Mooniswap

by 1inch.exchange

Anton Bukov
anton@1inch.exchange

Mikhail Melnik
mikhail@1inch.exchange

August 10, 2020

Abstract

This paper, presents Mooniswap – an AMM[1] that uses the constant-product invariant, with the extension of virtual balances. Unlike other AMM designs, Mooniswap's virtual balances allows to blunt the price impact of the short-term trading volume slippage. This allows profits that would otherwise be captured by arbitrageurs to instead be captured by liquidity providers. This makes Mooniswap significantly more profitable for liquidity providers than most other AMM designs.

1 Introduction

This document will first describe the current state of AMMs on the market today. It will then provide a detailed description of Mooniswap and highlight its key differences from other AMMs.

2 Prior work

The constant-product invariant $x \cdot y = const$ and its generalized version $x_1^{\omega_1} \cdot x_2^{\omega_2} \cdot \ldots \cdot x_n^{\omega_n} = const$ were initially proposed by Alan Lu from Gnosis[1] to create AMMs on Ethereum. Vitalik Buterin also called on[2] developers to run on-chain DEXes[2] using the same formula. Several products were launched using this fundamental design, some with the proposed formulas, others with own custom formulas.

[1]AMM – Automatic Market Maker
[2]DEX – Decentralized Exchange

2.1 Bancor Network

Bancor launched the first-ever AMM in 2017. Bancor pools initially consisted of the BNT token and another ERC20-compatible token. By requiring existence of BNT in every pool, Bancor effectively created a star topology network of tokens, with BNT at the center. Bancor also utilized custom formulas to create weighted pools and achieve effectively infinite instant liquidity of BNT and other "Smart Tokens":

$$price = \frac{ReserveBalance}{SmartTokenTotalSupply \cdot ReserveRatio}$$

2.2 Uniswap Protocol

Uniswap's AMM was launched in November 2018. Uniswap allowed the creation of pools consisting of ETH and any ERC20-compatible token using the constant-product invariant:

$$\begin{aligned} x \cdot y &= const \\ \text{where } x &- \text{balance of ETH} \\ y &- \text{balance ERC20 token} \end{aligned}$$

By using a simple constant-product invariant and using ETH as the central asset, Uniswap was able to create lightweight and gas-efficient contracts.

2.3 Balancer Protocol

Balancer was launched in May 2020. It utilized a generalized constant-product invariant to support multiple tokens with arbitrary weights in a single pool:

$$\begin{aligned} \prod_{i=0}^{n} b_i^{\omega_i} &= const \\ \text{where } b_i &- \text{balance of } i\text{-th asset} \\ \omega_i &- \text{weight of } i\text{-th asset} \end{aligned}$$

While Uniswap naturally serves as a DEX, Balancer Protocol uses the generalized constant-product invariant to offer an automatically rebalancing portfolio. Both Uniswap and Balancer fundamentally use the same constant-product invariant.

2.4 Curve Protocol

Curve Protocol was launched in January 2020. Curve found its initial success as an AMM for mean-reverting assets (such as stable tokens) by introducing its own formula suitable for mean-reverting assets. Curve combines the constant-sum invariant $x + y = const$ with the constant-product invariant $x \cdot y = const$, based on the pool imbalance ratio χ:

$$\chi D^{n-1} \sum b_i + \prod_{i=0}^{n} b_i = \chi D^n + \left(\frac{D}{n}\right)^n$$

$$\begin{aligned} \text{where } b_i &- \text{balance of } i\text{-th asset} \\ D &- \text{sum of all asset balances before the swap} \\ \chi &- \text{imbalance coefficient} \end{aligned}$$

3 How It Works

All existing AMMs use swap fees to earn profits for their liquidity providers. (The swap fees are configurable in Balancer for each pool, whereas Uniswap charges 0.3% and Curve currently charges 0.04% per swap.) Liquidity providers are ultimately compensated via these fees. But if the pricing function significantly misprices the assets in the pool, as might happen after a sudden exogenous price crash, liquidity providers lose potential profit to arbitrageurs who purchase the mispriced assets.

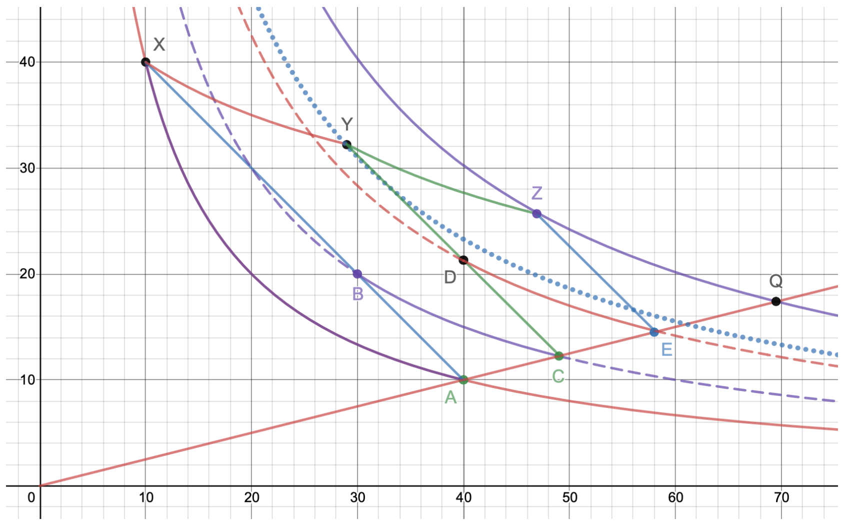
An AMM can thus maximize its profit in one of two ways: maximizing trading fees, or minimizing arbitrageur profits. Mooniswap seeks specifically to pursue the latter strategy: by introducing virtual balances, arbitrageurs are less able to profit on temporarily mispriced pools, leaving more profit for liquidity providers.

All AMMs with constant product pricing functions offer worse slippage as the trade size increases. Other AMMs instantly provide an arbitrage opportunity in the opposite direction after a sufficiently large swap (specifically, if the slippage was larger than the protocol trading fee). Arbitrageurs then compete for these arbitrage opportunities by participating in priority gas auctions[3], paying a significant portion of potential arbitrageur profits to miners. In this model, liquidity providers are not able to capture any of the subsequent profits—aside from the trading fee, all of the revenue from a temporary mispricing is captured by miners and arbitrageurs.

Mooniswap fixes this issue by creating an asymmetry between the two trading directions. Rather than moving both the buy-sell prices simultaneously and offering an immediate arbitrage opportunity, Mooniswap gradually increases the price of the opposite trade. Consequently, the size of the arbitrage opportunity in the opposite direction also increases gradually. This allows the pool to capture some portion of the slippage via further organic trading, rather than giving it all away to the fastest arbitrageur. To achieve that behavior, we have introduced virtual balances that emulate different prices for different swap directions. With virtual balances, purchase of asset A leads the curves for purchasing asset A and asset B to temporarily diverge. The curves eventually converge again over some predetermined time decay. The idea of using virtual balances in AMMs was initially proposed[4] by Vitalik Buterin, to mitigate front-running issues.

3.1 Mathematical Model

In Mooniswap, when a swap takes place, the pool does not immediately offer a profitable trade in the opposite direction. Instead, it slowly improves the price over some period of time. The following chart shows how several trades would significantly increase the constant-product invariant from point X to point Q.



where A – Initial balances,

X – Balances after a swap with significant slippage,

B – Virtual balances for the opposite swap after some period of time,

$\overrightarrow{0A}, \overrightarrow{0Q}$ – Line representing the true exogenous price,

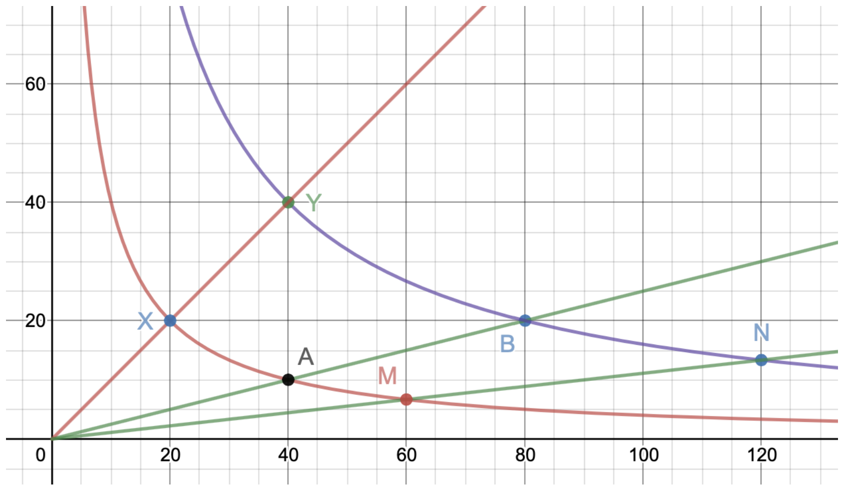
$\overrightarrow{BC}, \overrightarrow{DE}, \overrightarrow{ZQ}$ – Arbitrage trades based on virtual balances,

$\overrightarrow{XY}, \overrightarrow{YZ}, \overrightarrow{ZQ}$ – Parallel translation of arbitrage trades to real balances,

Chart 1. Swap with a huge slippage and 3 subsequent arbitrage trades (source)

After the above swap takes place, the virtual balance for the opposite swap will linearly move from point A to point X. At some point before this full transition takes place, arbitrageurs will attempt to exploit the smaller temporary arbitrage opportunities along the way. For example, when the virtual balance reaches point B, an arbitrageur may choose to arbitrage the price back to the true price at point C. Note that points A and C (and the origin) are located on the same line, which means they have the same price. The chart depicts three sequential arbitrage trades ($\overrightarrow{BC}$, $\overrightarrow{DE}$, $\overrightarrow{ZQ}$) until the real balance reaches an equilibrium price at point Q.

Deposits and withdrawals scale virtual balances in the same proportion as real balances do. The effect of a +100% deposit is shown in the following chart.



where A – Initial real balances,

B – Real balances after deposit,

X, M – Initial virtual balances,

Y, N – Virtual balances after deposit.

Chart 2. Deposits and withdrawals effect on virtual balances (source)

Note that the following equality describes the proportionality between real and virtual balances immediately after deposit or withdrawal:

$$\frac{\overrightarrow{0A}}{\overrightarrow{0B}} = \frac{\overrightarrow{0X}}{\overrightarrow{0Y}} = \frac{\overrightarrow{0M}}{\overrightarrow{0N}}$$

3.2 Backtesting

To estimate potential earnings of Mooniswap model we analysed slippage of several Uniswap V2 markets. Results of the analysis are shown in the chart below. For demonstration purposes we took some of the most liquid stablecoin markets on Uniswap V2.

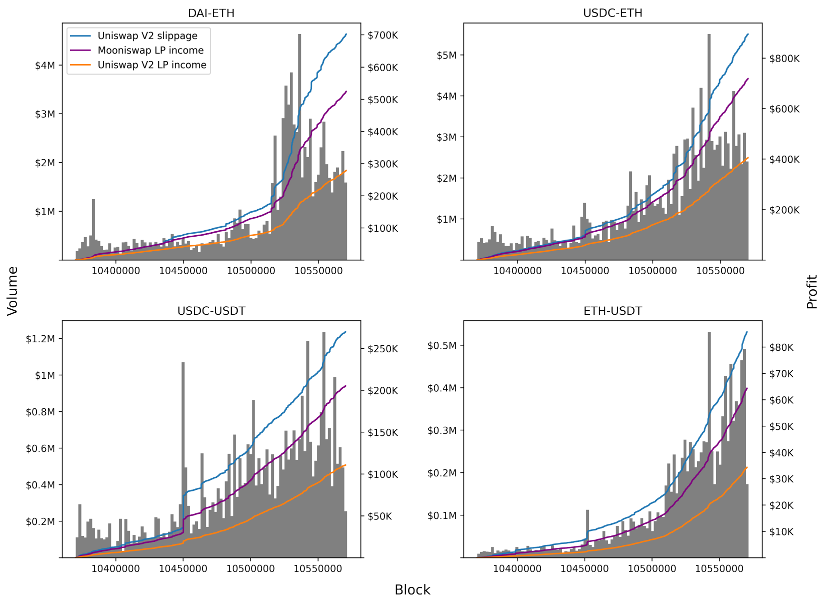


Chart 3. Comparison of potential LP income between Mooniswap and Uniswap V2 on different pools

On average on liquid markets we expect Mooniswap to generate from 50% to 200% more income for liquidity providers than Uniswap V2 due to redirection of price slippage profits.

3.3 Fees

Mooniswap initially utilizes 0.3% swap fee which can be lowered all the way down to 0% in the future as a way to provide more competitive prices to the market.

Mooniswap introduces referral fee to incentivize integrations with wallets, dapps and other services that increase trading volume and provide additional income for liquidity providers. Referral fee does not introduce additional pressure on the exchange rate and rewards external actors who contribute to the protocol by providing external trading volume. Referral fee is only charged when referral wallet is specified in transaction arguments.

Referral fee is fixed and is equal to 5% of income earned by liquidity providers on the trade. So initial 0.3% swap fee will be split into 0.015% going to referral and 0.285% going to liquidity providers. Additional profits acquired due to using virtual balances are also split in the same ratio with 5% going to referral.

Apart from swap fee and referral fee, Mooniswap does not charge any other protocol fees.

3.4 Native ETH support

Mooniswap introduces abstraction layer on top of native ETH asset and ERC20 standard to allow direct usage of native ETH. This abstraction avoids wrapping and unwraping WETH token therefore significantly lowering gas usage on pools with native ETH.

3.5 Price Oracle

Mooniswap also introduces on-chain volume-weighted average price oracles. Price oracle data is stored as a cumulative sum of all trade inputs and outputs in both directions and it is updated after every transaction. By choosing different periods oracle users can configure the required level of price recency and manipulation resistance. We believe that due to Mooniswap's utilization of virtual balances VWAP oracles will be hard to manipulate by malicious actors.

References

[1] Building a decentralized exchange in Ethereum, March 2017. `https://blog.gnosis.pm/building-a-decentralized-exchange-in-ethereum-eea4e7452d6e`.

[2] Let's run on-chain decentralized exchanges the way we run prediction markets, October 2016. `https://www.reddit.com/r/ethereum/comments/55m04x/lets_run_onchain_decentralized_exchanges_the_way/`.

[3] Flash boys 2.0: Frontrunning, transaction reordering, and consensus instability in decentralized exchanges, April 2019. `https://arxiv.org/abs/1904.05234`.

[4] Improving front running resistance of x*y=k market makers, October 2016. `https://ethresear.ch/t/improving-front-running-resistance-of-x-y-k-market-makers/1281`.