DEEPER: Enhancing Liquidity in Concentrated Liquidity AMM DEX via Sharing

Srisht Fateh Singh
Electrical & Computer Engineering
University of Toronto
Toronto, Canada
srishtfateh.singh@mail.utoronto.ca

Panagiotis Michalopoulos
Electrical & Computer Engineering
University of Toronto
Toronto, Canada
p.michalopoulos@mail.utoronto.ca

Andreas Veneris
Electrical & Computer Engineering
University of Toronto
Toronto, Canada
veneris@eecg.toronto.edu

2023 IEEE International Conference on Blockchain and Cryptocurrency (ICBC) | 979-8-3503-1019-1/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICBC56567.2023.10174969

***Abstract*—This paper presents DEEPER, a design for a decentralized exchange that enhances the average active liquidity via reserve sharing. By doing this, it addresses the problem of shallow liquidity in low trading volume token pairs. DEEPER allows liquidity providers of multiple trading pairs against a common token to share liquidity. This is achieved by creating a common reserve pool for the shared token that is accessible by each trading pair. Independent from the shared liquidity, providers are free to add liquidity to individual token pairs without any restriction. The trading between one token pair does not affect the price of other token pairs even though the reserve of the shared token changes. The proposed design is an extension of concentrated liquidity market maker-based DEXs that is simple enough to be implemented on smart contracts. Experiments show that for a batch consisting of 8 trading pairs, DEEPER enhances liquidity by over $2.6-5.9\times$. This enhancement in liquidity can be increased further by increasing participating tokens in the shared pool.**

***Index Terms*—AMM, DEX, concentrated liquidity, market maker, tokens, cryptocurrency**

I. INTRODUCTION

Following the success of Ethereum [1], blockchains are fostering the promise of decentralization by removing central points of failure and bringing trustlessness in traditional technology sectors. This includes revolutionizing Financial Technology (FinTech) with Decentralized Finance (DeFi) [2], research platforms with decentralized science [3], gaming with game finance [4], web services [5], supply chains [6], and social engagement platforms [7]. A key aspect in the design of a decentralized model is an economic incentive that is enabled using cryptocurrency tokens [8]. Apart from incentives, tokens can provide utility in web applications, governance rights via voting, or tokenization of real-world assets [9].

These tokens can be traded on a Centralized Exchange (CEX) that adopts the traditional Central Limit Order Book (CLOB) mechanism or a Decentralized Exchange (DEX) running Automated Market Making (AMM) algorithms. In AMMs, liquidity providers (LPs) deposit a pair of assets that other traders can swap. For every swap, the trader pays a fee proportional to the swap amount that goes to the LPs. Traders prefer an asset pair with significant liquidity because otherwise, it can lead to unconventional price movements (also known as "slippage") where the trading pair's price becomes vulnerable to manipulation. At the same time, LPs are not incentivized to provide significant liquidity when the trading volume of a pair is very low since fewer fees will be distributed to them. Therefore, a new design of a marketplace is needed where tokens that have a low trading volume also enjoy significant liquidity without incurring additional costs to the LPs to acquire more token reserves.

This paper presents DEEPER, a design for a decentralized exchange that allows LPs of multiple tokens against a common currency to assemble and share their liquidity for that common currency. Unlike other multi-token pool platforms like Balancer [10], DEEPER remains a sovereign AMM DEX and does not depend on arbitrageurs to adjust the price of a token when a trade is made between other tokens in the pool. DEEPER extends a concentrated liquidity AMM with the functionality of shared reserve for the common currency in a batch of multiple trading pairs. The shared reserve access mechanism ensures that the reserves in the shared pool never go negative. Sharing currency reserves is optional and does not prevent LPs to provide concentrated liquidity for individual trading pairs. Lastly, the design of DEEPER is kept simple enough making it practical to implement it on EVM-based smart contracts.

Experiments presented here on historic price movements of low trading volume tokens show that DEEPER can enhance the liquidity for a trading pair in a batch of 8 assets by a factor of up to $5.9\times$. In doing so, LPs do not need to provide any additional reserves of tokens compared to the contemporary AMM designs. This increased liquidity can be enhanced further by increasing the number of pooled tokens in a batch or by frequently updating liquidity profiles.

The paper is organized as follows: Section II gives preliminaries on AMMs and concentrated liquidity, Section III describes the problem statement and gives an overview of the solution, Section IV presents the design of DEEPER DEX and comments on divergence loss for LPs and finally, the paper is concluded in Section VI.

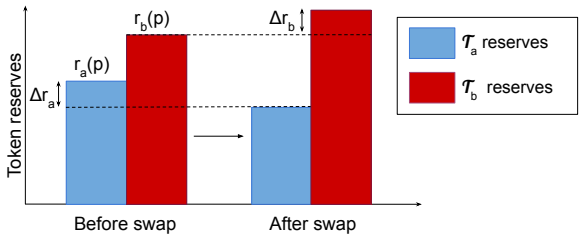
II. BACKGROUND

A. Constant Product Automated Market Makers (CPAMMs)

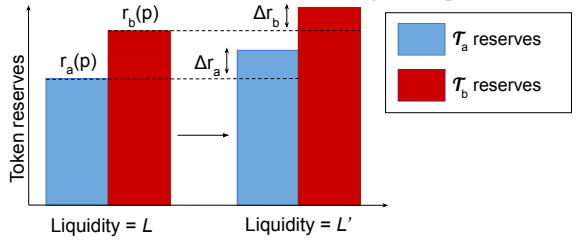
CPAMM was the first successful algorithmic market maker introduced by the Uniswap DEX [11], which is

979-8-3503-1019-1/23/$31.00 ©2023 IEEE

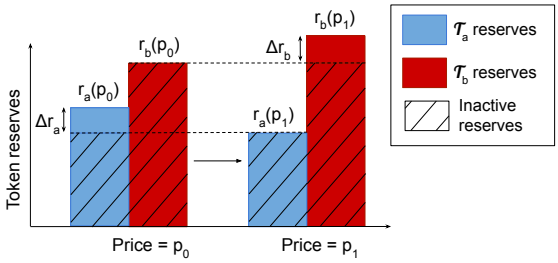
Authorized licensed use limited to: University College London. Downloaded on June 28,2024 at 00:17:07 UTC from IEEE Xplore. Restrictions apply.



(a) Token reserves during a swap



(b) Token reserves during liquidity provision



(c) Token reserves in concentrated liquidity

Fig. 1: (a) and (b) illustrate token reserves during a swap and liquidity provision respectively in CPAMM. (c) illustrates real token reserves during a swap in concentrated liquidity AMM.

governed by the *constant product formula*. In a nutshell, consider a trading pair consisting of tokens $\mathcal{T}_a$ and $\mathcal{T}_b$ such that the price of $\mathcal{T}_a$ with respect to $\mathcal{T}_b$ is p. Then, the liquidity pool of the above pair has active token reserves as a function of price comprising $r_a(p)$, and $r_b(p)$ units of $\mathcal{T}_a$ and $\mathcal{T}_b$ respectively, such that $r_a(p)r_b(p) = L^2$, where L does not depend on p. The constant L is called the *liquidity* of the pool. As derived in [12], the marginal price p turns out to be the ratio of the current token reserves *i.e.* $p = \frac{r_b(p)}{r_a(p)}$. This can be interpreted as the equivalent amount of $\mathcal{T}_b$ per unit amount of $\mathcal{T}_a$ in the reserves. The expression for token reserves can therefore be derived as follows:

$$\begin{aligned} r_a(p) &= \sqrt{r_a(p)r_b(p)}\sqrt{\frac{r_a(p)}{r_b(p)}} = \frac{L}{\sqrt{p}} \\ r_b(p) &= \sqrt{r_a(p)r_b(p)}\sqrt{\frac{r_b(p)}{r_a(p)}} = L\sqrt{p} \end{aligned} \tag{1}$$

When a trader swaps Δr_a units of $\mathcal{T}_a$ at price p, then $(1-\mu)\Delta r_a$ is collected as *liquidity fees* for LPs where $\mu \in [0,1]$ and is set close to 1. In return, the trader receives Δr_b units of $\mathcal{T}_b$ following the constant product rule, *i.e.*

$$(r_a(p) + \mu\Delta r_a)(r_b(p) - \Delta r_b) = L^2 \tag{2}$$

An illustration of token reserves before and after the swap for $\mu = 1$ is shown in Figure 1a.

Furthermore, LPs can alter liquidity by adding or removing token reserves. If an LP provides Δr_a and Δr_b of $\mathcal{T}_a$ and $\mathcal{T}_b$ respectively at price p, then the following needs to hold:

$$\frac{\Delta r_a}{\Delta r_b} = \frac{r_a(p)}{r_b(p)} \tag{3}$$

Here Δr_a, Δr_b should either be both positive (mint) or negative (withdraw). The new liquidity L' can now be calculated as $(r_a(p) + \Delta r_a)(r_b(p) + \Delta r_b) = (L')^2$. Figure 1b gives an illustration of token reserves when liquidity increases in the pool. Although in CPAMM liquidity does not change with the price of the tokens, this is not the case in concentrated liquidity AMM as presented next.

B. Concentrated Liquidity AMM (CLAMM)

Consider Figure 1c where the price of $\mathcal{T}_a$ increases from p_0 to p_1, $\mathcal{T}_a$ reserves reduce by Δr_a while $\mathcal{T}_b$ reserves increase by Δr_b. In this price interval, only Δr_a of $\mathcal{T}_a$ and Δr_b of $\mathcal{T}_b$ are actively swapped while the rest of the reserves (marked with dashes) remain inactive. This is the key idea behind CLAMMs where LPs can provide liquidity in a price interval $[p_0, p_1]$ by only supplying Δr_a units of $\mathcal{T}_a$ and Δr_b units of $\mathcal{T}_b$. Meanwhile, the liquidity in CLAMM is the same as CPAMM, i.e. $L^2 = r_a(p)r_b(p)$. When the price is outside the above price interval, the liquidity becomes inactive. Hence in a CLAMM, $r_a(p), r_b(p)$ are called *virtual reserves*. The real reserves of $\mathcal{T}_a, \mathcal{T}_b$ at price $p \in [p_0, p_1]$, denoted by $r'_a(p), r'_b(p)$ can therefore be calculated in terms of virtual reserves as follows:

$$\begin{aligned} r'_a(p) &= r_a(p) - r_a(p_1) = L\left(\frac{1}{\sqrt{p}} - \frac{1}{\sqrt{p_1}}\right) \\ r'_b(p) &= r_b(p) - r_b(p_0) = L(\sqrt{p} - \sqrt{p_0}) \end{aligned} \tag{4}$$

We used the result from Equation 1 to derive the expressions for virtual reserves in terms of liquidity and price. Observe that at the boundaries of the interval, i.e. $p \in \{p_0, p_1\}$, the real reserves consist of either only $\mathcal{T}_a$ or $\mathcal{T}_b$.

L remains constant within a price interval but can vary across intervals. Concentrated liquidity, therefore, allows LPs to add arbitrary liquidity in different price intervals. Figure 2 shows an example liquidity profile by an LP across price intervals. The liquidity at price p is denoted by $L(p)$. Figure 3 presents the corresponding real reserves provided for each price interval. The active price interval is marked with circled ticks and consists of both $\mathcal{T}_a$ and $\mathcal{T}_b$ reserves. The inactive price intervals consist of only $\mathcal{T}_a$ or $\mathcal{T}_b$ for prices higher or lower than the current price.

Authorized licensed use limited to: University College London. Downloaded on June 28,2024 at 00:17:07 UTC from IEEE Xplore. Restrictions apply.

Fig. 2: Liquidity distribution during interval T.

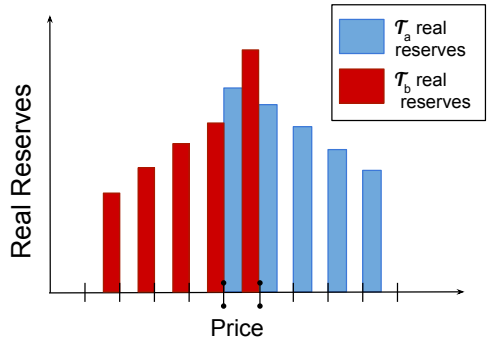


Fig. 3: Illustration of real reserves distribution of an LP in concentrated liquidity DEX.

III. Work Motivation

A. Capital Efficiency and Fair Markets

As mentioned in Equation 2, the revenue of an LP comes from the trading fee that is charged as a constant fraction of the tokens swapped. Problems arise when the trading volume of a token pair $\mathcal{T}/\mathcal{T}_c$ ($\mathcal{T}_c$ is a highly liquid currency) is orders of magnitude lower than other trading pairs. This can be caused due to the following reasons: $\mathcal{T}$ is a newly launched token and its underlying utility has not gained traction among users; $\mathcal{T}$ has a seasonal utility (*e.g.*, a DAO token used for voting in a protocol [13], an access token for a real-world event [7], or a football club fan token); the overall market has low liquidity due to the high cost of acquiring capital, *i.e.* high borrowing rates; $\mathcal{T}$ targets a small niche of users (*e.g.*, $\mathcal{T}$ represents an LP token of a Uniswap V2 pool).

One or more of the above conditions leads to a lower trading volume of the $\mathcal{T}/\mathcal{T}_c$ pair that causes the following cascading consequences:

1) As lower trading volume leads to lower LP fees, this reduces the incentive for an LP to participate. Moreover, this exposes them to the risk of an overall loss if their impermanent loss (as discussed later in Section IV-D) dominates the trading fees. If the trading fee is set high enough to increase the LP's incentives, it discourages the token users, and traders including arbitrageurs to trade $\mathcal{T}$.
2) If the $\mathcal{T}/\mathcal{T}_c$ pair ends up with low liquidity, then it is subject to unfavorable economic events of high volatility and high slippage. Moreover, low liquidity also makes a trading pair vulnerable to price manipulations since low capital is now required to manipulate reserve ratios and hence the prices. This also gives room to DeFi attacks such as sandwich attacks [14] where an attacker sandwiches a buy order for $\mathcal{T}$ with a large buy and sell order respectively of equal amounts and the swapper ends up paying a higher price for their trade. These consequences can seriously damage the integrity of a platform whose operation relies on the fairness of the token price, for example, when $\mathcal{T}$ represents a DAO or a voting token.
3) Despite their exposure to impermanent loss, if an LP provides deep liquidity to a low-volume trading pair, this approach is not capital efficient. This is because the liquidity capital stays idle for the majority of the time and suffers opportunity costs.

Expanding on the last point, with the explosion of DeFi and other ecosystems in blockchain and the reduced trust in CEXs, it has become crucial for on-chain DEXs to sustain a fair market for thousands of token pairs [15]. This requires a mechanism that enhances the liquidity profile while consuming low input capital (*i.e.*, real reserves). Furthermore, the solution design should be easily implementable on common blockchain environments such as the Ethereum virtual machine or EVM.

B. Evaluation Metric

In this paper, we consider N trading pairs of tokens $\mathcal{T}_0, \mathcal{T}_1 \ldots \mathcal{T}_{N-1}$ against a digital currency $\mathcal{T}_c$ with high liquidity (*e.g.*, ETH). We profile the liquidity on an AMM and the price of these trading pairs in a constant time interval I. Such a time interval gives a finite range of prices that are processed. The price of the token at time t is denoted by $p(t)$. The price of each $\mathcal{T}_i$ is divided into intervals of uniform width. We assume that during I, no LP mints or withdraws their liquidity.

Let $L_i(p(t))$ denote the liquidity profile for $\mathcal{T}_i$ at price $p(t)$ at a given instance t. At $t = 0$, the total real reserves of $\mathcal{T}_i, \mathcal{T}_c$ for the pair i are denoted by $\mathbf{r'_i}, \mathbf{r'_{i,c}}$ respectively. Suppose there are two LPs with their proposed liquidity profiles $L_i(p)$ and $L'_i(p)$ with respect to the price for each trading pair. Then for each pair, we compare the two profiles by calculating the metric z_i as follows:

$$I \cdot z_i = \int_0^I \frac{L_i(p(t))}{L'_i(p(t))}\, dt \tag{5}$$

The above metric informs, on average, the enhancement in experienced liquidity when the liquidity profile is $L(p)$ in comparison to $L'(p)$ for a trading pair i. The total initial real reserves for $\mathcal{T}_c$ provided collectively by the LPs equals:

$$\mathbf{r'_{total,c}} = \sum_{\mathbf{i=0}}^{\mathbf{N-1}} \mathbf{r'_{i,c}} \tag{6}$$

Formally, our objective is to maximize the average increase in liquidity z_i for each trading pair without changing the total initial currency reserves $\mathbf{r'_{total,c}}$ and token reserves $\mathbf{r'_i}$ for $\mathcal{T}_i$.

Authorized licensed use limited to: University College London. Downloaded on June 28,2024 at 00:17:07 UTC from IEEE Xplore. Restrictions apply.

C. DEEPER *Overview*

DEEPER is a DEX design that allows LPs of different trading pairs against a common currency to gather and pool their currencies to enhance the available liquidity of each of the trading pairs. Given a batch of N trading pairs $\mathcal{T}_i/\mathcal{T}_c$ on DEEPER, each pair gets real $\mathcal{T}_c$ reserve for liquidity provision in one of the two ways:

1) Individual reserves for concentrated liquidity in each price range (same as in CLAMM).
2) Shared reserves where an LP provides initial liquidity profiles for the inactive price intervals of each pair, and deposits a lump sum $\mathcal{T}_c$ in the shared reserves pool accessible by all pairs.

Let R denote the amount of $\mathcal{T}_c$ in the shared pool at a given instance. During the time interval $[0, T]$, if the price of $\mathcal{T}_i$ increases activating a higher price inactive interval that consists of only $\mathcal{T}_i$, then the accumulated $\mathcal{T}_c$ from the recently inactivated interval is added to the shared reserves pool and R increases. Likewise, if the price of $\mathcal{T}_i$ decreases such that a lower price inactive interval requiring only $\mathcal{T}_c$ becomes active, then those tokens are withdrawn from R and allocated to the newly activated interval. The reserves allocation from the shared pool is designed such that the shared reserve pool never goes negative. In the unlikely event of reducing $\mathcal{T}_i$ prices leading to a dried-up shared reserves pool, the individual concentrated liquidity provision per token pair starts to dominate. This serves as a fail-safe mechanism when the price crash of one trading pair consumes significant shared reserves.

IV. PROTOCOL DESIGN

DEEPER is a CLAMM-based DEX design that allows LPs of several trading pairs with a common currency to come together and share their currency reserves. This, however, does not prevent an LP to provide individual liquidity to just one pair. Thus, for each asset pair $\mathcal{T}_i/\mathcal{T}_c$, we define two kinds of liquidity provisions: *shared* and *individual*. The shared liquidity provision is explained below.

A. Shared Liquidity Provision

The total available shared reserves of $\mathcal{T}_c$ represented by R is split between *busy* reserves or R^b and *available* reserves or R^a so that $R = R^a + R^b$.

The virtual liquidity profile, for shared liquidity provision, of a trading pair i is divided into intervals of prices of uniform width. Each interval is mapped to an integer tick such that if an interval $[p_0, p_1)$ has tick k, then $p[k] = p_0$. Similarly, $L_i[k]$ and $r'_{i,c}[k]$ represent the liquidity and the corresponding $\mathcal{T}_c$ reserves (either active or inactive) in tick k at a given instance. Let K_i be the tick of the currently activated interval with $\mathbf{K_i}$ being its value at the beginning, *i.e.* $t = 0$ for each trading pair $\mathcal{T}_i/\mathcal{T}_c$.

Liquidity provision using shared $\mathcal{T}_c$ reserves consists of the following specifications:

1) To provide shared $\mathcal{T}_c$ at $t = 0$, LPs need to *(i)* provide liquidity profile for each trading pair i, *i.e.* $L_i[k]\ \forall k \leq \mathbf{K_i}$; *(ii)* deposit $\mathbf{r'_{i,c}}$ units of $\mathcal{T}_c$ reserves in accordance with the above liquidity profile for each i. For shared provision, we call L_i the *skeleton* liquidity profile because it will be used to calculate the actual liquidity of a price interval. Similarly, $\mathbf{r'_{i,c}}$ are called the skeleton $\mathcal{T}_c$ reserves. Since sharing LPs can only provide $\mathcal{T}_c$ reserves, the skeleton liquidity is positive in price ticks that are less than the activated ticks and zero elsewhere. The skeleton liquidity in each interval remains constant unless some LP mints or burns their shared liquidity.
2) The actual liquidity of pair i in the active price interval is denoted by $\mathcal{L}_i$ and it remains constant when the price of $\mathcal{T}_i$ lies within the active price interval. Swaps that do not change the price interval are executed based on CLAMM with $\mathcal{L}_i$ as the virtual liquidity of the active interval.
3) When the price of $\mathcal{T}_i$ decreases such that the tick transitions from K_i to $K_i - 1$ with the new interval having $\mathbf{r'_{i,c}}[K_i - 1]$ skeleton $\mathcal{T}_c$ reserves, the following events occur:
   - The new active interval secures $r'_{i,c}[K_i - 1]$ units of $\mathcal{T}_c$ from available reserve pool such that:
   $$r'_{i,c}[K_i - 1] = R^a\left(\frac{\mathbf{r'_{i,c}}[K_i - 1]}{\sum_{j=0}^{K_i-1} \mathbf{r'_{i,c}}[j]}\right) \quad (7)$$
   This is secured by reducing the available reserves pool and increasing the busy reserves pool. Since $r'_{i,c}[K_i - 1]$ is always a fraction of the available reserves R^a, it never goes negative after the operation. The actual liquidity can be derived from real reserves using the relation in Equation 4.
   - The token $\mathcal{T}_i$ in the tick K_i becomes inactive with its amount stored in memory and it reactivates in the future when the tick transitions from $K_i - 1$ to K_i as discussed next.
4) When the price of $\mathcal{T}_i$ increases and the tick crosses from K_i to $K_i + 1$, then the accumulated $\mathcal{T}_c$ in the newly inactive tick K_i is transferred from busy reserves to available reserves and any $\mathcal{T}_i$ reserves stored in the memory are released for tick $K_i + 1$.

Lastly, if the price of a pair $\mathcal{T}_i$ goes all the way down to tick 0, then this pair consumes all available shared $\mathcal{T}_c$. In such a case, individual reserves become dominant. The shared reserves, however, are restored when the price of this pair starts to increase. Therefore, the available shared $\mathcal{T}_c$ can be potentially secured by any trading pair, and the first one to access it secures $\mathcal{T}_c$ while the quota per price interval of every other pair reduces.

B. Shared Liquidity Withdrawal & LP Fees

LPs can withdraw their shared $\mathcal{T}_c$ reserves at any point. In doing so, they receive the proportion of available reserves and any inactive $\mathcal{T}_i$ that belongs to them, and the skeleton liquidity provided by them is removed for each pair. Further, any trading fee that is accrued in a price interval is distributed amongst the LPs in proportion to their skeleton liquidity.

Authorized licensed use limited to: University College London. Downloaded on June 28,2024 at 00:17:07 UTC from IEEE Xplore. Restrictions apply.

C. Individual Liquidity

An LP is free to provide individual liquidity in any price range by providing $\mathcal{T}_i$ or $\mathcal{T}_c$ or both depending on the active state of the interval of interest. Since they cannot provide shared reserves for $\mathcal{T}_i$ for liquidity, individual liquidity is the only way to serve this purpose.

D. Divergence Loss for Shared Liquidity Providers

Divergence loss is defined as the opportunity cost for an LP to provide token reserves as liquidity compared to just holding them. In CPAMM and CLAMM, given an initial liquidity profile (for *e.g.*, Figure 2), the divergence loss is a function of the token price [16]. Since an LP can recover any accrued losses when $\mathcal{T}_i$ trades back at the initial price (when liquidity was provided), divergence loss is not permanent. Therefore, it is also referred to as *impermanent loss*. In the case of DEEPER, however, the divergence loss is not a function of just the token price, because the total $\mathcal{T}_c$ reserves owned by an LP depend on the relative order of securing $\mathcal{T}_c$ from the shared reserves pool by the trading pairs.

However, the divergence loss *still* remains impermanent for an LP. This is described in the lemma below:

Lemma IV.1. *The divergence loss of an LP providing shared reserves for $\mathcal{T}_c$ at price p_i for token $\mathcal{T}_i$ with respect to $\mathcal{T}_c$ and subsequently withdrawing their liquidity at the same initial price for each token is zero and independent of any intermediary price movements.*

Proof. Suppose LPs provide a total R reserve of shared $\mathcal{T}_c$ and a skeleton liquidity profile L_i at an initial price $\mathbf{p}_i$ for each pair i. Then, this profile is defined for the ticks less than the current active tick K_i for each pair. We prove that the shared reserve for $\mathcal{T}_c$ equals R when the price of $\mathcal{T}_i$ becomes $\mathbf{p}_i$ for all i.

When the current tick transitions from k to $k-1$ and secures $\mathcal{T}_c$ from the shared pool, the amount of shared reserves secured by tick $k-1$ is stored in its memory which serves as its history. Eventually, when the tick transitions back from $k-1$ to k, the shared $\mathcal{T}_c$ that was withdrawn earlier is added back to the shared pool.

Therefore, when the final prices become the same as the initial prices, the initial and final active ticks become the same for all the pairs. Therefore, any borrowed $\mathcal{T}_c$ from the shared reserve pool is released back. The LPs can now withdraw exactly the initially supplied $\mathcal{T}_c$ from the shared pool. Any $\mathcal{T}_c$ or $\mathcal{T}_i$ that was supplied as part of individual liquidity remains the same since individual liquidity reserves are a function of token prices. Since the LP can withdraw exactly the same amount of reserves that they supplied initially and at the same initial price, the total divergence loss suffered is zero. □

E. Smart Contract Friendliness

A smart contract [17] is a Turing-complete program that is deployed on a blockchain. A blockchain account can asynchronously trigger functions in these programs by paying gas fees that are charged per contract operation during a call. Therefore, an ideal smart contract design performs the minimal amount of updates in the state (*i.e.,* variables) of the contract during a function call. Without loss of generality, our implementation of DEEPER DEX is an extension to Uniswap V3 [18]. We add the following parameters on top of the previous design to include shared liquidity:

1) We define skeleton liquidity that remains invariant throughout the period of no deposit or withdrawal of liquidity. This is implemented exactly how liquidity is implemented in Uniswap. Further, we differentiate skeleton liquidity from actual liquidity. Actual liquidity can be of one of two forms: active and inactive.
2) Each tick (as implemented in Uniswap) keeps track of ΔL which is the change in skeleton liquidity. At the same time, it keeps track of the cumulative skeleton $\mathcal{T}_c$ below the active tick and total available $\mathcal{T}_c$ or R^a. This allows it to calculate the $r'_{i,c}$ for a range using Equation 7 when the price decreases and a tick is crossed. The actual liquidity of the current tick is calculated using the linear relationship of L and $r'_{i,c}$ at the interval edge as shown in Equation 4. The value of the frozen $\mathcal{T}_i$ is stored for all ranges crossed from the above tick. Every time when a tick is crossed, R^a is updated accordingly.

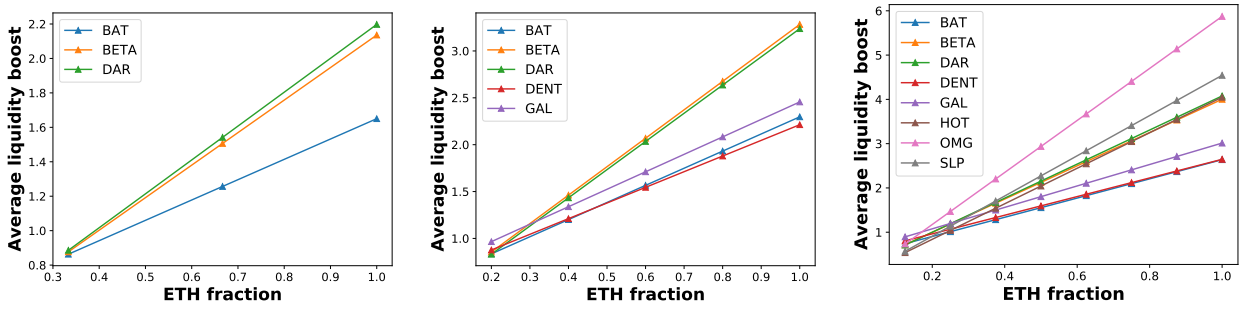
V. PROTOCOL EVALUATION

In this section, we evaluate the benefits of the DEEPER DEX and study the parameters that optimize them. Our experiments attempt to answer the following questions: *(i)* What is the average increase in the experienced liquidity of our shared liquidity model compared to the individual liquidity model? *(ii)* What is the relationship between the shared liquidity boost and the number of trading pairs in a shared batch? *(iii)* How can LPs optimize their enhanced liquidity by varying I?

A. Methodology

For the purposes of this evaluation, we use the historical price data from the month of December 2022 of trading pairs from the Uniswap V3 DEX. Each of the trading pairs exhibits the following two properties: (*i*) they are traded against Wrapped ETH, and (*ii*) had an average daily volume between \$0–50k. Such a trading pair with a trading fee of 0.3% generates a total revenue of \$0–150 per day for all the LPs collectively. We create three batches each containing 3, 5, and 8 trading pairs. We use three time periods of 1 day, 7 days, and 14 days for I during which the skeleton liquidity profiles remain constant. For a given token batch and time period I, we simulate the liquidity environment by initializing liquidity between the minimum and maximum price during that month. For the shared ETH, we input the skeleton liquidity profile and the corresponding ETH reserves for each pair. The skeleton liquidity peaks at the start price and decays slightly from there. For prices above the start price, we initialize individual liquidity by providing the asset token in each price interval. This liquidity decreases as price increases similar to ETH. We also create, for comparison, a model where ETH

Authorized licensed use limited to: University College London. Downloaded on June 28,2024 at 00:17:07 UTC from IEEE Xplore. Restrictions apply.



(a) Batch of 3 assets (b) Batch of 5 assets (c) Batch of 8 assets

Fig. 4: Average liquidity per initial ETH boost for different asset batches

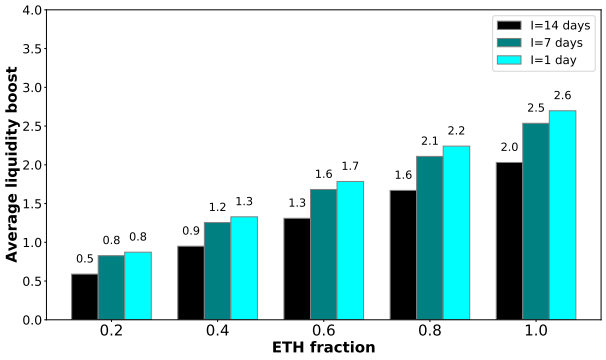


Fig. 5: Liquidity increase for multiple values of I.

is provided via individual liquidity with equal ETH allocation to each pair.

B. Summary of Results

Liquidity boost: Figure 4a shows the average increase in liquidity (z in Equation 5) for a batch consisting of 3 trading pairs and I set to 1 day. The x-axis represents the total amount of initial ETH deposited as a fraction of the ETH used in the individual liquidity provision. The graph is plotted for three values of ETH fractions: $\frac{1}{3}$, $\frac{2}{3}$, 1. We can observe that the shared liquidity is more than 80% of its individual counterpart while costing only a third of ETH reserves. Moreover, the amount of initial ETH required reduces by $55.1\%, 60.2\%$, and 60.9% for the three trading pairs respectively to achieve the same average liquidity as in the individual model. When the initial ETH reserves are increased so as to consume the same ETH as the individual model, the experienced liquidity increases by $1.6\times$, $2.1\times$, $2.2\times$ for the three trading pairs respectively. Therefore, DEEPER DEX significantly increases liquidity without consuming any surplus asset reserves.

Batch size: Figures 4b, 4c illustrates the liquidity increase with respect to the ETH reserves used for a batch of size 5, and 8 assets respectively. For these batches, the cost of initial ETH reduces by 70.0–78.2% and 75.6–83.2% respectively to achieve the same liquidity as the contemporary model. The corresponding average liquidity increase is between 2.2–3.3 and 2.6–5.9 respectively. This shows that as more trading pairs pool their ETH, the liquidity per pair increases while the cost of ETH to achieve similar levels of liquidity decreases.

Variation with I: Figure 5 shows the liquidity boost averaged over all the trading pairs for different values of I i.e. 1 day, 7 days, and 14 days respectively. The key observation here is that the liquidity increase decreases over longer values for I. This is because when a trading pair consumes the available ETH, there is less ETH left in the shared pool for other pairs. This becomes dominant in longer time intervals in a market scenario with decreasing prices. Therefore, LPs should either update their skeleton liquidity profiles more frequently or provide the skeleton liquidity over a longer price range to observe high shared liquidity enhancement.

VI. Conclusion and Future Work

The experimental results for DEEPER shows that reserve sharing significantly enhances average liquidity. With its shared reserve allocation mechanism and simple design, DEEPER is a practical solution to the problem of shallow liquidity provision in low trading volume trading pairs.

Future work includes accounting for LP earnings from trading fees and impermanent loss when calculating the common currency allocation to a trading pair's pool.

References

[1] V. Buterin, "Ethereum: A next-generation smart contract and decentralized application platform," 2014, https://ethereum.org/en/whitepaper/.
[2] S. M. Werner, D. Perez, L. Gudgeon, A. Klages-Mundt, D. Harz, and W. J. Knottenbelt, "Sok: Decentralized finance (defi)," *arXiv preprint arXiv:2101.08778*, 2021.
[3] W. Ding, J. Hou, J. Li, C. Guo, J. Qin, R. Kozma, and F.-Y. Wang, "Desci based on web3 and dao: A comprehensive overview and reference model," *IEEE Transactions on Computational Social Systems*, vol. 9, no. 5, pp. 1563–1573, 2022.
[4] J. Proelss, S. Sevigny, and D. Schweizer, "Gamefi-the perfect symbiosis of blockchain, tokens, defi, and nfts?" *Tokens, DeFi, and NFTs*, 2023.
[5] Z. Liu, Y. Xiang, J. Shi, P. Gao, H. Wang, X. Xiao, B. Wen, Q. Li, and Y.-C. Hu, "Make web3. 0 connected," *IEEE transactions on dependable and secure computing*, vol. 19, no. 5, pp. 2965–2981, 2021.
[6] P. Dutta, T.-M. Choi, S. Somani, and R. Butala, "Blockchain technology in supply chain operations: Applications, challenges and research opportunities," *Transportation research part e: Logistics and transportation review*, vol. 142, p. 102067, 2020.
[7] (2022) Friends with benefits. https://www.fwb.help/events.
[8] M. C. Ballandies, "To incentivize or not: Impact of blockchain-based cryptoeconomic tokens on human information sharing behavior," *IEEE Access*, vol. 10, pp. 74 111–74 130, 2022.

Authorized licensed use limited to: University College London. Downloaded on June 28,2024 at 00:17:07 UTC from IEEE Xplore. Restrictions apply.

[9] L. Oliveira, L. Zavolokina, I. Bauer, and G. Schwabe, "To token or not to token: Tools for understanding blockchain tokens." ICIS, 2018.
[10] M. Ottina, P. J. Steffensen, and J. Kristensen, "Balancer," in *Automated Market Makers: A Practical Guide to Decentralized Exchanges and Cryptocurrency Trading*. Springer, 2023, pp. 69–116.
[11] H. Adams, N. Zinsmeister, and D. Robinson, "Uniswap v2 core," 2020.
[12] G. Angeris, H.-T. Kao, R. Chiang, C. Noyes, and T. Chitra, "An analysis of uniswap markets," *arXiv preprint arXiv:1911.03380*, 2019.
[13] L. Liu, S. Zhou, H. Huang, and Z. Zheng, "From technology to society: An overview of blockchain-based dao," *IEEE Open Journal of the Computer Society*, vol. 2, pp. 204–215, 2021.
[14] P. Züst, "Analyzing and preventing sandwich attacks in ethereum," 2021.
[15] A. J. Morales, S. Somin, Y. Altshuler, and A. Pentland, "User behavior and token adoption on erc20," *arXiv preprint arXiv:2005.12218*, 2020.
[16] S. Loesch, N. Hindman, M. B. Richardson, and N. Welch, "Impermanent loss in uniswap v3," *arXiv preprint arXiv:2111.09192*, 2021.
[17] Z. Zheng, S. Xie, H.-N. Dai, W. Chen, X. Chen, J. Weng, and M. Imran, "An overview on smart contracts: Challenges, advances and platforms," *Future Generation Computer Systems*, vol. 105, pp. 475–491, 2020.
[18] H. Adams, N. Zinsmeister, M. Salem, R. Keefer, and D. Robinson, "Uniswap v3 core," *Tech. rep., Uniswap, Tech. Rep.*, 2021.

Authorized licensed use limited to: University College London. Downloaded on June 28,2024 at 00:17:07 UTC from IEEE Xplore. Restrictions apply.