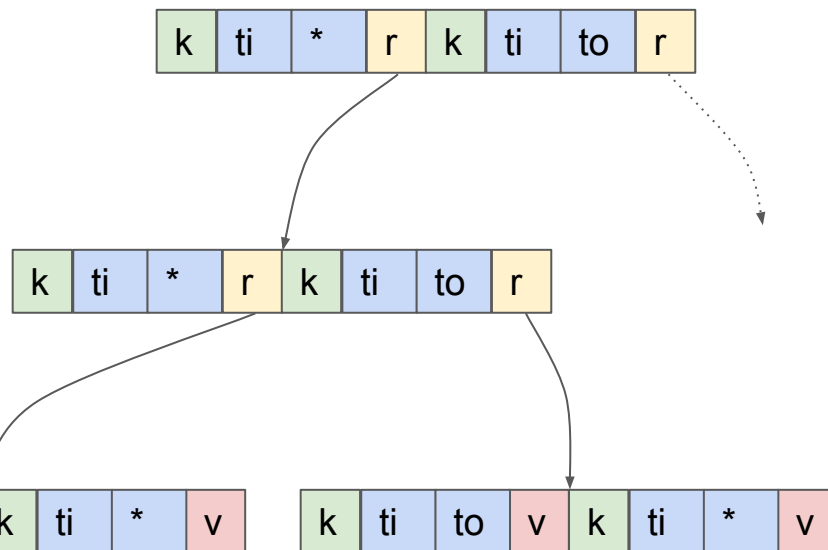
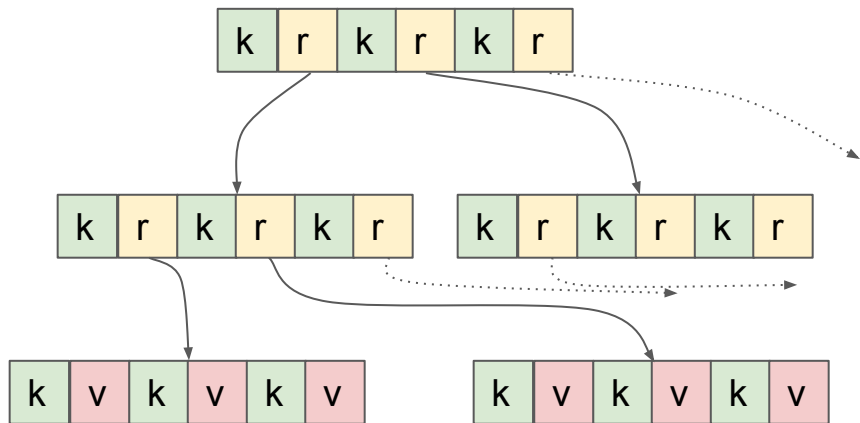
Bruno Becker[†] Stephan Gschwind[†] Thomas Ohler[†]
Bernhard Seeger[‡] Peter Widmayer[§]

Efficient Bulk Updates on Multiversion B-trees

2013

Daniar Achakeev and Bernhard Seeger
Philipps-Universität Marburg
Marburg, Germany
{achakeev,seeger}@mathematik.uni-marburg.de

B-Tree vs MVB-Tree



Status

Reorgs are working - Turbo geth can now follow the tip of mainnet

Next steps:

1. Making all RPC APIs work
2. Release to early adopters
3. Support retesteth, hive
4. Switch database to MVBT