How Liveness Separates CFMMs and Order Books

Tarun Chitra
tarun@gauntlet.network

Guillermo Angeris
angeris@stanford.edu

Alex Evans
ahe4nc@gmail.com

October 2021

Abstract

Constant function market makers (CFMMs) are popular mechanisms for trading of digital assets. However, they have notable drawbacks relative to conventional limit order books common to equity markets. These drawbacks include economic adverse selection costs for liquidity providers and poorer latency response for active traders. But are order books really better in a decentralized setting? Empirical evidence suggests that when decentralized systems lose liveness, CFMMs are preferred to order books by liquidity providers. Liveness losses often correlate with large price changes off-chain, so that residual on-chain liquidity can dampen liquidation cascades upon regaining liveness. We provide a theoretical underpinning for when one should prefer a CFMM to an order book. Our results demonstrate that application-level liveness interacts with consensus liveness in a non-trivial manner.

While constant function market makers (CFMMs) have grown to hundreds of billions of dollars of annualized trading volume, they have not been able to provide the latency response of traditional trading mechanisms (*e.g.* order books). The latency issue is acute for CFMMs with unbounded price support (e.g. there exists liquidity to execute a trade at every possible price) as they take longer to adjust to order flow since they cannot concentrate liquidity around highly demanded prices. On the other hand, traditional order books and bounded support CFMMs (also known as 'concentrated liquidity') allow liquidity providers to adjust their liquidity to only be executed against certain types of order flow. Historically, these mechanisms have not worked on blockchains due to the relatively high latency of confirmation (which is on the order of seconds) [Kov].

Recently, as (relatively) low confirmation latency blockchains such as Solana [Yak18] have launched, there have been a number of order book and order book like mechanisms implemented on these new chains. Moreover, the launch of Uniswap V3 has begin a process of "hybridization" between continuously priced mechanisms such as CFMMs and discrete ones like order books. Uniswap V3 localizes a CFMM's liquidity, allowing users to effectively create limit orders by only contributing liquidity within a price range. A natural question to ask is if there is a theoretically sound way to classify the differences between CFMMs and order books.

We turn to historical events to guide our framework. Recently, there was a loss of liveness on the Solana blockchain that lasted for 7 hours. While the chain was not live, the Proof of Stake (PoS) asset SOL lost over 25% in market value on centralized exchanges. Decentralized exchanges hosted on SOL, both order books and CFMMs, had prices frozen whereas off-chain exchanges (*e.g.* centralized exchanges such as FTX, Coinbase, or Binance) still allowed for trading. Upon regaining liveness, there was a chaotic rush to update existing positions. Market makers on off-chain order books had to cancel orders at the previous price (25% above fair value) while on-chain lending platforms had loans on their books that were in default that needed to be liquidated. This effectively led to a race condition to decide which orders would be confirmed upon regaining liveness. Observers could see some liquidations execute before order book market makers could pull orders [Shu]. On the other hand, Solana-based CFMMs such as Saber, had smoother operations, as arbitrageurs smoothly adjusted the on-chain price to the external market price. The Serum order book had a roughly 37% loss of liquidity upon nework restart versus the Saber CFMM's roughly 5% loss in liquidity [def]. Given this difference in outcomes in practice, a natural question to ask is, "how and when exactly do order books and CFMMs behave differently?"

In this note, we will formalize this intuition by analyzing liquidity provider losses under large price shocks for LPs of CFMMs and order books. This will provide a way for us to look at differences in losses when a price shock is caused by a tail event such as a loss of liveness in a blockchain. To do this, we will utilize the equivalence between pro-rata limit order books [FL08] and concentrated liquidity market maker such as Uniswap V3 [AZS+21a]. We will then show that, asymptotically, there are different rates of growth for the losses than an LP realizes. Using this asymptotic rate of growth, we are then able to bound the number of operations (*e.g.* blockchain transactions) needed to handle a price change of size Δp. Our results demonstrate that in the worst case, order books requires $\Omega(\Delta p)$ transactions to be executed whereas traditional CFMMs require $O(1)$ transactions to adjust the price.

When a blockchain is unable to regain liveness or has inconsistent liveness, it will have dramatically reduced throughput. In such scenarios, market mechanisms that require $O(1)$ transactions to update prices are strongly preferred to those whose number of transactions scales with price. This result demonstrates, at least asymptotically, that CFMMs with non-concenrated liquidity outperform order books in such events. In Appendix A, we illustrate a framework for a more direct comparison between consensus protocol liveness and liquidity provider losses. This suggests that an application-specific formulation of liveness that quantifies the number of transactions needed to return an application to a safe state is necessary for DeFi protocols.

1 Preliminaries

1.1 Constant function market makers

A *constant function market maker* is a contract that holds some amount of *reserves* $R, R' \geq 0$ of two assets and has a *trading function* $\psi : \mathbf{R}^2 \times \mathbf{R}^2 \to \mathbf{R}$. Traders can then submit a *trade*

(Δ, Δ') denoting the amount they wish to tender (if negative) or receive (if positive) from the contract. The contract then accepts the trade if

$$\psi(R, R', \Delta, \Delta') = \psi(R, R', 0, 0),$$

and pays out (Δ, Δ') to the trader.

Curvature. We briefly summarize the main definitions and results of [AEC20] here. Suppose that the trading function ψ is differentiable (as most trading functions in practice are), then the marginal price for a trade of size Δ is

$$g(\Delta) = \frac{\partial_3 \psi(R, R', \Delta, \Delta')}{\partial_4 \psi(R, R', \Delta, \Delta')}.$$

Here ∂_i denotes the partial derivative with respect to the ith argument, while Δ' is specified by the implicit condition $\psi(R, R', \Delta, \Delta') = \psi(R, R', 0, 0)$; *i.e.*, the trade (Δ, Δ') is assumed to be valid. Additionally, the reserves R, R' are assumed to be fixed. The function g is known as the *price impact* function as it represents the final marginal price of a positive sized trade. When there are fees, one can show that $g^{fee}(\Delta) = \gamma g(\gamma \Delta)$ where $1 - \gamma$ denotes the percentage fee. We say that a CFMM is μ-*stable* if it satisfies

$$g(0) - g(-\Delta) \leq \mu \Delta$$

for all $\Delta \in [0, M]$ for some positive M. This is a linear upper bound on the maximum price impact that a bounded trade (bounded by M) can have. Similarly, we say that a CFMM is κ-liquid if it satisfies

$$g(0) - g(-\Delta) \geq \kappa \Delta$$

for all $\Delta \in [0, K]$ for some positive K. One important property of g is that it can be used to compute Δ' [AEC20, §2.1]:

$$\Delta' = \int_0^{\Delta} g(t) dt \tag{1}$$

Simple methods for computing some μ and κ in common CFMMs are presented in [AEC20, §1.1].

Two-sided bounds. We can define similar upper and lower bounds for $g(\Delta) - g(0)$, with constants μ' and κ', which hold when the trades Δ are in intervals $[0, M']$, $[0, K']$, respectively. For the remainder of this paper, we will refer to μ-stability as the upper bound for both $g(0) - g(-\Delta)$ and $g(\Delta) - g(0)$, and similarly for κ-liquidity. More specifically, given μ, μ', we say that a CFMM is symmetrically μ''-stable if

$$|g(\Delta) - g(0)| \leq \mu |\Delta|,$$

when $-M \leq \Delta \leq M'$, and symmetrically κ'' stable if

$$|g(\Delta) - g(0)| \geq \kappa |\Delta|.$$

when $-K \leq \Delta \leq K'$. From the above, it suffices to pick $\mu'' = \min\{\mu, \mu'\}$ and $\kappa'' = \min\{\kappa, \kappa'\}$.

Note that any two-sided μ-stable and κ-liquid market maker is automatically η-stable and η-liquid for $\eta = \max(\kappa, \mu)$. An η-liquid and η-stable impact function is *bi-Lipschitz* and admits an inverse $g^{-1}(p)$ that is also bi-Lipschitz [CMN19]. In particular, if g is η bi-Lipschitz, then $g^{-1}(p)$ is $\frac{1}{\eta}$ bi-Lipschitz:

$$\frac{1}{\mu}p \leq |g^{-1}(p) - g^{-1}(0)| \leq \frac{1}{\kappa}p$$

Bounded Support CFMMs and Order Books. CFMMs with bounded price support (also known as "concentrated liquidity") were first introduced by Uniswap V3 [AZS+21b]. Such CFMMs were also shown to be able to replicate a variety of concave payoff functions such as covered calls and variance swaps [AEC21b, AEC21c]. Any bounded support mechanism $\mathcal{M}$ is defined by a triple, $\mathcal{M} = (\mathcal{T}, L, g)$ where

- $\mathcal{T} = \{(a_i, b_i) : a_i < b_i\} \subset \mathbf{R}_+ \times \mathbf{R}_+$ is a totally ordered set of ticks that partition the price space, e.g. $\bigcup_{\{a_i,b_i\}\in\mathcal{T}}[a_i, b_i) = [0, \infty)$
- L is the amount of liquidity at each tick, represented as a function $L : \mathcal{T} \to \mathbf{R}$
- g is the price impact function of the market maker

Since the set of ticks is totally ordered, we will refer to $(L_i, L_i') = L(T_i)$ where T_i is the ith tick in the ordering of $\mathcal{T}$, L_i is the liquidity of the risky asset and L_i' is the liquidity of the numéraire.[1] Moreover, we define the *size* of a tick $T = (a, b)$ to simply be $|T| = b - a$

Uniswap V3 allows LPs to deposit coins into a set of CFMM pools specified by a price range $[a, b] \subset \mathbf{R}$. The mechanism $\mathcal{M}_{V3}$ ensures that an LP's coins are only used when the prices offered by the market maker are in the range $[a, b]$. More precisely, Uniswap V3 generates a series of logarithmically-spaced ticks $T_i = [b^i, b^{i+1}]$ for a base $b > 1$. Here we assume that b^i is in units of numéraire, so that when the price goes up from b^i to b^{i+1}, it costs more numéraire to purchase the risky asset. At each tick, there is a CFMM pool that utilizes the constant product formula [AEC21a] for executing increasing trades when the price p is within the band $[b^i, b^{i+1})$. When an LP deposits liquidity for a price range $[a, b]$, it is added to all pools associated to ticks T_i such that $T_i \cap [a, b] \neq \emptyset$. Define the support of any bounded support CFMM $\mathcal{M}$ (including $\mathcal{M}_{V3}$), $\mathsf{supp}(\mathcal{M})$, to be the set of ticks that have non-zero liquidity, *e.g.*

$$\mathsf{supp}(\mathcal{M}) = \{t \in \mathcal{T} : L(t) > 0\}$$

One can view Uniswap V3 as a pro-rata order book. Suppose that $(L_i, L_i') \in \mathbf{R}_+$ is the aggregated quantity of liquidity provided in the risky asset L_i and numéraire L_i' in the pool

[1] To simplify narration, we assume that one asset is risky and the other is a numéraire. However, our construction works for any 2-asset CFMM

associated to T_i. When a trade demands Δ units of risky asset to trade within T_i (*e.g.* the price before and after the trade is in T_i), liquidity is adjusted to $L_i - \Delta$ and $L_i + \Delta'$, where Δ' is as in (1). On the other hand, if a trade of size Δ crosses a tick boundary, some fraction $\Delta_i \leq \Delta$ will be executed against each tick T_i such that $\Delta = \sum_i \Delta_i$. When a trade of size Δ_i is executed against T_i, all LPs with $T_i \subset [a, b]$ will earn a pro-rata share of fees generated by the trade of size Δ_i. Therefore, we can view the liquidity in each tick T_i as equivalent to a price level in a pro-rata order book [FL08]. In particular, a Uniswap V3 pool can replicate any liquidity profile of an order book provided that liquidity providers *rebalance* or repeatedly update their ranges as a function of price [Cla21, NRMP21, Fri21].

This equivalence between pro-rata order books and Uniswap V3 implies that we can analyze a sequence of n bounded liquidity pools versus a single large pool to compare order books to (traditional) CFMMs. We note that the pro-rata nature of the order book can lead to different arbitrage strategies than FIFO (or time priority) order books [GP15]. However, these distinctions become less important as $|T_i| \to 0$. For the the rest of this paper, we will interchangeably refer to pro-rata order books and sequences of bounded support CFMMs (like Uniswap V3).

2 Asymptotic Liquidity Provider Returns

Prior theoretical work on analyzing liquidity provider returns has focused on the unbounded support CFMM setting [AEC20, §2]. In order to analyze the behavior of CFMMs and order books under a large price shock (such as one caused by a loss of liveness in the underlying chain), we need to consider liquidity provider returns for bounded liquidity market markers. We will do this in two steps: first for Uniswap's constant product curve and then for a general, positive curvature CFMM. Uniswap, as an example, will illustrate some of the necessary properties for providing non-trivial bounds.

Portfolio Value. Recall that the *portfolio value* is the net present value of the assets owned by an LP in numéraire terms. For a two asset CFMM with risky and numéraire reserves (R, R'), the portfolio value is $V(p_0) = p_0 R + R'$, where p_0 is the quoted price of the risky asset in numéraire terms [AEC21c]. Given a trade of size Δ and an initial price p_0, we can write the change in portfolio value in terms of the price impact function g as [AEC20, §3.1]

$$PV(\Delta) = g(\Delta)(R - \Delta) + (R' + \Delta')$$

where $g(\Delta) \geq g(0) = p_0$ and Δ' is as per (1). We will analyze the change in portfolio value when there is a large price change from p_0 to p with $p \gg p_0$. Our main claim is that CFMMs with unbounded support have a lower asymptotic change in portfolio value than those with bounded support. To show this, we will analyze the change in portfolio value

function $\delta V(p) = PV(g^{-1}(p)) - V(p_0)$. Plugging this into the formula for $PV(\Delta)$ yields

$$\begin{aligned}\delta V(p) &= p(R - g^{-1}(p)) + (R' + \Delta') - p_0 R - R' \\ &= p(R - g^{-1}(p)) + \Delta' - p_0 R \\ &= p(R - g^{-1}(p)) - p_0 R + \int_0^{g^{-1}(p)} g(t)dt \end{aligned} \tag{2}$$

where the last line uses (1). Note that since g is increasing on $[0, M]$,

$$\Delta' \in [g(0)g^{-1}(p), g(g^{-1}(p))g^{-1}(p)] = [p_0 g^{-1}(p), pg^{-1}(p)]$$

this allows us to lower bound (2):

$$\delta V(p) \geq p(R - g^{-1}(p)) - p_0 R + p_0 g^{-1}(p) = (p - p_0)(R - g^{-1}(p)) \tag{3}$$

Note that as per §1.1, g^{-1} exists provided that g is two-sided μ-stable and κ-liquid, which we will assume throughout this section.

Our goal will be to study the rate of growth of $\delta V(p)$ as $p \to \infty$ or $p \to 0$, which will be termed *asymptotic portfolio value*. The asymptotic portfolio value represents the rate of growth of profits or losses (PNL) for LPs under big price shocks. We will mainly be interested in trying to upper and lower bound $\delta V(p)$ for large and small p. Our first result will demonstrate that LPs in bounded support CFMMs have different asymptotic portfolio values.

Uniswap. The no-fee price impact function for Uniswap's constant product curve is [ZCP18, AEC20]

$$g_{uni}(\Delta) = \frac{k}{(R - \Delta)^2}$$

where k is the product constant. We will continue our analysis under the no-fee case, but note that our analysis can be applied to fees as per [AEC20, App. B]. To compute $\delta V_{uni}(p)$, we will first need to compute $g_{uni}^{-1}(p)$. Simple algebra shows that

$$g_{uni}^{-1}(p) = R - \sqrt{\frac{k}{p}}$$

Substituting this into (3) gives

$$\delta V_{uni}(p) \geq (p - p_0)\sqrt{\frac{k}{p}} = \sqrt{kp} - \sqrt{\frac{kp_0^2}{p}} \geq \sqrt{kp} - \sqrt{k' p_0} \tag{4}$$

where the last inequality comes from $\frac{p_0}{p} \in (0, 1)$. Note that result matches [AC20, §2.5], where the authors explicitly computed $PV(p) = \Theta(\sqrt{p})$. However, using equation (3) to bound δV is generic enough to handle other market makers.

Uniswap V3. Since Uniswap V3 uses the constant product formula, we can use (4) to bound for $\delta V_{V3}(p)$. When p is in $\mathsf{supp}(\mathcal{M}_{V3})$, we receive the same δV as the generic constant product curve. On the other hand, when $p \notin \mathsf{supp}(\mathcal{M}_{V3})$, $\delta V_{V3}(p)$ is necessarily linear in p as an LP is only holding one asset. Let $p_+ = \sup_{(a_i,b_i)\in\mathsf{supp}(\mathcal{M}_{V3})} b_i$ be the maximum price with non-zero liquidity in a Uniswap V3 pool. Similarly, let $p_- = \inf_{(a_i,b_i)\in\mathsf{supp}(\mathcal{M}_{V3})} a_i$ be the maximum price quoted. Then we have:

$$PV_{V3}(p) = \begin{cases} \sqrt{kp} & p \in \mathsf{supp}(\mathcal{M}_{V3}) \\ p - p_+ + \sqrt{kp_+} & p > p_+ \\ p_- - p + \sqrt{kp_-} & p < p_- \end{cases}$$

This implies that $|\delta V_{V3}(p)| = \Omega(p)$ as $\mathsf{supp}(\mathcal{M}_{V3})$ is bounded. Bounded support LPs don't have the benefit of worst case asymptotically square root portfolio loss, unlike unbounded support LPs.

The quadratic gap in portfolio value (which represents numéraire-denominated PNL) between Uniswap V2 and V3 represents one notable distinction between order books and unbounded support CFMMs. Suppose a blockchain (like Solana), loses liveness and p starts to drift outside of $\mathsf{supp}(\mathcal{M}_{V3})$. When this happens, it is possible upon regaining liveness, for LPs to suddenly start realizing the square of their 'impermanent loss' (opportunity cost). Unbounded support CFMMs, however, have LPs with consistent asymptotic losses across their price range.

Curved CFMMs. We can use eq. (3) to lower bound the asymptotic portfolio value for curved CFMMs with unbounded support. Recall that by definition, $g(0) = p_0$, so $g^{-1}(p_0) = 0$. For unbounded support CFMMs with positive curvature ($\kappa > 0$) and reserves R, it can be shown that [AEC20]:

$$\lim_{r\to R} g(r) \to \infty \qquad\qquad \lim_{r\to R'} g(-r) = 0$$

Therefore, $\mathbf{Dom}(g) \subseteq [-R', R]$ and subsequently $\mathbf{Range}(g^{-1}) \subset [-R', R]$. Therefore the function $p \mapsto R - g^{-1}(p)$ has a range of $[0, R + R']$. Moreover, the previous limits imply that $\lim_{p\to\infty} R - g^{-1}(p) = 0$ or equivalently, $\lim_{p\to\infty} g^{-1}(p) = R$. This shows that $R - g^{-1}(p) = o(1)$ and therefore $(p - p_0)(R - g^{-1}(p)) = o(p)$. This demonstrates that the lower bound (3) on asymptotic portfolio value for curved, unbounded support CFMMs is sublinear.

We can show a similar upper bound. Recall from (2) that $\delta V(p) = p(R - g^{-1}(p)) + \Delta' - p_0 R$ with $\Delta' \in [p_0 g^{-1}(p), p g^{-1}(p)]$. We can bound the norm of $\delta V(p)$ as follows:

$$\begin{aligned} |\delta V(p)| &\leq |p(R - g^{-1}(p)) + p g^{-1}(p) - p_0 R| \\ &\leq |p(R - g^{-1}(p))| + |p g^{-1}(p) - p_0 R| \\ &\leq 2|p(R - g^{-1}(p))| \end{aligned}$$

From the above discussion, $p(R - g^{-1}(p)) = o(p)$, which implies that the change in portfolio value is sublinear.

For a curved CFMM with bounded support, $\mathcal{M}$, we instead have

$$\lim_{r \to R} g(r) = p_+(\mathcal{M}) \qquad \lim_{r \to R'} g(-r) = p_-(\mathcal{M})$$

where p_+ and p_- are defined analogously to Uniswap V3. For $p > p_+$, we have $g(p) = g(R) + (p - g(R)) < \infty$, so $\lim_{p \to \infty} g^{-1}(p) \to c \neq 0$. This ensures that $R - g^{-1}(p) = \Omega(1)$ implying worst case linear loss, $\delta V(p) \geq Cp$, like Uniswap V3.

2.1 Approximation Error

Order books and collections of bounded support CFMMs (like Uniswap V3) can approximate any price impact g that an unbounded support CFMM expresses. However, due to the tick sizes, there is some approximation error. Concretely, this approximation error arises when one tries to approximate the integral in eq. (2) by a sum of smaller integrals:

$$\int_0^{g^{-1}(p)} g(t)dt \approx \sum_{T_i \in \mathsf{supp}\, \mathcal{M} \cap [0, g^{-1}(p)]} \int_{T_i} g_i(t)dt \tag{5}$$

where $g_i(t)$ is the price impact of the ith bounded support market maker (e.g. for tick T_i). A natural question to ask is: how well can we approximate an unbounded support CFMM price schedule with a sequence of bounded support CFMMs. In particular, the main question is how many summands do we need on the right hand side of eq. (5) to ensure that

$$\left| \int_0^{g^{-1}(p)} g(t)dt - \sum_{T_i \in \mathsf{supp}\, \mathcal{M} \cap [0, g^{-1}(p)]} \int_{T_i} g_i(t)dt \right| < \epsilon$$

The number of summands is controlled by the tick size T_i and the precise impact function g_i.

Approximations of this form are known in the numerical analysis literature as *quadrature rules*. For a function of bounded variation, which all price impact functions are [AEC20], one can use Gaussian quadrature bounds to show that the best possible approximation is proportional to $\frac{TV(g)}{n}$, where TV is the total variation [FP91]. Monotone functions f on a compact interval $[a, b]$ have a total variation equal to $TV(f) = f(b) - f(a)$. Therefore, the error we can approximate an unbounded support CFMM price impact by an order book is

$$\left| \int_0^{g^{-1}(p)} g(t)dt - \sum_{T_i \in \mathsf{supp}\, \mathcal{M} \cap [0, g^{-1}(p)]} \int_{T_i} g_i(t)dt \right| < \frac{C(g(g^{-1}(p)) - g(0))}{|\,\mathsf{supp}\, \mathcal{M} \cap [0, g^{-1}(p)]|} = \frac{C(p - p_0)}{n} \tag{6}$$

As such, we need to increase the number of ticks that we sum over as a function of the maximum price interval we want to replicate. To ensure a constant upper bound on approximation error over many prices, this implies the tick size needs to decrease as $O\left(\frac{1}{p}\right)$.

This suggests that bounded support CFMMs and order books need to adjust their tick size in response to a large price shock. In the appendix, we connect this to the notion of liveness from consensus protocols. As it turns out, order books require $O(p - p_0)$ update operations to rebalance the tick size to ensure constant approximation error. In Appendix A, we provide a way of formalizing the notion of "how many operations / transactions does a market making mechanism need to adjust liquidity to a price change $p - p_0$?".

3 Conclusion

In this note, we demonstrated that the worst case loss for LPs pro-rata order books and concentrated liquidity CFMMs is asymptotically worse than that of unbounded CFMMs. This helps empirically support some of the observed behavior on the Solana blockchain post loss of liveness. Our results also demonstrate that there is a clear connection between the transaction complexity for on-chain prices to synchronize with those off-chain and the support of a market making mechanism. Future work will illustrate how the traditional consensus protocol notion of liveness is intertwined with the ability of an on-chain mechanism to synchronize prices. This suggests that further analysis into how decentralized applications interact with the consensus mechanisms and replicated state machines they are run on is necessary for providing realistic safety guarantees.

References

[AC20] Guillermo Angeris and Tarun Chitra. Improved Price Oracles: Constant Function Market Makers. In *Proceedings of the 2nd ACM Conference on Advances in Financial Technologies*, pages 80–91, New York NY USA, October 2020. ACM.

[AEC20] Guillermo Angeris, Alex Evans, and Tarun Chitra. When does the tail wag the dog? curvature and market making. *arXiv preprint arXiv:2012.08040*, 2020.

[AEC21a] Guillermo Angeris, Alex Evans, and Tarun Chitra. A note on privacy in constant function market makers. *arXiv preprint arXiv:2103.01193*, 2021.

[AEC21b] Guillermo Angeris, Alex Evans, and Tarun Chitra. Replicating market makers. *arXiv preprint arXiv:2103.14769*, 2021.

[AEC21c] Guillermo Angeris, Alex Evans, and Tarun Chitra. Replicating monotonic payoffs without collateral. 2021.

[AZS+21a] Hayden Adams, Noah Zinsmeister, Moody Salem, River Keefer, and Dan Robinson. Uniswap v3 core. 2021.

[AZS+21b] Hayden Adams, Noah Zinsmeister, Moody Salem, River Keefer, and Dan Robinson. Uniswap v3 core. Technical report, Tech. rep., Uniswap, 2021.

[Cla21] Joseph Clark. The replicating portfolio of a constant product market with bounded liquidity. *Available at SSRN*, 2021.

[CMN19] Ştefan Cobzaş, Radu Miculescu, and Adriana Nicolae. *Lipschitz functions*, volume 2241. Springer, 2019.

[def] Defi dashboard.

[FL08] Jonathan Field and Jeremy Large. Pro-rata matching and one-tick futures markets. Technical report, CFS Working Paper, 2008.

[FP91] Klaus-Jürgen Förster and Knut Petras. Error estimates in gaussian quadrature for functions of bounded variation. *SIAM journal on numerical analysis*, 28(3):880–889, 1991.

[Fri21] Robin Fritsch. Concentrated liquidity in automated market makers. *arXiv preprint arXiv:2110.01368*, 2021.

[GP15] Fabien Guilbaud and Huyên Pham. Optimal high-frequency trading in a pro rata microstructure with predictive information. *Mathematical Finance*, 25(3):545–575, 2015.

[Kov] Eduard Kovacs. U.s. government asks victims of 2017 etherdelta hack to come forward.

[NRMP21] Michael Neuder, Rithvik Rao, Daniel J Moroz, and David C Parkes. Strategic liquidity provision in uniswap v3. *arXiv preprint arXiv:2106.12033*, 2021.

[Shu] Camomile Shumba. Solana says it is back up and running after a surge in transactions caused the network to crash the day before.

[Yak18] Anatoly Yakovenko. Solana: A new architecture for a high performance blockchain v0. 8.13. *Whitepaper*, 2018.

[ZCP18] Yi Zhang, Xiaohong Chen, and Daejun Park. Formal specification of constant product (xy=k) market maker model and implementation. 2018.

A Market Maker Liveness

Recall that a decentralized consensus protocol is defined to satisfy liveness if the network eventually processes all transactions submitted to the network. Liveness guarantees are contingent on various assumptions about network quality, with the three main assumptions being synchrony, partial synchrony, and asynchrony. If we assume there is a minimum time resolution that all nodes can measure (which is usually enforced to be a slot or block via cryptographic means), synchrony assumes that all messages sent by users are received by honest nodes within a single block or slot. On the other hand, partial synchrony assumes

that there exists a finite value $\tau > 0$ such that all nodes receive all messages sent to the network within τ blocks or slots. Partial synchrony can be subdivided into two further categories by requiring either all nodes to know the uniform upper bound τ or for it to only be known to be finite. Finally, asynchrony assumes that there is an no bound on the time that it takes for nodes can receive messages.

We focus on making an analogue of partial synchrony for decentralized exchange mechansism, both CFMMs and order books. Our definition specializes the traditional decentralized consensus definition to be *liquidity aware*. In particular, a decentralized trading mechanism can only achieve liveness up to some function of the liquidity held on the exchange. Moreover, the atomicity of how trades are executed is a crucial component to capture in any definition of liveness. Order books are generically non-atomic: if there are n resting limit orders on an order book and an aggressive trade crosses with $\Omega(n)$ orders, then the order book has to write $\Omega(n)$ state to the blockchain. On the other hand, when a CFMM receives an order of size Δ, provided that $\Delta < R$, where R is the CFMMs reserves or liquidity, then it will always execute the trade in $O(1)$ time (relative to R). This distinction plays a big role on blockchains, where there is a scarcity of slots for a transaction to be executed within.

First, we will formalize the state variables contained within CFMMs and order books. For CFMMs, this is relatively straightforward:

1. $\varphi : \mathbf{R}^2 \to \mathbf{R}$: Trading function that ensures an invariant is kept constant before and after a trade

2. $R \in \mathbf{R}^2$: Reserves or liquidity at each time step

For order books, we have two main functionalities that are more complex than their CFMM counterparts:

1. $\mathsf{BookDepth}(s)$: Total resting liquidity on side $s \in \{\mathsf{buy}, \mathsf{sell}\}$.

2. $\mathsf{NumOrders}(s)$: Total number of resting passive orders on side s

A CFMM order $\mathcal{O}$ is defined as a pair $\mathcal{Q} = (\Delta, \eta) \in \mathbf{R} \times [0, 1]$, where Δ is the desired quantity to be bought or sold and η is the *slippage limit*. Suppose that we have a CFMM that is initially at reserve R_0 and has price impact function $g(\Delta, R_0)$. We say that a trade $\mathcal{O}$ is *valid* if

$$\frac{|g(\Delta, R_0) - g(0, R_0)|}{g(0, R_0)} \leq \eta$$

On the other hand, an order book order $\mathcal{O}^{CLOB} = (\Delta, s)$ is simply a sided quantity and executes if $\Delta < \mathsf{BookDepth}(s)$. Note that $\mathsf{BookDepth}(s)$ is dynamic (e.g. market makers will constantly add, cancel, and modify resting limit orders) and as such, is significantly harder to compute than $g(\Delta, R_0)$.

Our goal is to unify a notion of atomicity (given liquidity constraints) for these two mechanisms. Given a notion of atomicity, it will be easy to extend to a notion of liveness that resembles what one sees in traditional consensus mechanisms. The main parameters for computing atomicity are:

- $N_{\max}$: The maximum number of trades that can be executed in a block
- L: The liquidity of the market making mechanism (a simple function of either $\mathsf{BookDepth}(s)$ or R)
- $n(s)$: Number of resting orders on a side s, where $n(s) = 1$ for all CFMMs (since the liquidity pool can be viewed as a single buy or sell order)

We are now ready to define $(k, \mathcal{L})$-atomicity:

Definition 1. *A market making mechanism $\mathcal{M}$ is $(k, \mathcal{L})$-**atomic** if $\forall \Delta < \mathcal{L}$ it at most k transactions to process Δ, regardless of $n(s)$ provided that $L > \mathcal{L}$*

One sample construction of a $(k, \mathcal{L})$-atomic mechanism is an order book that has the following properties:

1. Limit orders must have size $\geq \lceil \frac{\mathcal{L}}{k} \rceil$
2. If the mempool has passive and active orders, active orders are executed before any state change to passive orders (cancels, adds, modifies).

Note that the second condition ensures that $\mathsf{BookDepth}(s) > \Delta$ provided that Δ is less than the previous known book depth (as market makers cannot cancel in front of a big order). The less than k transactions condition is guaranteed by the size of the order, since it can only take k time steps to process the maximum size order.

Using this definition, a CFMM with reserves R is $(1, (1-\epsilon)R)$-atomic, whereas an order book is $(\mathbf{NumOrders}(s), \mathsf{BookDepth}(s))$-atomic on each side. Note that the reason for the $(1-\epsilon)$ factor is because any CFMM with unbounded support cannot empty its reserves on a single trade. CFMMs with bounded support (often called "concentrated liquidity" in cryptocurrency parlance) will achieve $(1, R)$-atomicity provided that the trade is less than the liquidity over the bounded range. Order book's atomicity clearly fluctuates with order size and resting liquidity and also has a notion of execution time that fluctuations (unlike CFMMs).

Properties of Atomicity. From the above definition, it is clear that if a mechanism $\mathcal{M}$ is $(k, \mathcal{L})$-atomic for some $k > 0$ then it is also $(k', \mathcal{L})$-atomic for any $k' > k$. This implies that for a fixed amount of liquidity $\mathcal{L}$, there exists a minimum

$$k^*_{\mathcal{M}}(\mathcal{L}) = \min\{k \geq 1 : \mathcal{M} \text{ is } (k, \mathcal{L})\text{-atomic}\}$$

such that $\mathcal{M}$ is $(k^*(\mathcal{L}), \mathcal{L})$ atomic. Similarly, a $(k, \mathcal{L})$-atomic mechanism is $(k, \mathcal{L}')$-atomic for any $\mathcal{L}' < \mathcal{L}$. This implies that for fixed $k \geq 1$, there is a maximum liquidity

$$\mathcal{L}^*_{\mathcal{M}}(k) = \max\{\mathcal{L} > 0 : \mathcal{M} \text{ is } (k, \mathcal{L})\text{-atomic}\}$$

that can be achieved for a fixed k. For brevity, we will elide the subscript $\mathcal{M}$ when it is clear from context. For both k^* and $\mathcal{L}^*$, we define them to be infinite if there exists no value of k or $\mathcal{L}$, respectively, that is $(k, \mathcal{L})$-atomic

Given that CFMMs with bounded support can be composed — *e.g.*$\mathcal{L}_1$ units of liquidity between prices $[a, b)$ and $\mathcal{L}_2$ units of liquidity between prices $[b, c)$) — it is natural to inquire about how atomicity composes. Suppose that $\mathcal{M}$ consists of a $(k_1, \mathcal{L}_1)$-atomic pool and a $(k_2, \mathcal{L}_2)$ atomic pool. Consider a trade of size $\Delta \in (\mathcal{L}_1, \mathcal{L}_1 + \mathcal{L}_2)$. Processing such a trade will take at least $\max(k_1(\mathcal{L}_1), k_2(\mathcal{L}_2)$ transactions, since we take at least k_i transactions to trade against all of the liqudity in pool 1 or pool 2. The remaining trade size, $\Delta - \mathcal{L}_1$ or $\Delta - \mathcal{L}_2$ is executed against a $(k_2, \mathcal{L}_2)$-atomic or $(k_1, \mathcal{L}_1)$-atomic pool, respectively. Therefore, we have

$$k^*(\mathcal{L}_1 + \mathcal{L}_2) \geq k_1(\mathcal{L}_1) + k_2(\mathcal{L}_2) \geq k^*(\mathcal{L}_1) + k^*(\mathcal{L}_2)$$

— that is, k^* is *superadditive* in its second argument. Similarly, we can show that $\mathcal{L}^*(\mathcal{M}, k_1 + k_2)$ is *subadditive* in its second argument.

This illustrates that composition of atomic operations is at least superadditive. Superadditivity implies that concentrated liquidity pools make a trade-off: the more pools that are needed to service a fixed amount of liquidity, the more transactions are necessary to process larger trades. In Uniswap V3, this is more clearly depicted by the fact that crossing a tick boundary requires more transactions than doing two independent trades at separate ticks.

The opposing superadditivity and subadditivity properties of k^* and $\mathcal{L}^*$ suggests a saddle point algorithm for trading off liquidity versus liveness:

- Start with an initial liquidity level $\mathcal{L}_0$
- Compute an optimal atomicity $\hat{k}_i = k^*(\mathcal{L}_{i-1})$
- Compute an optimal liquidity $\hat{\mathcal{L}}_i = \mathcal{L}^*(k_i)$

After some number of rounds R, the hope is that the sequence converges in the sense that $\forall \epsilon > 0, \exists r > R$ such that $\hat{k}_{r+1} = \hat{k}_r$ and $|\mathcal{L}_r - \mathcal{L}_{r+1}| < \epsilon$. The output of this algorithm will (hopefully) be a sort of analogue of a Nash equilibria: changing either k or $\mathcal{L}$ will not result in an improvement in atomicity.

The precise value or objective function that is constructed depends on the market making mechanism. For instance, for unbounded support CFMMs, we have $(1, (1-\epsilon)R)$-atomicity, which cannot be improved by this procedure as k is minimize and $\mathcal{L}$ is ϵ-close to a maximum. On the other hand, CFMMs with n pools where each pool is $(1, R_i)$-atomic will start off with $k^* \geq n$ and $\mathcal{L}_0 = \sum_i R_i$ and have a potentially large rebalance across pools as $n \to \infty$. Order books will behave similarly, as they start off as $(\mathsf{NumOrders}(s), \mathsf{BookDepth}(s))$-atomic.

Liveness. We can now define liveness in terms of atomicity:

Definition 2. *A market making mechanism $\mathcal{M}$ achieves $\mathcal{L}$-liveness if $k^*(\mathcal{L}) < N_{\max}$*

Note that a non-concentrated CFMM with reserves R achieves $(1-\epsilon)R$-liveness, whereas order books and concentrated liquidity CFMMs can achieve much lower liveness than $\mathsf{BookDepth}(s)$. By locking up liquidity (e.g. such as in our sample construction), an order book (or a hybrid order book and CFMM) mechanism can improve its level of liquidity liveness.

Moreover, using this definition, we can use (6) to provide a more precise bound on k for order books. In particular, given a fixed liquidity level $\mathcal{L}$, we need $O(p-p_0)$ pools (e.g. $O(\frac{1}{p})$ tick size) to achieve low quadrature error. This implies that $k^*(\mathcal{L}) = \Omega(p - p_0)$, where p is an upper bound on the maximum price impact expected. Note that if $\frac{p-p_0}{N_{\max}} \gg 1$ then we effectively cannot service demand to that price level given existing block size.