Arbitrage on Optimism Pools in Uniswap V3

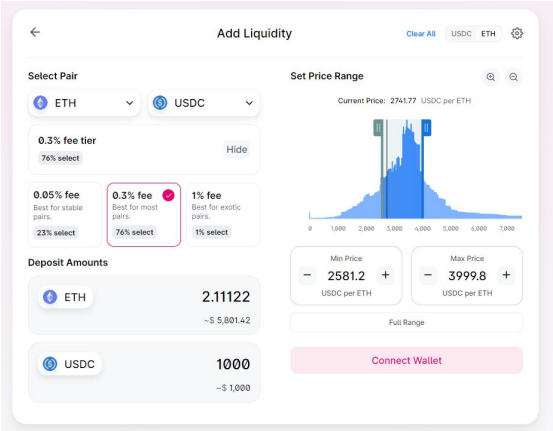
Thomas Mitrevski, Jackie Bai, Emmanuel Abebe, Chris Perez

Motivation and Summary - Jackie

- What our project aimed to do was to take 2 Uniswap pools of the same tokens (i.e., ETH-UNI), in different fee tiers, and look for profitable arbitrage opportunities between the two tokens. In order to facilitate the arbitrage, a deployed flash loan contract would be called.
- Even if we were not able to find any profitable opportunities, we want to be able to learn a lot about Uniswap V3, layer 2 scaling solutions, and flash loans, so that we have the ability to educate others.

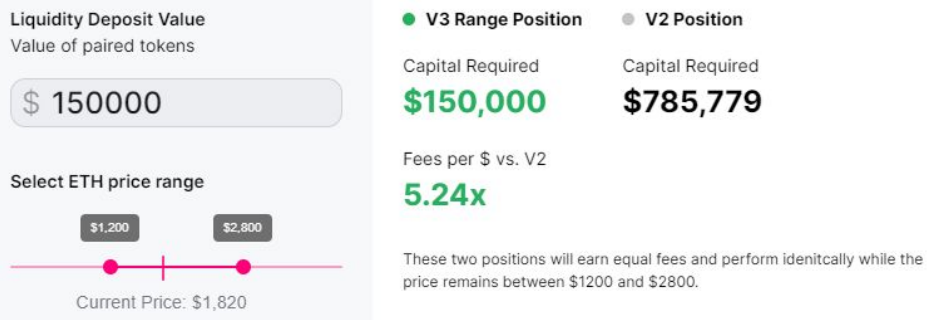
What is a Liquidity Pool? - C

- Is an essential part of an exchange which locks funds in a smart contract to be used for exchanges
- Allows users to swap funds without the use of a centralized market maker
- Individuals can deposit liquidity (two different tokens) into a pool and in exchange, receive fees for swaps that occur on their liquidity.



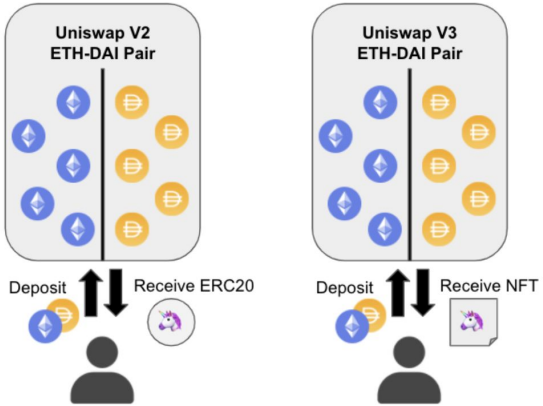
What is Uniswap v3 - E

- DeFi protocol for swapping ERC20 tokens
- Facilitated over $135bn in trading volume, The Largest decentralized exchange
- Provide **Concentrated liquidity**, giving individual LPs granular control over what price ranges their capital is allocated to
- Multiple fee tiers, LPs can appropriately taking on varying degrees of risk
- Up to **4000x Capital Efficiency** relative to Uniswap v2 !
- Users can now choose to provide liquidity between two particular price points, in order to gain capital efficiency.



How Uniswap V3 Works - J

- Users can deposit funds into a liquidity pool in a particular range and receive an NFT that represents their position
- Each position is broken down into individual “ticks” - which represent price ranges
- The price in each tick changes according to a constant product formula.
- Once the price hits a certain value, the price moves into a new tick and the liquidity information is updated in the pool.



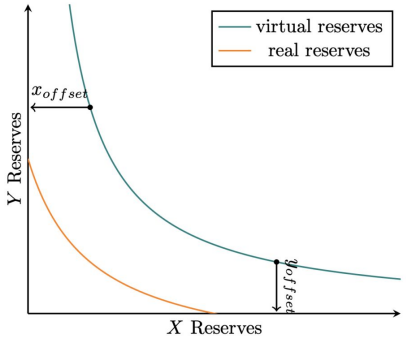
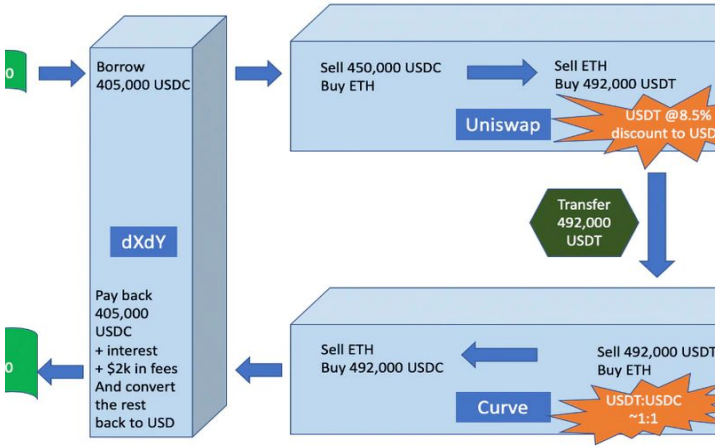


Figure 1: Uniswap v3 reserves in a single position

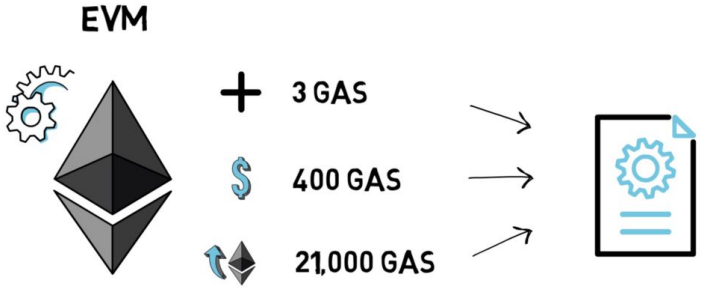
What is an arbitrage opportunity? - C

- In the crypto market they're often price discrepancies in the price of assets(such as ETH being 100 $ in one exchange and ETH being 99$ in another)
- These discrepancies also apply to different liquidity pools of a liquidity pair in uni swap v3
- We can then use the price difference of the pools to commit arbitrage

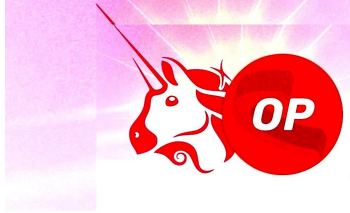


What are some downfalls of ETH Mainnet? - J

- ETH Mainnet can often have high gas prices which add up through every transaction
    - These fees can reduce the opportunities for otherwise profitable trades
- When ETH Mainnet is congested gas prices skyrocket and your transaction might not execute at a profitable time
- ETH Mainnet block times can take on average 13 seconds



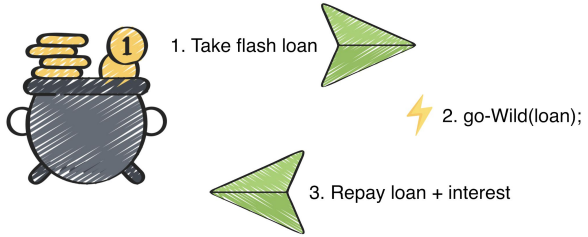
What is a Layer 2 Scaling Solution - C



- A Layer 2 solution is another blockchain built on top of the base chain for the purpose improving scalability (reducing cost and improving transaction time)
- Layer 2 solutions are designed so that gas fees are reduced drastically in comparison to the main chain
- Layer 2 solutions allow more people to use ETH without Mainnet's drawbacks
- Optimism is layer 2 solution that works by having all the computational effort happen in Optimism, which then all the transaction data is put on the main chain
- Optimism allows more transactions per second and decreases the gas cost

Optimism

What are Flash Loans? - E



- This type of loans are unsecured (No collateral needed) lending process in Decentralized finance
- Using smart contracts, loans are distributed and immediately returned to lending pools within the same transaction block
    - Smart Contracts are important because to this process because they provide the condition of the swap and guarantee the trade in an atomic, or all or nothing, manner
- Contracts are meant to be executed immediately
- Flash loans are a great way to take advantage of arbitrage opportunities and make profitable trades without risking your own capital.
- Contracts that are executed are pay for gas fees if the transactions are completed or not, and charge a small fee to execute the transaction.
- borrow large quantities of currencies, making small percentages of profit worth it

How did we implement a Flash Loan contract - T

- Our contract was built using a lot of the provided interfaces and libraries from Uniswap.
- Uniswap has a built in 'flash' function, which when called, calls back to the original contract to execute a callback function with the provided funds.
- We wrote a contract that calls the flash function with the tokens and amounts as parameters, as well as the pools we wish to withdraw and swap with.
- During the callback, we execute our swaps and pay back the pool we borrowed from, along with any accrued fees
- The most important part of this is that we only complete the transaction if it is profitable, this provides us some safety in borrowing a bunch of money and completing a transaction but not being able to fully pay it back.

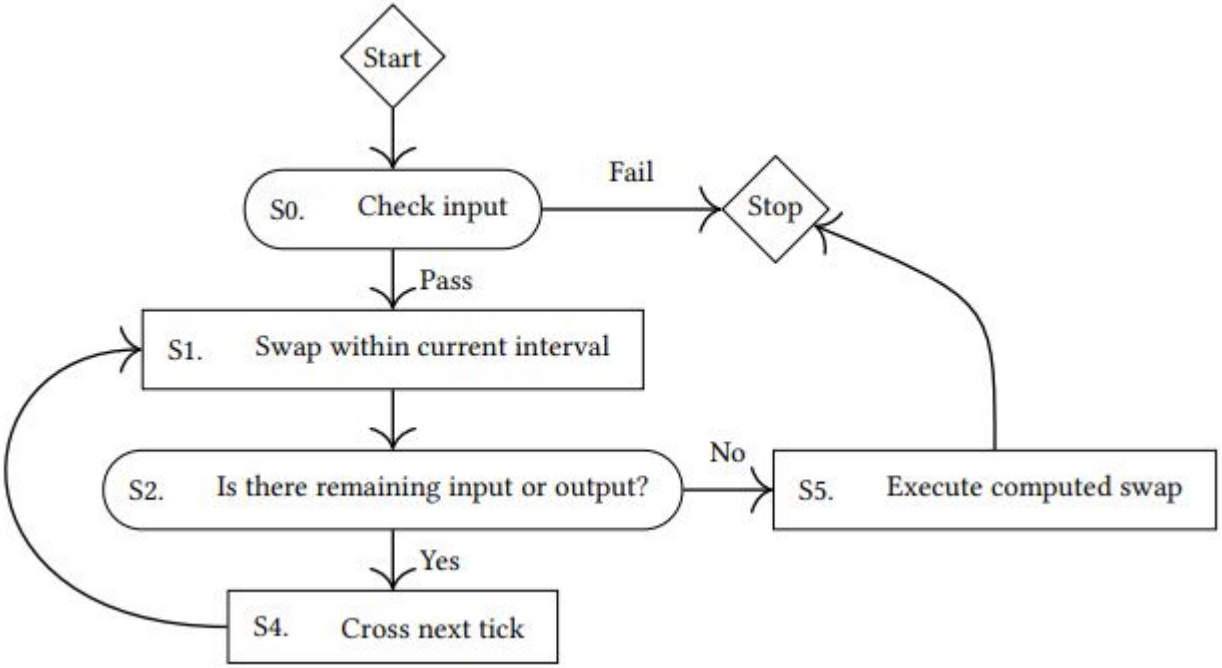
```
if (amountOut0 > amount0Owed) {
    uint256 profit0 = LowGasSafeMath.sub(amountOut0, amount0Owed);

    TransferHelper.safeApprove(token0, address(this), profit0);
    pay(token0, address(this), decoded.payer, profit0);
}
if (amountOut1 > amount1Owed) {
    uint256 profit1 = LowGasSafeMath.sub(amountOut1, amount1Owed);
    TransferHelper.safeApprove(token0, address(this), profit1);
    pay(token1, address(this), decoded.payer, profit1);
}
```

How did we Implement Arbitrage? - T

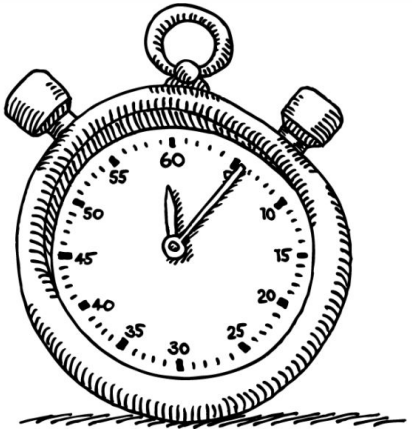
- We developed a Python script that makes API calls to theGraph.
  - This is a Graph Database that contains data about Uniswap from the most recent blocks
- We then grouped this data to identify which pools contained the same coins but had different fee tiers
- We then looked at the liquidity and price data for these pools to determine what price these pools could be arbitraged until.
  - This involved looking at the tick data for each pool, and identifying whether a swap would happen within an individual tick, or across multiple ticks - as each had a different profitability formula
- If a profitable trade was found, we called our flash loan contract in order to execute it

The Process - Chris

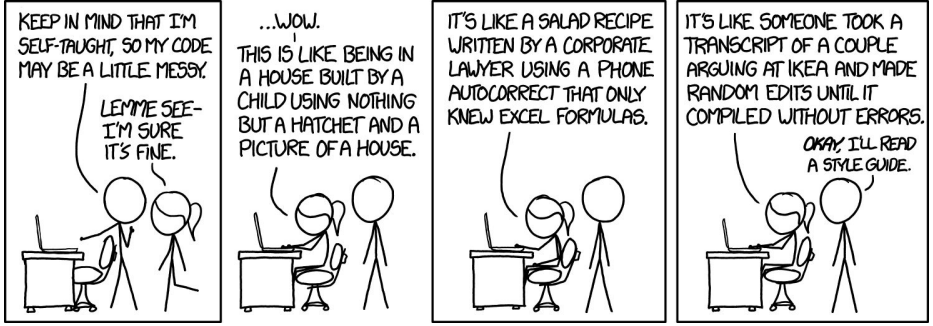


Drawbacks - Emmanuel

- Accuracy is an Opportunity Cost
  - Whether or not your transaction is successful, gas fees must be paid.
  - Price Slippage will be a factor in your transactions and can cause a failure of the Transactions success if not accounted for
- Timing is Extremely Important
  - Timing the trading opportunity is an essential part of the trade. Price can fluctuate in and out of the opportunities realized in a very short time.
  - There is plenty of competition when realizing a trading opportunity.
    - This competition is not exclusively human, there are bots that monitor the mempool and will out bid your gas fees with your exact transaction to beat you to the gains.



What did we find? - T



- Data provided by Uniswap is not high quality
  - Data was provided in really weird units that took a lot of manipulation in order to even comprehend (Why would you provide data as a 160 bit integer that requires bitshift operations and conversions to float in order to use?)
- Understanding how Uniswap works is hard
  - This team underwent HOURS of research and reading in the Uniswap white papers to come up with the correct calculations for profitability
- Deploying on Optimism is difficult
  - Remix's built in Optimism compiler doesn't work very well, so the contracts needed to be built and deployed locally using Hardhat
- Uniswap doesn't keep their code base very well up to date
  - A lot of Uniswap's code doesn't compile very well straight out of the box, as ERC20 and other OpenZeppelin contracts have been updated to use newer compilers, Uniswap's have not
- Not everything that works in Solidity in Ethereum works on Optimism
  - Certain functionality, such as getting "self's" balance, doesn't exist in the Optimistic virtual machine, and workarounds needed to be researched in order to get the code to compile and deploy

Conclusions- T

- Profitable trades are very difficult to find on Optimism. Often, there is not enough liquidity yet in these pools for any price difference to result in a profitable trade.
    - A small amount of volume trade will result in a quick convergence of prices (not enough to make up for gas costs and pool fees.)
- We were able to successfully deploy and test our contract against the Optimistic Kovan network and Uniswap pools
- There were only very few pools with small profitable opportunities, and none that would require enough capital to need to leverage flash swaps
    - However, we did find some trades that could be profitable, up to $40
- In the future, we could look at identifying profitable opportunities on ETH mainnet, or across different DeFi exchanges
- We could also look into trade routing, only using flash swaps when necessary, better inclusion of up-to-date gas costs, and more complex arbitrage opportunities.

	Amount To Send	Pool 0 ID	Pool 1 ID	Profit	Swap From Fee	Swap To Fee	token0	token1
19	0.018603	0xf9ca53854d1ac7adb43d9447aa87f17fe1454e31	0xf2e805fe3b15297e1df03b6036d01b32ab8f7998	2.026756	500	3000	USDC	DAI
25	4.827634	0x00a0e3cd857a7e5676c901bd349ed1d6afb59fb3	0xd515990647c39c4a0b8c03f811f9b746958a0eec	1.721535	10000	3000	LINK	ETH
43	0.016991	0x072611197970d6a9e57680f97f177ff947f09139	0x4cff717ff0b0a4a3578e8bbb7a5f06d32574238b	0.007106	3000	10000	SNX	sUSD