Superimposed Liquidity: Enhancing Concentrated Liquidity AMM Pool with On-Chain Limit Order Book

Problem Statement

Limit orders play a crucial role in cryptocurrency trading as they allow traders to maintain better control over their trades by setting buy/sell prices based on their comfort level. However, primitive decentralized limit order mechanisms lack sophistication, forcing traders to rely on centralized exchanges, which contradict the principles of blockchain technology. Moreover, centralized exchanges are subject to regulatory scrutiny and can restrict users' trading and access to funds. To address these issues, it is crucial to develop a secure, scalable, and efficient on-chain mechanism that can handle the volume and complexity of limit order trades. The proposed solution must be robust enough to withstand market volatility and allow users to customize their trading strategies. Decentralized limit order mechanisms must become reliable enough to support their adoption as a trustworthy trading mechanism.

Analysis of Existing Decentralized Solutions for Limit Orders

The incumbent decentralized limit order engines maintain users' signed limit orders and execute them in one of two ways:

- Via an off-chain order matching system.
- Via off-chain triggers that perform a market price swap when the desired price is reached.

Drawbacks of these solutions:

Although christened as "decentralized" limit order systems, these are "semi-decentralized" at best. Nonetheless, one benefit of placing limit orders through this approach is that the custody of the asset stays with the owner. However, the shortcomings presented by this approach are far more significant than the benefit it provides.

- Although the trade settlement occurs on-chain, orders are aggregated on off-chain nodes, due to which limit order data is shrouded from scrutiny.
- Another major disadvantage of placing off-chain limit orders is the delay in triggers that can occur while executing the trade. Since the takers have to fetch the order from an off-chain database and then settle the order on-chain, the time for completing the trade increases, potentially leading to missed trading opportunities.
- There is no assurance that a taker will fill the order even at the desired price level. For a taker to consider filling the order, it must be profitable for them, which requires them to consider factors such as the order's size, gas fees, and personal profit margin. As a result, there is no guarantee that an order will be filled.
- As mentioned above, limit orders in such mechanisms are placed off-chain. Hence such orders act as traders that squeeze the liquidity instead of market makers that add to it. This is quite the opposite of the on-chain scenario we'll be presenting in this paper, where limit orders add to the exchange's liquidity.
- DEXes offering off-chain limit orders pocket any difference between the limit order price and the actual market price, which in principle, should belong to the traders. Take, for example, a limit order placed at $10. If the trade is fulfilled after the price reaches $12 ($2 above the asking price), the exchange pockets the difference between the limit price and the realized price.

These limitations of existing limit order implementations highlight the need for more efficient and reliable solutions for decentralized limit orders to improve the user experience and increase adoption.

Dfyn V2's On-chain Limit Order Model

Introduction:

Dfyn V2 has introduced an innovative hybrid model that leverages concentrated liquidity AMM and limit order liquidity to address the aforementioned limitations and offer a superior solution for decentralized on-chain limit orders. In this system, the limit order liquidity is superimposed on the concentrated liquidity curve

. This model overcomes the challenges of low liquidity and unpredictable slippage by combining the advantages of both order types. The concentrated liquidity AMM allows the exchange to offer a high level of liquidity for popular trading pairs, while the limit order liquidity provides users with greater control over the execution price of their trades. This hybrid approach provides users with the best of both worlds, ensuring fast and reliable trade execution while giving them greater flexibility and control over their trades.

Overview of Concentrated Liquidity and How it is implemented using Ticks

What is a Tick?

In financial markets, a tick refers to the smallest possible change in the price of a financial instrument. For instance, if a particular asset has a tick size of $0.01, this means that its price can only be altered in increments of $0.01 and cannot fluctuate by a smaller amount. The tick size for different asset classes may vary depending on various factors, such as the liquidity of the asset and the minimum price movement that the market can accommodate.

Tick Implementation in Concentrated Liquidity:

[

Concentrated Liquidity Ticks

2250×2250 71.4 KB

](https://ethresear.ch/uploads/default/original/2X/7/7fdb29d622bc54c03201497a0386d8c660370065.png)

Concentrated liquidity allows liquidity providers (LPs) to concentrate their capital into smaller price intervals, resulting in individualized price curves and increased capital efficiency. Dfyn V2 allows for more precise trading by dividing the price range [0, ∞] into granular ticks, similar to an order book exchange.

- The system defines the price range corresponding to each tick rather than relying on the user

input.

- Trades within a tick still follow the pricing function of the constant product market maker, but the equation is updated every time the price crosses a tick's price range.
- For every new position, we insert two elements into the linked list based on the range's start and end prices. To keep the list manageable, we only allow the prices to be represented as powers of 1.0001

. For example, our range could start at tick(0)

with a price of 1.00010^0 = 1.0000

and end at tick(23028)

, which corresponds to a price of approximately 1.0001^{23028} = 10.0010

. We can use this approach to cover the entire (0, ∞)

price range using only 24-bit integer values.

Tick structure:

struct Tick { int24 previousTick; int24 nextTick; uint128 liquidity; uint256 feeGrowthOutside0; uint256 feeGrowthOutside1; uint160 secondsGrowthOutside; }

To represent the linked list, we create a mapping of 24-bit integers to "Tick" structures, where each tick holds pointers to the previousTick

and the nextTick

, the amount of liquidity within the tick's price range, as well as other variables to track fees and time spent within a price range.

Limit Order Ticks

[

Limit Order Ticks

2250×2250 61.3 KB

](https://ethresear.ch/uploads/default/original/2X/b/b60f64c3e8a40ea0117dfeec7f750984e906f6d8.png)

Dfyn V2 introduces limit order ticks, allowing for highly concentrated liquidity on a single price point, resulting in improved price precision. This feature enables users to place limit orders at a specific price, emulating order book exchanges. The

implementation involves inserting new ticks on the AMM curve, with liquidity concentrated on a single tick. Overall, this offers a significant improvement over legacy tick logic, where liquidity had to be distributed between ticks in a range. The limit order ticks are embedded in the concentrated AMM curve itself and all the limit order ticks together represent an on-chain order book.

Nature of Liquidity

- Concentrated Liquidity

It is a bi-directional concept that refers to liquidity that is used for a swap in a certain direction and can be utilized again

if the trend reverses to the opposite direction, thus converting back

to the previous token.

- Limit Order Liquidity

It is a uni-directional concept that refers to liquidity that is used for a swap in a certain direction and is not utilized

if the trend reverses to the opposite direction, hence not converting back

to the previous token.

Dfyn V2 has integrated these concepts into a single curve by using a superimposed liquidity model.

Superimposed Liquidity: The Hybrid Model

[

Hybrid Curve

2250×2250 82.8 KB

](https://ethresear.ch/uploads/default/original/2X/6/6b8462857408d0d0b29e4796ca3a792550315531.png)

Superimposed liquidity combines the price granularity of concentrated liquidity ticks and precise pricing of on-chain limit orders in the same pool for improved liquidity depth and greater capital efficiency. This model allows traders to execute trades at favorable prices with minimized slippage. The integration of on-chain limit orders further enhances trading precision.

Now a tick on any curve can hold two types of liquidity, concentrated and limit liquidity.

Concentrated Liquidity AMM

Superimposed Liquidity AMM

Has only bi-directional liquidity

Has bi-directional and uni-directional liquidity

Liquidity is strictly range-bound

Liquidity can be added in a range as well as on a tick

Limit orders cannot be implemented

Limit orders can be implemented

In a range, only concentrated liquidity is available to be utilized for swaps

In a range, concentrated, and limit order liquidity, both are utilized for swaps

Prone to high slippage

Comparatively lower slippage, as two types of liquidity are available on a tick leading to better depth

Due to concentrated liquidity, the price offered for trading is as per the industry standards.

Offers even better prices than industry standards due to increased depth of liquidity on a single price point.

Liquidity Interaction between Concentrated and Limit Order Liquidity

In this section, we'll take a look at how both types of liquidity are handled and how trade settlements are done in the superimposed liquidity curve model with the help of an example.

- Consider that Alice provides liquidity in the ETH-USDC curve between the 1800 to 2200 range.
- Liquidity is evenly spread between the ticks in that range, each tick represents Alice's liquidity (1800 to 2000 contains USDC, 2000 to 2200 contains ETH).
- Bob wants to buy 1 ETH when the price of ETH reaches 1900 USDC, so he places a limit order at that price point.
- Since there is a superimposition of liquidity, the tick at the 1900 price point contains both Alice's concentrated liquidity and Bob's limit order liquidity, increasing liquidity depth on that tick.
- Alice's liquidity proportion will be updated in tandem with the ETH price fluctuation.
- When the ETH price reaches the 1900 price point tick, both Alice and Bob's liquidity will be utilized to fulfill swaps on the ETH-USDC curve, filling Bob's buy limit order.
- Once the price reduces and crosses the 1900 price point tick completely, both Alice and Bob's liquidity at the 1900 tick will consist of ETH.

This was an overview of how concentrated liquidity and limit order liquidity go hand in hand.

Let's take a deep dive into how limit orders are created, settled, and claimed and all the conditions involved.

On-chain Limit Order Book and Order Settlement

1. Creating a Limit Order
2. To create a limit order, liquidity is added to the Limit Order Tick.
3. When a limit order is created, an NFT called LimitOrderToken

is minted based on the user's tokenId

.

- A snapshot of the order's state is taken and stored in the limitOrders

mapping.

- This snapshot contains the tokenClaimableGrowth

on that tick during that time which is used to track the claimable amount of the user's limit order.

1. To create a limit order, liquidity is added to the Limit Order Tick.
2. When a limit order is created, an NFT called LimitOrderToken

is minted based on the user's tokenId

.

1. A snapshot of the order's state is taken and stored in the limitOrders

mapping.

1. This snapshot contains the tokenClaimableGrowth

on that tick during that time which is used to track the claimable amount of the user's limit order.

1. On-Chain settlement of Limit Orders
2. As we know, a user's limit order liquidity is embedded on the concentrated liquidity curve, so when swaps happen limit order liquidity is utilized along with concentrated liquidity.
3. When a swap happens, the system uses the SwapExecuter

library's _executeConcentrateLiquidity

function to utilize the concentrated liquidity and _executeLimitOrder

function to utilize limit order liquidity.

- When utilizing the limit order liquidity of a tick, tokenClaimableGrowth

of that tick is updated.

- When both types of liquidity have been exhausted, we cross to the next tick in that direction.
- As we know, a user's limit order liquidity is embedded on the concentrated liquidity curve, so when swaps happen limit order liquidity is utilized along with concentrated liquidity.
- When a swap happens, the system uses the SwapExecuter

library's _executeConcentrateLiquidity

function to utilize the concentrated liquidity and _executeLimitOrder

function to utilize limit order liquidity.

1. When utilizing the limit order liquidity of a tick, tokenClaimableGrowth

of that tick is updated.

1. When both types of liquidity have been exhausted, we cross to the next tick in that direction.
2. Claiming a Limit Order
3. After limit order liquidity is utilized to perform a swap, the user can claim the limit order, either partially or entirely.
4. In order to differentiate the limit order liquidity of users who placed limit orders before the tick was crossed, from the users who placed after the tick was crossed and the trend reversed, we check the snapshot of tokenClaimableGrowth

of their order.

- For a user who had placed a limit order before, the current tokenClaimableGrowth

is greater than the tokenClaimableGrowth

in their snapshot which was taken when the order was created.

- For a user who had placed a limit order later, the current tokenClaimableGrowth

is not greater than the tokenClaimableGrowth

in their snapshot which was taken when the order was created. Hence a user who placed the order later cannot claim the filled order of the previous users. He has to wait till the market reaches his price again and utilizes his limit order liquidity.

1. After limit order liquidity is utilized to perform a swap, the user can claim the limit order, either partially or entirely.
2. In order to differentiate the limit order liquidity of users who placed limit orders before the tick was crossed, from the users who placed after the tick was crossed and the trend reversed, we check the snapshot of tokenClaimableGrowth

of their order.

1. For a user who had placed a limit order before, the current tokenClaimableGrowth

is greater than the tokenClaimableGrowth

in their snapshot which was taken when the order was created.

1. For a user who had placed a limit order later, the current tokenClaimableGrowth

is not greater than the tokenClaimableGrowth

in their snapshot which was taken when the order was created. Hence a user who placed the order later cannot claim the filled order of the previous users. He has to wait till the market reaches his price again and utilizes his limit order liquidity.

Example:

[

Example

1642×884 45.6 KB

](https://ethresear.ch/uploads/default/original/2X/7/79d5392413f0569ccadb6fe5b34c0e25f7ce7a63.png)

The above cases represent the execution of limit orders placed on the hybrid liquidity curve of the ETH-USDC pair where swaps are happening simultaneously.

- Alice and Bob place sell limit orders on the same tick, Tick B.
- Alice places a sell limit order of 1 ETH and Bob places a sell limit order of 2 ETH.
- The total limit order liquidity of sell orders on Tick B is 3 ETH.
- The current market price arrives at the previous tick, Tick A.

When swaps happen in the same direction, all the concentrated liquidity available on Tick B is used first. If there is still an amount to be swapped, then the tick is to be crossed. Before crossing the tick, the system checks for limit order liquidity on Tick B.

Now for order settlement multiple cases arise.

Case 1: The swap is for buying 4 ETH

[

Case 1

2250×2250 70.1 KB

](https://ethresear.ch/uploads/default/original/2X/6/6992d125be394007aab3a3673be8dbaa7c7ffe49.png)

- The swap amount is greater than all the available limit order liquidity on Tick B.
- Alice and Bob’s limit liquidity (3 ETH) is completely utilized, and their orders are completely filled.
- Alice and Bob can then claim their fully filled orders.
- Since the entire limit order liquidity is exhausted, the current tick is crossed to Tick B.
- Now there is still 1 ETH remaining to be bought.
- We check for concentrated liquidity on Tick C.
- If concentrated liquidity is available on Tick C, we use that to complete the swap.
- If concentrated liquidity is unavailable on Tick C, we check limit order liquidity on Tick C.

Case 2: The swap is for buying 2 ETH

[

Case 2

2250×2250 77.8 KB

](https://ethresear.ch/uploads/default/original/2X/1/1d4c5ca42910340d4c0f9f0bce43668ce07b39fb.png)

- The swap amount is less than all the available limit order liquidity on Tick B (3 ETH).
- In this case, only a partial amount of limit liquidity, i.e. 2 ETH, is utilized.
- The partially filled limit liquidity amount can be entirely claimed either by Bob filling his sell order of 2 ETH, or Alice can claim her entire sell order of 1 ETH leaving Bob with a partial fill of 1 ETH.
- Both cases can happen depending on who claims first, creating a race condition for claiming.
- The current tick is still on Tick A since there is 1 ETH limit liquidity to be filled on Tick B.

Case 3: The swap is for buying 4 ETH but the trend reverses

[

case3

2092×702 55.7 KB

](https://ethresear.ch/uploads/default/original/2X/7/7ccfab6691acf0672559333e5a1700c5e0e1255e.png)

- The swap amount is greater than all the available limit order liquidity on Tick B.
- Alice and Bob’s limit orders have been entirely filled, similar to Case 1

, however, they haven’t claimed it.

- The trend of the market changes and swaps start occurring in the opposite direction, making Tick A the current tick again.
- A new user Carol adds a 1 ETH sell order on Tick B.
- Since limit order liquidity is uni-directional, Tick B has filled limit order liquidity (worth 3 ETH) of Alice and Bob and unfilled limit order liquidity of Carol (1 ETH).
- The system has to differentiate which user had placed the limit order before and after the cross, which is done with the help of tokenClaimableGrowth

as we read earlier.

- For Carol, tokenClaimableGrowth

has not been updated, hence cannot claim any of the filled limit liquidity on Tick B

Case 4: Race condition for claiming

If multiple sellers step in to place limit sell orders at Tick B and multiple buyers purchase ETH at market price, the buyers will keep filling the limit order liquidity at Tick B simultaneously. This creates a race condition for all the participants to claim their entirely/partially filled sell orders at Tick B.

The fee structure in Superimposed Liquidity

As we have seen till now, superimposed liquidity AMM utilizes concentrated liquidity and limit order liquidity. Both types of these liquidities have different fees. When a trader’s swap utilizes both types of liquidity, the trader must pay the standard AMM fee for the portion of their trade fulfilled via the AMM liquidity, while the remainder fulfilled via limit order liquidity incurs a separate fee payable to the protocol.

This fee can be adjusted independently for each liquidity pool, and since the protocol directly benefits from limit order liquidity, it can have lower fee requirements for limit order creators.

Hence the trader’s total fee charged for a trade in superimposed liquidity is less than concentrated liquidity

This model results in lower trading fees for traders, minimal or zero cost for limit order creators, and an additional revenue stream for the protocol.

Fee Rebate:

In Dfyn V2, to implement a limit order the user adds limit order liquidity to the tick on the curve.

Unlike traditional limit order models, a Limit Order Creator in Dfyn V2 is a Maker.

Hence we have implemented a fee rebate mechanism that incentivizes the limit order creators of certain pools.

Simulations on Real-Time Data

The data represent the results of a simulation of a buy swap with two different liquidity types - concentrated and superimposed - for increasing WETH amounts for the same start price.

The simulation also accounts for a fee charged during the trade.

The simulation was conducted for 10 different WETH amounts, ranging from 100 to 1000. For each WETH amount, two trades were simulated - one with concentrated liquidity and the other with superimposed liquidity. The end price and fee charged for the trades varied for each simulation.

WETH Amount

Concentrated Liquidity

Fee Charged On Concentrated Liquidity

Superimposed Liquidity

Fee Charged on Superimposed Liquidity

100

1923.736272

0.07524009332

1921.813593

0.03021438534

200

1925.660874

0.1332471955

1923.736272

0.08024009332

300

1927.587402

0.1542258961

1925.660874

0.1382471955

400

1943.069171

0.2213269276

1935.312806

0.1893628111

500

1950.856623

0.2743538478

1945.013115

0.2325817149

600

1974.406743

0.3303091548

1950.856623

0.2806271488

700

1980.33856

0.3567773814

1974.406743

0.3365824558

800

2000.240294

0.4288850947

1994.248864

0.3933841369

900

2014.290391

0.4504022722

1995

0.4333841369

1000

2050.874054

0.5004013277

1996

0.4333854869

Liquidity Depth:

[

Liquidity depth

2372×1210 105 KB

](https://ethresear.ch/uploads/default/original/2X/8/8d96debe87fe0262420a4d5a17d36c4ce57891bd.png)

As seen in the graph above we can see the two types of liquidity, concentrated and limit order, involved in this simulation.

In the case of a concentrated pool, only concentrated liquidity is taken into consideration on a single price point, whereas in the case of a superimposed pool, both concentrated and limit order liquidity are considered.

Swap Result:

[

Swap Result

1864×1391 182 KB

](https://ethresear.ch/uploads/default/original/2X/9/976292fa270dd9ad4239630a7f592f8811ae938c.png)

As we can observe from the graph, as the amount of WETH involved in the trade increases, the end price also increases for both liquidity types. However, the end price for the concentrated liquidity type pool is generally higher than that for the superimposed liquidity type pool. This is because the superimposed liquidity type pool has a higher depth available at each price point due to limit order liquidity embedded in the curve.

Fee Charged:

[

Fee Charged

1770×1292 148 KB

](https://ethresear.ch/uploads/default/original/2X/d/d5e80daf67b6dcb825e330687dc255482e3c9cb1.png)

Additionally, we can observe that as the amount of WETH involved in the trade increases, the fee charged also increases for both liquidity types. The fee charged for the concentrated liquidity type is generally higher than that for the superimposed

liquidity type. This is because, in the superimposed liquidity type, the user's swap is filled by concentrated and limit order liquidity, which reduces the average fee charged for the swap.

Advantages of Dfyn V2's On-Chain Limit Orders

The implementation of on-chain limit orders in the trading of assets can offer several advantages, including:

Automation:

Since the limit order liquidity is embedded in the concentrated liquidity pool curve, the execution of on-chain limit orders is triggered automatically when the market price of the pool reaches the specified price, thus eliminating the need for a third-party intermediary and resulting in a more streamlined and efficient trading process.

Guaranteed Fill:

Limit order liquidity embedded on the price curve assures that limit orders placed at a specific price point are bound to be filled before the price crosses to the next point. This guarantees an order fill and ensures that the market price does not move beyond without utilizing limit order liquidity.

Since limit order liquidity is uni-directional in nature, even after the trend has reversed the filled limit order asset won't be converted to the previous asset.

Transparency:

The execution of on-chain limit orders is recorded on a blockchain, providing a transparent and immutable record of all trades. This enhances trust and confidence in the market.

Security:

On-chain limit orders are facilitated by smart contracts, which offer a secure and tamper-proof mechanism for trade execution. This minimizes the risk of fraud and ensures that users retain full control over their assets.

Future Work

To improve on the race-for-claim of limit orders, we can implement the following methods :

1. Implementing an auto-claim mechanism
2. An auto-claim mechanism can be implemented by integrating oracles or incentivizing relayers.
3. Oracles can be used to monitor the order book and automatically claim partially filled orders on behalf of the users.
4. Relayers can be incentivized to automatically claim the orders by offering them rewards for processing orders in a timely manner.
5. This solution can significantly reduce the likelihood of disputes or conflicts arising due to the race for claims, as users do not need to manually claim their orders.
6. An auto-claim mechanism can be implemented by integrating oracles or incentivizing relayers.
7. Oracles can be used to monitor the order book and automatically claim partially filled orders on behalf of the users.
8. Relayers can be incentivized to automatically claim the orders by offering them rewards for processing orders in a timely manner.
9. This solution can significantly reduce the likelihood of disputes or conflicts arising due to the race for claims, as users do not need to manually claim their orders.
10. Implementing a priority system
11. A priority system can be implemented by assigning different priority levels to orders based on certain criteria.
12. The priority levels can be based on the time of placement or the amount of liquidity provided.
13. Orders placed earlier or with higher liquidity may be given higher priority, which can help prevent disputes and conflicts arising from the race for the claim.
14. This solution can be effective in reducing the race for claims, but may not completely eliminate it.

Conclusion

To summarize, the article discusses the limitations of existing decentralized limit order mechanisms and introduces Dfyn V2's innovative superimposed liquidity AMM model which has on-chain limit orders. The current decentralized solutions, which rely on off-chain order matching systems and market price swaps, have several drawbacks, including delayed triggers, shrouded data, and limited liquidity. In contrast, Dfyn V2's model combines concentrated liquidity AMM and limit order liquidity, providing users with greater control over the execution price of their trades while ensuring fast and reliable trade execution. The system divides the price range into granular ticks, enabling more precise trading and increased capital efficiency. Overall, Dfyn V2's superimposed liquidity AMM model offers a superior solution for decentralized limit orders, providing users with the best of both world's liquidity, and control over their trades.