SenSwap: A Meshing Protocol for Automated Market Making

Tu Phan
Descartes Network
tuphan@descartes.network

***Abstract*—Based on the idea of constant product market makers, this paper proposes an advanced protocol that reduces pool complexity to $\mathcal{O}(n)$. Theoretically, the protocol will simulate a virtual anchor that plays a role of bridge to pair any tokens and forms a complete market. The system is designed platform-free. This means it possibly works on any blockchains supporting smart contracts. Although there are many such blockchains, Solana is chosen to implement the system because of low fee and short confirmation time. With regard to security, the protocol has a flexible slippage function. In different pairs of token, the curve can be varied. However, they all belongs to the quadratic family that can saturate to infinity which prevents attackers drain off the pools. Otherwise, SenSwap also suffers the impermanent loss similarly to Uniswap. Liquidity providers are required to monitor price changes and act accordingly to minimize the loss. In conclusion, SenSwap has erased the pain of symmetric AMM when the protocol allows single-side liquidity provision, and mesh exchange.**

***Index Terms*—blockchain, decentralized finance, automated market makers**

I. Introduction

In recent years, we are witnessing a great migration of applications from centralized finance (CeFi) to decentralized finance (DeFi). This development is mostly based on the beauty of blockchain that has introduced several attractive properties such as decentralization, transparency, immutability, democracy, and many others. Uniswap, Compound, and Synthetix are typical financial services in the decentralized economy. They respectively implement a decentralized exchange, a decentralized lending & borrowing, and a decentralized derivative, which are three most well-known applications in DeFi [1]. Within scope of this paper, we will discuss about decentralized exchanges.

There are two most popular categories in exchange implementation. One is order books, and the other is automated market makers (AMM). Comparing order-book-based exchanges and AMM-based exchanges, an order-book-based exchange usually maintains a database of orders in both sides bid, and ask to match them, while a AMM-based exchange is supported by function market makers; the function can calculate amount in stock pools and generate corresponding transactions to user's orders. Obviously, AMM-based exchanges can significantly reduce the time of orders matching, increase market liquidity, and open a chance for DeFi applications.

In AMM-based exchanges, stocks in pools are deposited by liquidity providers (LP). The returns for LPs will be an accumulated profit from transaction fees. Each transaction travels through the system will burden a factional amount of fee. The fee is collected and proportionally redistributed to LPs. As a compulsory rule in many common application like Uniswap[1], Curve[2], or Balancer[3], LPs must create a pool of two tokens at least. It means LPs have to deposit a pair of tokens with a predefined rate into a new pool to create a complete market for these tokens. Therefore, it needs $\frac{n(n-1)}{2}$ pools to build a fully complete $n - tokens$ market, which represents $\mathcal{O}(n^2)$ complexity [2].

There exist some rough solutions to alleviate the problem of no direct connection between two tokens. StableSwap, in Curve, proposed a solution in which a pool can include more than two tokens, then reduced complexity to $\mathcal{O}(\frac{n^2}{i^2})$, where $i \in [1..n]$ (see II). Nevertheless, all tokens must be involved in the initial state; it cannot be added more afterwards. Another approach is routing order. The routing tries to find a third shared token, then create a bridge to connect two original tokens. This solution is very simple and straight forward to implement, however, very expensive due to a large fee of multiple hops.

SenSwap is inspired by the idea of routing. Each pool now is a single-side pool which contains a token and an anchor. The anchor is usually defined as a virtual stable token in each pool. Based on this, the protocol computes the value of reserve and the marginal price. Concretely, the value of reserve is identical to the number of anchors. The marginal price is equal to the number of anchors divided by the reserve. These indicators is not only for market evaluation but also for constructing key formulae in SenSwap.

Basically, an AMM must satisfy two intuitive criteria. Firstly, when an LP deposits, the quoted price must be constant over all tokens in all pools. Secondly, when a transaction is executed, the new state must lie on a predefined pricing curve, the constant product function for example. It's trivial to satisfy these in 2-sides pools, but difficult in single-side pools.

II. Related work

The concept of AMM has been studied for many years in academic literature [3]. Instead of human decision makers, some algorithmic makers are created for self-operating in electronic markets. To commence, it's relevant to have a look

[1]https://app.uniswap.org/
[2]https://curve.fi/
[3]https://balancer.exchange/

at bonding curves, a simple mechanism used to define token price based on its supply. The curve can be a linear function, a quadratic function, or even a cube function. In details, giving p is the current price and S is the current supply,

$$p \propto S \qquad \text{(A linear bonding curve)} \tag{1}$$
$$p \propto S^2 \qquad \text{(A quadratic bonding curve)} \tag{2}$$
$$p \propto \frac{1}{3}S^3 \qquad \text{(A cube bonding curve)} \tag{3}$$

Even though bonding curves can serve a price function over supply, it's not observed as a "complete" AMM because bonding curve doesn't examine the relationship between two or more stocks, particularly tokens. Actually, the most famous AMM is the logarithmic market scoring rules (LMSR) [4]. Original applications of LMSR are in prediction markets. One of largest tests of LMSR is Gates Hillman Prediction Market. The test sets a context to predict the opening day of a new computer science buildings, the Gates and Hillman Centers, at Carnegie Mellon University [5]. After that, LMSR started evolving to DeFi, Augur[4], Gnosis[5] for example, which utilized Liquidity-Sensitive LMSR (LS-LMSR) [6], LMSR Primer respectively. In general, the cost function C in the LMSR family is represented as following,

$$C(\overrightarrow{q}) = b \ln \sum_i e^{q_i/b}, \tag{4}$$

where $\overrightarrow{q}$ is the payout vector and $b > 0$ is the liquidity parameter. When b is small, the price will quickly response to the trading, and vice versa. Particularly, in LS-LMSR,

$$b(\overrightarrow{q}) = \alpha \sum_i q_i, \tag{5}$$

where $\alpha > 0$ is a constant.

Beside the variants of LMSR, a Bayesian market maker has introduced a number of advantages over LMSR in binary markets. It offers lower expected loss compared and quickly adapt a market shock [7].

All of the aforementioned approaches are more or less being applied by a number of DeFi platforms over years. However, Uniswap coming was such a game changer. It really created a significant break though for DeFi. Since the moment, not only the number of applications but also the number of users in DeFi has exploded drastically. The main idea behind seems very simple, but it has strongly proved functionality, possibility in both experimental [8] and practical. Uniswap's algorithm, called the constant product function which is a class of constant function market makers (CFMM), formulates the quoted price via the pair of token reserves. Giving token A, and B with corresponding reserves R_A, R_B, the algorithm will maintain the following equation,

[4]https://augur.net/
[5]https://gnosis.io/

$$R_A \times R_B = k, \tag{6}$$

where $k > 0$ is a constant defined at the initial state [2]. In other words, let's call α is the "changing rate" of R_A after an exchange (sell A, buy B for example). Then $R'_A = \frac{1}{\alpha}R_A$ induces $R'_B = \alpha R_B$, where $0 < \alpha < 1$, to maintain the constant product,

$$R_A \times R_B = R'_A \times R'_B = k. \tag{7}$$

At an arbitrary timestamp t, the quoted price $p^{(t)} = R_B^{(t)}/R_A^{(t)}$. Therefore, the first liquidity provider must deposit both tokens, A and B, with reserves that satisfy the reference market price at the initial state, $p^{(0)} = R_B^{(0)}/R_A^{(0)}$. After the pool's setup, subsequent liquidity providers must follow the quoted price by depositing both tokens accordingly. It's a problem when users usually have only one token.

A variant of the constant product function is the constant mean function that has introduced by Balancer. Actually, the function is a weighted geometric mean constant product,

$$\prod_{i=1}^{n} R_i^{\omega_i} = k, \tag{8}$$

where $\omega_i \in \mathbb{R}_+^n$ $(i = 1, ..., n)$ and $n >= 2$ is the number of tokens included in the pool [9]. The constant mean function allows a pool can add n tokens instead of 2 tokens at most like Uniswap. Additionally, the constant mean function is the constant product function when $n = 2$ and $\omega_i = 1, \forall i$.

Consequently, there exists a slippage rate s in both cases because of the constant product.

Definition 1. *For p is the current quoted price, p' is the next quoted price, then a slippage rate,*

$$s = \frac{p'}{p}. \tag{9}$$

Applying to the constant product function in Uniswap and the constant mean function in Balancer,

$$s_{\text{UNISWAP}} = \frac{R'_B/R'_A}{R_B/R_A} = \alpha^2 \tag{10}$$
$$s_{\text{BALANCER}} = \frac{R'^{\omega_B}_B/R'^{\omega_A}_A}{R^{\omega_B}_B/R^{\omega_A}_A} = \alpha^{2\omega_A/\omega_B}. \tag{11}$$

With high slippage rates, these market makers are slightly unsuitable for some relatively stable-priced markets like a pool of two stable tokens. The market needs a less sensitive algorithm. Due to the demand, Curve has introduced Stableswap, a hybrid CFMM, which employs both a constant sum function (CSF) and a constant product function (CPF). This function shows a great feature when it can regulate the slippage rate. It acts like a constant sum at the center of the curve, and shift towards to a constant product at two boundaries [10]. Intuitively, Stableswap connected CSF and CPF via a "leverage" parameter χ,

$$\chi CSF + CPF = k. \tag{12}$$

When $\chi = 0$, the function is obviously identical to a CPF. But when $\chi \to \infty$, the CSF will dominate the whole function and the function becomes a CSF.

In this paper, SenSwap takes care another problem of AMMs, the pool complexity. In general, the protocol introduces the term of anchor in single-side pools. The anchor is supportive to regulate the pools. When there exists liquidity provision cause reserve changing, the quantity of anchor must increase or decrease accordingly. Otherwise, to keep the new state staying on the curve in case of swapping, the quantity of anchor must be unchanged no matter how the reserve varies. It reflects that the price only depends on the scarcity of token. In practice, SenSwap will make use of quick block confirmation time based on Proof-of-History on Solana to alleviate the front-run attack, and low fee for user attraction [11].

III. Core criteria

This section focuses on the aspects of liquidity provision, and exchange which are important to learn main criteria for the meshing protocol. Uniswap will be the object for examination.

A. Liquidity Provision

In Uniswap, a pool contains two tokens, A and B for example. At a timestamp t with the quoted price $p^{(t)}$, a user, who wishes to be an LP, must deposit r_A and $r_B = p^{(t)} r_A$ to the pool. The price before and after deposit is constant; And so withdrawal is, however, r_A is negative.

Remark 1. *The quoted price will be constant with any liquidity provision, r_A and $r_B = \frac{R_B}{R_A} r_A$,*

$$\frac{R_B}{R_A} = \frac{R_B + r_B}{R_A + r_A}. \tag{13}$$

Furthermore, it's clear that reserve of both tokens will be changed after deposit or withdrawal. With every deposits, LPs will receive an amount of share, called liquidity provider token (LPT). Let's assume that $1\ LPT = 1\ USD$, then total value of the pool, in US Dollar, is equal to the sum of LPTs.

B. Exchange

With regard to exchange, the price is not a constant anymore. Instead, it will be changed by following a CPF. Giving a transaction with the changing rate $1/\alpha = R'_A/R_A$ (similarly $\alpha = R'_B/R_B$, $0 < \alpha < 1$), the price curve is a quadratic function, $p' = \alpha^2 p$, which can be rewritten as $\frac{R'_B}{R'_A} = \alpha^2 \frac{R_B}{R_A}$. We also know that the values of pool A and pool B are equal to each other. Therefore, if R_A change by $1/\alpha$, R_B will change by α,

$$\frac{R'_B}{R'_A} = \frac{\alpha R_B}{\frac{1}{\alpha} R_A}. \tag{14}$$

Let LPT_A and LPT_B be the values of pool A and B respectively, then $LPT_A = LPT_B$. We transform both sides of Eq. (14) with p_A and p_B being marginal prices of A and B over USD.

Firstly,

$$\frac{R'_B}{R'_A} = \frac{R'_B}{LPT'_B} \frac{LPT'_A}{R'_A} = \frac{p'_A}{p'_B}. \tag{15}$$

Secondly,

$$\frac{\alpha R_B}{\frac{1}{\alpha} R_A} = \frac{\alpha R_B}{LPT_B} \frac{LPT_A}{\frac{1}{\alpha} R_A} = \frac{\alpha p_A}{\frac{1}{\alpha} p_B}. \tag{16}$$

By connecting these transformation, the obtained equation is equivalent to Eq. (14).

$$\frac{p'_A}{p'_B} = \frac{\alpha p_A}{\frac{1}{\alpha} p_B}. \tag{17}$$

Roughly speaking, if the market demand of A loses $1/\alpha$ $(0 < \alpha < 1)$, the market demand of B will grow up by $1/\alpha$ in terms of price. Additionally, we know that $R'_A = \frac{1}{\alpha} R_A$, $p'_A = \alpha p_A$ (see Eqs (14) and (17)). Therefore, the marginal prices change due to reserve variation while the LPT is fixed.

Remark 2. *The value of pool is unchanged despite any exchanges,*

$$LPT = pR = constant. \tag{18}$$

Here the formula introduces a conservation-of-value paradox. Verbally, the pool A has increased by a number of reserve and the pool B has decreased correspondingly. These likely change the pool value, however, Remark 2 shows the opposite. The rationale is that the exchange, without fee, just changes the pool equilibrium at moment t to another equilibrium at $t + 1$. The pool equilibrium only explains the market demand via the pool price. It won't change the pool value or even trader's total value of balance in theory.

Remark 3. *(Without exchange fees) The slipped values of $r_A = R'_A - R_A$ and $r_B = R_B - R'_B$ are equal to each other.*

$$|R'_A - R_A|_\alpha = |R_B - R'_B|_\beta, \tag{19}$$

where the pair of vertical bars is the operand of token value over the slipped price corresponding to slippage rates in subscripts.

For convenience, we would like to represent $|r_A|_\alpha$ in terms of α. First, we introduce the definition of acceleration of slippage rate.

Proposition 1. *Let a_A denote the acceleration[6] of slipped price of token A over USD. We have:*

[6] Because we are examining an transaction selling A and buying B, then the price p_A will go down and acceleration is negative obviously.

$$a_A = \frac{p'_A - p_A}{R'_A - R_A} = -\alpha \frac{p_A}{R_A}. \quad (20)$$

By the proposition above, we can transform $|r_A|_\alpha$ via a_A definition,

$$\begin{aligned} |r_A|_\alpha &= p_A(R'_A - R_A) + \frac{1}{2}a_A(R'_A - R_A)^2 && (21)\\ &= \frac{1-\alpha}{\alpha}p_A R_A + \frac{1}{2}(-\alpha\frac{p_A}{R_A})(\frac{1-\alpha}{\alpha})^2 R_A^2 && (22)\\ &= \frac{1-\alpha}{\alpha}p_A R_A - \frac{(1-\alpha)^2}{2\alpha}p_A R_A && (23)\\ &= \frac{1-\alpha^2}{2\alpha}p_A R_A. && (24)\end{aligned}$$

Corollary 1. *In the market with a changing rate α, the slipped value of r_A is:*

$$|r_A|_\alpha = \frac{1-\alpha^2}{2\alpha}p_A R_A. \quad (25)$$

All three remarks are fundamental which must be satisfied to maintain the correctness of an AMM.

IV. A single-side pool

Recall that an anchor, LPT, plays a role of a stable token. Because of US dollar is widely accepted as a common monetary unit for stable token in cryptocurrency markets, it's reasonable for us to define one LPT has extract value of one US dollar.

$$1\ LPT = 1\ USD$$

A pool in SenSwap has no longer contained a pair of tokens. Instead, it has only one token. The anchor, LPT, will be paired with the primary token in the pool to realize aforementioned remarks in III.

Liquidity Provision. With respect to Remark 1, the marginal price must be unchanged before and after any liquidity provision. When an LP deposits r_A, he/she will receive back an amount of lpt_A,

$$lpt_A = p_A r_A, \quad (26)$$

where $p_A = LPT_A/R_A$ is the current price of A, LPT_A is the pool value and R_A is the pool reserve.

With withdrawals, the price is a constant too. When an LP burns, he/she will be able to withdraw an amount of token A,

$$r_A = \frac{lpt_A}{p_A}. \quad (27)$$

This equation is reasonable because he/she will withdraw a proportional quantity to other LPs,

$$r_A = \frac{lpt_A}{p_A} = R_A \frac{lpt_A}{LPT_A}. \quad (28)$$

Exchange. Following Remark 2, the pool value must be fixes while exchanging, $LPT'_A = LPT_A$. On the other hand, R_A will be changed by $0 < \alpha < 1$ to $R'_A = \frac{1}{\alpha}R_A$. Therefore,

$$p'_A = \frac{LPT_A}{R'_A} = \frac{LPT_A}{\frac{1}{\alpha}R_A} = \alpha p_A. \quad (29)$$

In case of symmetric pools like Uniswap, the corresponding side, token B, is quite simple to compute when $r_B = R_B - R'_B = (1-\alpha)R_B$. Unfortunately, SenSwap is asymmetric AMM as design. It's more complex to compute the new reserves. For convenience, let β denote the changing rate in pool B corresponding to α. From Remark 3 and Corollary 1, we have:

$$\frac{1-\alpha^2}{2\alpha}p_A R_A = \frac{1-\beta^2}{2\beta}p_B R_B. \quad (30)$$

Let $\lambda = LPT_B/LPT_A$ be the pool ratio, we can shorten the equation to:

$$\frac{1-\alpha^2}{2\alpha}\frac{1}{\lambda} = \frac{1-\beta^2}{2\beta}. \quad (31)$$

Or,

$$\beta^2 + \frac{1-\alpha^2}{\alpha\lambda}\beta - 1 = 0. \quad (32)$$

Solving this quadratic equation, we can find $0 < \beta < 1$ and compute $R'_B = \beta R_B$.

Price Oracle. Arbitrage is the key that SenSwap utilizes to tackle the problem of price. Arbitrage could be considered as actions that instantaneously buy and sell securities, currency, or commodities in two or more different markets in order to take advantage of stock prices. People do arbitrage can be called arbitrageurs (Arbs). In the system, arbitrageurs buy tokens in a low-price market then sell it in another high-price market to make a profit from the difference in value. Because there are always a difference between SenSwap and reference markets, arbitrageurs have motivation to make risk-free profit from markets. It's clear that these actions will help the on-chain prices simultaneously updated to the price in the reference markets.

V. SenSwap analysis

A. Slippage rate

The slippage rates of p_A and p_B are $s_A = 1/\alpha$ and $s_B = \beta$ respectively. Then the ratio s_B/s_A is the slippage rate of the quoted price,

$$s = \frac{s_B}{s_A} = \alpha\beta. \quad (33)$$

By the quadratic formula, we also have:

$$\beta = \frac{\alpha^2 - 1 + \sqrt{(1-\alpha^2)^2 + 4\alpha^2\lambda^2}}{2\alpha\lambda}. \quad (34)$$

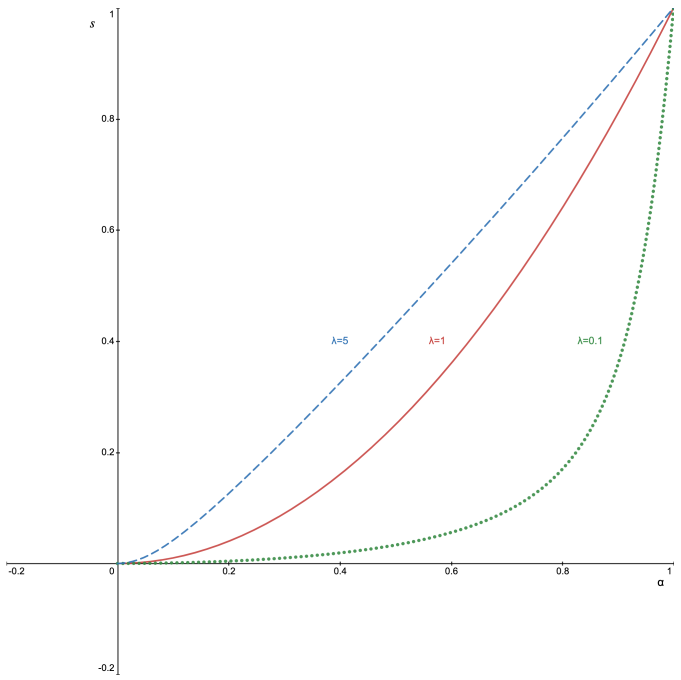

Fig. 1. The continuous line is the quadratic slippage curve of $\lambda = 1$, which is typically used in all Uniswap pools. By contrast, the slippage rates in SenSwap can be very varied depending on the pool ratio λ. The dotted line in SenSwap's slippage curve in case $\lambda = 0.1$, and the dashed line is for $\lambda = 5$.

Then the slippage rate of the quoted price is only dependent on α and λ (see Fig. 1).

Proposition 2. *In a market with $\lambda = LPT_B/LPT_A$, if there are exchanges that change R_A by $0 < \alpha < 1$, the slippage rate of the quoted price will be*

$$s = \frac{\alpha^2 - 1 + \sqrt{(1-\alpha^2)^2 + 4\alpha^2\lambda^2}}{2\lambda}. \tag{35}$$

B. Impermanent loss

The impermanent loss represents the possible loss of LPs compared to HODL[7] due to the divergence in price between deposit and withdrawal. Assume Bob initializes a pool with the initial state $\{R, p\}$. Alice holds a same amount R of token as Bob. After a while, the market state moves to $\{R', p'\}$. The difference in value of token between Alice as a HODL and Bob as an LP is the impermanent loss. For short, we use $loss$ to abbreviate the impermanent loss. Reusing $\alpha = p'/p$ as the changing rate, however, now $\alpha \in \mathbb{R}_+$ and $LPT = pR = p'R'$, the impermanent loss would be shown as:

$$loss = \frac{p'R - p'R'}{pR} = \frac{\alpha pR - p'R'}{pR} = \alpha - 1. \tag{36}$$

It's clear that if the market is in downward trends, becoming an LP gains more profits than HODL. Whereas, the LP must suffer the loss, if the market is in upward trends (see Fig. 2).

[7]A slang in the cryptocurrency community for holding a cryptocurrency rather than selling it

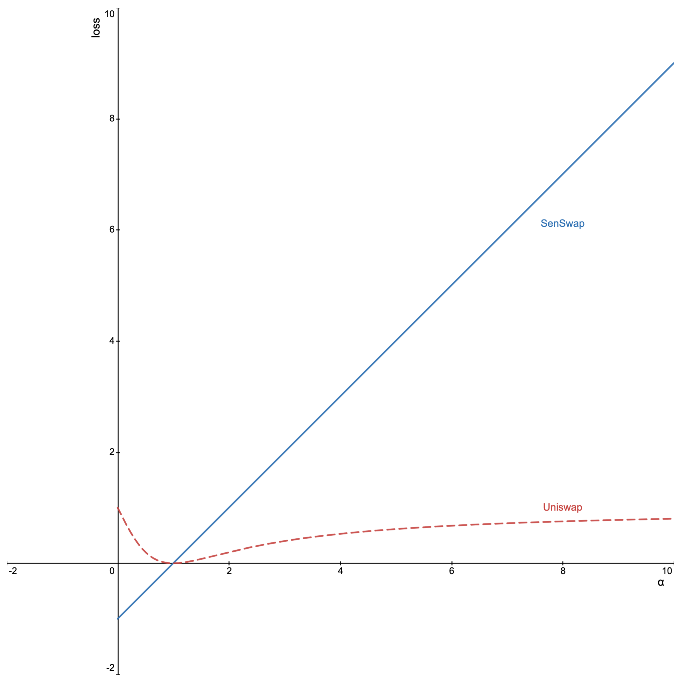

Fig. 2. SenSwap's loss is the continuous line. For example, $0 < \alpha < 1$ showing the decreasing in price causes $loss < 0$. That means LPs gain an advantage over HODL. The impermanent loss in Uniswap would be $loss = 1 - 2\alpha/(\alpha^2+1)$, the dashed line. Note that Uniswap's loss has been modified to adapt convention in this paper.

Proposition 3. *The impermanent loss corresponding to the changing rate $\alpha \in \mathbb{R}_+$ is:*

$$loss = \alpha - 1. \tag{37}$$

C. Universally Synchronous Price

In regard to the constant product function, it's obvious that one token can appear in several pairs. For example, WETH can in WETH-DAI and WETH-USDC , in which DAI and USDC are two stable tokens that have price stably at \$1. When the WETH's price changes, the market need to adjust both pools, WETH-DAI and WETH-USDC, to maintain the correctness of the whole market. This property introduces a remarkable latency in price, which is not good for the price oracle. On the contrary, SenSwap won't face that problem. Because of single-side pools, the meshing protocol can adapt a new price and immediately affect to the whole market. Reusing the aforementioned example, now there are three pools: WETH, DAI, USDC in the market. When WETH changes its price from p to p', it's clear that the quoted price between WETH and the other tokens will change as well,

$$p'_{\text{WETH/DAI}} = \frac{p'}{p_{\text{DAI}}} \tag{38}$$

$$p'_{\text{WETH/USDC}} = \frac{p'}{p_{\text{USDC}}}. \tag{39}$$

With the meshing protocol, there is no more room for arbitrageurs to exploit the swap in market shocks.

VI. Conclusions

The paper has proposed a meshing protocol using the single-side AMM. It significantly reduces the pool complexity to $\mathcal{O}(n)$. Now LPs can do deposit and withdrawal by a single token, users can pair any tokens to swap without the restriction of existence. This key can be an technical element that attracts flows of liquidity, the most critical thing for an AMM application's success.

References

[1] D. Lau, D. Lau, and S. Teh, *How to DeFi*. Independently Published, 2020. [Online]. Available: https://books.google.nl/books?id=g6WazQEACAAJ

[2] H. Adams, "Uniswap," *URl: https://docs. uniswap. io/. 13One possibility is that the parties could treat the option itself as the underlying asset, and constantly update their payment channel based on the option's current computed value*, 2019.

[3] A. M. Othman, "Automated market making: Theory and practice," 2012.

[4] R. Hanson, "Logarithmic markets coring rules for modular combinatorial information aggregation," *The Journal of Prediction Markets*, vol. 1, no. 1, pp. 3–15, 2007.

[5] A. Othman and T. Sandholm, "Automated market-making in the large: the gates hillman prediction market," in *Proceedings of the 11th ACM conference on Electronic commerce*, 2010, pp. 367–376.

[6] A. Othman, D. M. Pennock, D. M. Reeves, and T. Sandholm, "A practical liquidity-sensitive automated market maker," *ACM Transactions on Economics and Computation (TEAC)*, vol. 1, no. 3, pp. 1–25, 2013.

[7] A. Brahma, M. Chakraborty, S. Das, A. Lavoie, and M. Magdon-Ismail, "A bayesian market maker," in *Proceedings of the 13th ACM Conference on Electronic Commerce*, 2012, pp. 215–232.

[8] G. Angeris, H.-T. Kao, R. Chiang, C. Noyes, and T. Chitra, "An analysis of uniswap markets," *Cryptoeconomic Systems Journal*, 2019.

[9] F. Martinelli and N. Mushegian, "A non-custodial portfolio manager, liquidity provider, and price sensor," *URl: https://balancer. finance/whitepaper*, 2019.

[10] M. Egorov, "Stableswap-efficient mechanism for stablecoin liquidity," *URl: https://medium. com/yield-protocol/introducing-ydai-43a727b96fc7*, 2019.

[11] A. Yakovenko, "Solana: A new architecture for a high performance blockchain," 2018.