Excel@FIT 2023
http://excel.fit.vutbr.cz

Secure and efficient state preservation in Ethereum based smart contract platforms

Martin Eršek*

Abstract
This paper aims to reduce one of the Ethereum's transaction processing bottlenecks - namely the state preservation. As the state grows, the cost of the state related operation rises, which in turn decrease the Ethereum's potencial to process more transactions per seconds (assuming that there is no consensus level bottleneck). This work aims to solve this problem via change of the approach for storage of ethereum state by designing new custom authentificated dictionary called PMPT. Using this approach the FTPS metric (explained in the paper) decline was significantly reduced. Impact on the TPS has yet to be measured after completly finishing the integration of this PMPT approach to the Geth node implementation. This approach can be applied to any Ethereum based blockchain in order to improve its inherent capabilities, while also introducing the possibility of easier transactions parallel processing instead of the current sequential approach, which can lead to even higher transaction throughput increase.

*xersek00@stud.fit.vutbr.cz, *Faculty of Information Technology, Brno University of Technology*

1. Motivation

As outlined by various papers [1, 2, 3, 4], Ethereum's current approach to its state storage is one of the biggest bottlenecks in its transaction processing. For reference we can have a look at the result of the benchmark performed by the authors of the LMPT [1] paper in the Fig. 3. As can be seen, the amount of accounts in the state trie directly influence ethereum's ability to process transactions (assuming that there is no consensus introduced bottleneck e.g. proof of authority chain or disabled consensus layer). This fact about the ethereum transaction processing together with the fact that LMPT authors were able to improve the ethereum's performance using custom approach for state handling, introduce an opportunity to search for much more efficient solution for storage preservation than the currently integrated one.

2. Storage of Ethereums state

Ethereum hold its global state which is composed of the states of the accounts. Each account state is a mapping from the account address to the account state, this is shown in the Fig. 1. Ethereum can be viewed like a state machine which transit from the old global state to new global state by application of transaction. This global state is stored in the cryptographic authenticated dictionary called Merkle Patricia Trie, as ilustrated in the Fig. 2. As the amount of the accounts in this trie grows, the depth of the trie grows with it logaritmicaly. Since computers RAM memory is limited, accesing leafs of this trie inherently must sooner or later miss the cached state in the RAM and issue costly IO operation on disk storage. This operation leads to huge IO amplification as detaily explained in [2] which in turn result in the results in Fig. 3.

3. Alternative solutions

There have been proposed multiple alternative approached for the state storage. Some of them (RainBlock [3]) require completly reorganization of the nodes in the network and their roles, while others have been only theoreticaly designed but never implemented (mLSM [2]). There are however two approaches worth of our attention:

3.1 LMPT

Layered Merkle Patricia Trie is an approach to separate main ethereum state trie into 3 different tries. This tries are used as "caches" because they are able to be efficiently loaded into RAM memory thus they don't produce any IO overhead, this is illustrated in Fig. 4. Results seems promising, however authors did not published any source code nor any detailed informations about their implementation. The implementation was also made for the now obsolete ethereum client which is no longer functional nor supported by the Ethereum's mainnet network.

3.2 Verkle Trie

Verkle Tries [5] aims to replace the MPT authenticated dictionary with a new authenticated dictionary called Verkle Trie, which is based on Vector Commitments [6]. Utilising this approach the leaf value proofs are much smaller (as ilustrated in Fig. 5) allowing transition to stateless clients. The depth of the Verkle Trie is significantly reduced (so the IO cost is also reduced) since they are 256-ary tries instead of hexary tries as in case of Ethereum's MPT. However, this approach came with its computational complexity cost.

4. PMPT

Parallel Merkle Patricia Trie is our approach to try to alleviate Ethereum's storage problem. First N nibbles of the account address are used as the index to the array of MPTs, the choosen MPT (by the index) is then used to retrieve/store of the account state as ilustrated in Fig. 6. A cryptographic commitment in the form of Merkle Tree is build on top of these tries in order to calculate the final hash which will be used as the commitment to the current state (stateRoot hash). This way, the depth of the accesed tree is reduced by N levels, while this approach also allow easy implementation of concurrent transaction processing (assuming having the access to the transactions access list) without need for shared memory access handling as would be needed in the case of usage of single MPT.

5. Evaluation

PMPT was implemented and integrated into the Ethereum's Geth node. Some part of integrations are still not complemented so not all of the metrics have been measured yet (missing TPS measurement). A toolchain of tools for benchmarking various implementations of Geth have been developed in order to simplify the process of benchmarking of different solutions.

5.1 Metrics

There are multiple metrics measured, namely:

- TPS - transactions per second
- FTPS - filling transactions per second
- FTIME - filling time

Benchmarking toolchain first generate a special file containing required amount of accounts with dummy balance, and then try to fill the state of the private local ethereum chain with these account states. The total time of filling FTIME is measured. Then the filling 'transactions' per second FTPS are calculated as the result of diving the amount of account by the time it takes to fill them into the Geth node. Next a program sending transaction to random new accounts is executed while measuring the average transaction throughput of the Geth node resulting in metric TPS.

TPS metric represent the real transaction throughput, however it contains a lot of overhead time made by the communication with the geth node. FTPS metric don't measure the real transaction throughput rather it measure the amount of account state changes which can be performed during one second.

5.2 Results

The benchmarking of the three different version of Geth node, namely the official release v.1.11.5, verkle trie testnet release and my custom pmpt were performed yielding results show in the figures Fig. 7, 8, 9. The decrease of performance with increasing amount of accounts in the state trie can be observed in all 3 metrics. When examining the FTPS metric, its increase in the first part of the chart is the result of measuring whole time of Geth node running in init phase not the time it took to fill the state trie. As the state trie grows, majority of the whole running time is spend in the process of filling of the state trie, thus yielding 'correct' results.

6. Contribution

In this master thesis, a set of tools for automatic benchmarking of Geth node implementation was developed. These can be subsequently used by the developpers trying to measure the impact of their various optimalizations. A new approach for state storage was proposed, implemented and almost fully integrated (will be completed until the end of thesis) into the Geth node. This can be used for the further experiments or even later for production run on private networks.

Acknowledgements

I would like to thank my supervisor Ivan Homoliak for his help and the original idea to try this approach.

References

[1] Jemin Andrew Choi, Sidi Mohamed Beillahi, Peilun Li, Andreas Veneris, and Fan Long. Lmpts: Eliminating storage bottlenecks for processing blockchain transactions. In *2022 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, pages 1–9, 2022.

[2] Pandian Raju, Soujanya Ponnapalli, Evan Kaminsky, Gilad Oved, Zachary Keener, Vijay Chidambaram, and Ittai Abraham. mlsm: Making authenticated storage faster in ethereum. In *Proceedings of the 10th USENIX Conference on Hot Topics in Storage and File Systems*, HotStorage'18, page 10, USA, 2018. USENIX Association.

[3] Soujanya Ponnapalli, Aashaka Shah, Amy Tai, Souvik Banerjee, Vijay Chidambaram, Dahlia Malkhi, and Michael Yung Chung Wei. Rainblock: Faster transaction processing in public blockchains. In *USENIX Annual Technical Conference*, 2020.

[4] Markus Schäffer, Monika Di Angelo, and Gernot Salzer. *Performance and Scalability of Private Ethereum Blockchains*, pages 103–118. 08 2019.

[5] Vitalik Butherin. Verkle trees. online, 06 2021.

[6] Dario Catalano and Dario Fiore. Vector commitments and their applications. In *Public-Key Cryptography – PKC 2013*, pages 55–72, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.