zkSwap - Scaling Decentralized Exchanges through Transaction Aggregation

Paul Etscheit

March 2021

0.1 Introduction

When being launched in 2015, Ethereum set out to change the way we compute. A trustless, permissionless, and decentralized world computer was envisioned, set to open a new class of applications. The importance of running verifiable code in a permissionless and trustless manner cannot be overstated and enables products and services that are not possible without it. However, the technical limitations have also become apparent quickly. Computations are expensive, theoretical transactions per second are low, and the overall throughput has been stagnant. While Eth2 gives a path towards scaling the network, it is expected to take years to complete.

The first major use case for Ethereum was tokenization. With the development of the ERC-20 standard, launching a token on the Ethereum blockchain was trivial. As tokens run as smart-contracts on the Ethereum blockchain, they are protected by its proof of work consensus, which, given Ethereums PoW hash rate, makes consensus attacks infeasible. This is a significant benefit when looking to tokenize things, as network security can be assumed. While tokenizations are a step in the right direction, they dont come close to the initial vision. While the standerizations enable simple integration into exchanges and wallets, most tokens are isolated in their functionality and ecosystem, and lack productive usage. While smart-contracts where also used to raise money in a somewhat trustless way, most projects used smart-contracts purely for raising money and then tokenizing.

token all isolated, not working together... Not a lot of gas needed blabla

With all of these developments over the past couple of years, it seems we have now entered a new phase of smart-contract use-cases, namely DeFi. While DeFi has a lot of different products and functionalities, at its core aims to utilize tokenized assets in some productive form. Collaterized lending is possible with Aave, yields can be generated by providing assets as liquidity or token trading can be done with uniswap. It can be questioned, how useful or neccecary these protocols really are, but the core idea behind them, is impressive. Rebuilding traditional financial products, running as non-custodial and permissionless smart-contracts, all based on the same standerizations has the potential to reshape the way finance works. With these developments not looking to slow down, they are quickly overwhelming the Ethereum blockchain, pushing transaction costs higher and higher. With the DeFi space moving quickly, this is becoming a real hinderence of innovation and probably the biggest challenge Ethereum is currently facing.

With longer term scaling solutions in development, a number of shorter-term approaches have been proposed. While these do differ, they all aim to move transactional data to a layer-2[1] system, ensuring correctness of that data. One of the approaches is called zk-rollup, the focus of this work. Moving data to a layer-2 system can increase the number of transactions that fit into a block, while also reducing transaction costs for the user. Currently, the most

[1]A layer-2 system is a data-structure that is not on the blockchain but has its state committed to it in some way

used smart-contract is the Uniswap router, which handles all Uniswap trades. To date, it has accured $290m in transaction fees and makes up around 15% of a blocks gas limit. In this work, we will explore how zk-rollups can be used to aggregate Uniswap trades in order to reduce on-chain transactions, while ensuring correctness with a zkSNARK proof.

0.2 Background

good idea for introducing gas as power/efficiency metric

When discussing scaling, gas is a good metric to look at. Every block in Ethereums blockchain has a gas limit that can't be oversteped, so its a good indicator of proccessing power of the network. At the same time, transactions with smart-contracts use an amount of gas for there execution. The amount of gas needed is defined by the opcodes used for that transaction, which have a fixed price. In that sense, a smart-contracts efficiency can be defined by the amount of gas needed for a transaction.

check this, miners should be able to seal blocks still i think

As with all computer systems, scaling can be achived by increasing processing power. Increasing the processing power of a distributed system however, is a complex process. Given that the system is optimized, the only real approach to increasing processing power is sharding. Conceptually, sharding can be seen as upgrading to a multi-core CPU. We seperate the network into individual shards, not requiring every node to proccess every transaction, thereby increasing the processing power of the entire network. Transitioning to a sharded network, namely Eth2, is a highly complex process expected to take years to complete. While this is an inevitable step, aimed at increasing the networks processing power, it wont help in scaling the network for the next couple of years. Shorter term solutions are needed.

Another approach for scaling is increasing a smart-contracts efficiency. As in traditional software applications, reducing the computational cost per operation will in general increase our throughput. In smart-contracts this can be achived by reducing the amount of gas needed for a executing a transaction. As every block has a maximum amount of gas that can be proccessed, reducing it will result in an increased amount of transactions to be proccessed in that block. In smart-contracts data-storage is very expensive. To reduce cost

Bibliography