The Conceptual Flaws of Constant Product Automated Market Making

Andreas Park[*]

June 8, 2021

Abstract

Blockchain-based decentralized exchanges are a pre-requisite and the backbone of decentralized finance. They fall into two broad categories: decentralized limit order books where an order is a smart contract registered on the blockchain, and swap exchanges where prices are set by a deterministic automated market making rule. The most common form of the latter is the constant product rule where relative prices of crypto assets are determined by iso-liquidity curves. Although this pricing rule is simple, its use is conceptually problematic and gives rise to persistent arbitrage opportunities when there are multiple competing trading systems. It also allows intrinsically profitable front-running opportunities. A traditional market maker pricing rule, on the other hand, does not suffer from these flaws. Calibrated to a less liquid but frequently used trading pair on UniSwap, 14% of transactions see an implicit theoretical excess cost of at least 50bps, which is orders of magnitude larger than the common trading costs for this pair on centralized exchanges.

[*] University of Toronto andreas.park@rotman.utoronto.ca. I am grateful for financial support from the *Global Risk Institute* and the *Social Sciences and Humanities Research Council* and to Cam Harvey, Julien Prat, Agostino Capponi, and Fas Qureshi for insightful discussions and feedback.

Permissionless blockchains such as Ethereum are a technological infrastructure that allows "decentralized" digital resource transfers, a possibly disruptive threat to the traditional world of finance. Yet in its first 5 years of existence, the Ethereum blockchain barely made a dent to legacy finance. Moreover, most of the transfers of the trillion dollars worth of blockchain assets occurred, somewhat ironically, on centralized, "off-chain" exchanges. Using these is risky: not only are these intermediaries largely unregulated, users also have to transfer custody of their tokens to the exchange and are then exposed to the risk of hacking or outright theft.[1]

Things changed, however, in the summer of 2020 with the arrival of decentralized *swap* exchanges such as UniSwap, SushiSwap, or Balancer. By now these venues process more trading volume than many prominent centralized exchanges, including the recently IPO-ed Coinbase, with a core algorithm of just over 200 lines of code.

What sets swap exchanges apart and makes them interesting is their underlying organization. Traditional financial markets rely on limit order books or on networks of market makers that investors access through various intermediaries and that require several separate systems to sync. In contrast, using a swap exchange is a three-click process: Traders connect their browser crypto-wallet to the swap exchange app, enter desired the amount, sign the transaction with their wallet, and the trade gets executed on the Ethereum blockchain. Functional simplicity aside, the economically relevant promise and possible genius of decentralized swap exchanges is that they use the smart contracts functionality of the blockchain to *aggregate* and *automate* the provision of

[1] Examples are the Mt. Gox hack, the mysterious death of QuadrigaX's founder Gerald Cotton, and the outright theft by the founder of crypto-exchange Thodex. There are blockchain based exchanges, but these commonly simply copy traditional limit order market with their high message volumes, and that makes them cumbersome, expensive, and resource intensive to use; an example is EtherDelta.

liquidity. Liquidity providers deposit and pool their assets in a swap exchange smart contract which then allows others to trade against this aggregate set of assets. Liquidity providers receive a fee in return, while retaining their exposure to the underlying assets.

There is a problem, however. All of the swap exchanges use a particular hard-coded function to determine the price for accessing the liquidity of this contract, and this pricing function is economically flawed. In this paper, I outline how this function intrinsically creates disincentives and arbitrage opportunities that impose excess costs on users and liquidity providers. However, I also provide a remedy in the form of a pricing approach that is based on a canonical market making model from the microstructure literature that overcomes all these issues.

The first mention of decentralized market making is in a 2016 Reddit post by Ethereum's Vitalik Buterin. Martin Koppelmann of the Gnosis project later expands on this idea and proposes a constant product automated market making (henceforth: CPAMM) pricing scheme. All swap exchanges to date all use variations of this pricing scheme. Namely, under CPAMM, liquidity for a pair of tokens A and B is arranged via a smart contract into which liquidity providers deposit X units of tokens A and Y units of tokens B. The ratio Y/X is the implicit marginal price of an A token measured in B tokens. Very commonly, one of the tokens, say B, is a U.S. dollar-pegged stablecoin, i.e., a token that has a price of approximately one dollar; examples are Tether/USDT, USDC, and DAI. Using a stablecoin as the unit of measurement, the exchange rate Y/X is the dollar-price for an infinitesimal amount of A tokens. Pricing for non-zero quantities is such that the contract keeps liquidity invariant at a level $X \times Y = c$ for some constant c. This means that if someone wants to buy x of the A tokens, she has to pay with y of the B tokens where y is such that $c = (X - x) \times (Y - y)$. In the language of economics, tokens are priced along an iso-liquidity curve (points of same level of liquidity).

The first fundamental problem arises because of the operation of blockchain settlement which creates the possibility of front-running. After a contract submits a swap trade, it sits in the so-called mempool of authenticated but not settled transactions and awaits settlement, i.e., the inclusion in a new block by a miner.[2] Crucially, the trade price is determined by the CPAMM formula and it is driven by the number the tokens that are in the contract at the time of settlement — not at the time when the trade was submitted. Any user with access to the mempool can now do the following: submit the same trade but with a higher mining fee (so that the trade will have higher priority of getting included in the next block) and then submit a second trade with a lower fee to close or reverse the position.

The possibility of this front-running is a feature of most blockchains. Although it is annoying for an investor, front-running is only a concern if the pricing algorithm makes it intrinsically profitable. The first main point of this paper is that this is *always* the case for CPAMM pricing.[3] I then show that there is another way: with a pricing function that follows a canonical market maker model from the financial market microstructure literature (Biais (1993)), mempool front-running is *never* profitable.

[2] The concern that I describe in this paper applies whenever transactions are visible to anyone prior to settlement, and when the order of inclusion in a block is not according to a timestamp. This is the case for most proof-of-work blockchains such as Ethereum or Conflux, and likely also applies to many proof-of-stake blockchains.

[3] I am by no means the first to highlight this problem — see, e.g., Vitalik Buterin's post here. Some protocols limit the "slippage," but this does not eliminate the arbitrage problem. My contribution is to show that the functional form of pricing that arises from a canonical economic model would not suffer from this deficiency. Angeris and Chitra (2021) formalizes the concept as "path deficiency."

Mempool front-running is not a hypothetical problem — bots scan and exploit profit opportunities in practice as Daian, Goldfeder, Kell, Li, Zhao, Bentov, Breidenbach, and Juels (2019) outline in great detail. In fact, some of the entities that are well-known for their high-frequency arbitrage activities in regular financial markets, such as Jump Trading, are also highly active in the DeFi space. Move broadly, CPAMM arbitrage profits are a form of so-called Miner Extractable Value (MEV) (see Section I.). The presence of such profit opportunities is associated with so-called high frequency priority gas auctions (PGAs) that have raised the costs of transactions on Ethereum dramatically over the last half year (coincidentally, since the introduction of decentralized swap exchanges in 2020) — a very clear case of a negative externality.

The second fundamental problem arises from the organization of liquidity. Consider a scenario with two identical swap exchanges. An investor should always take advantage of all available liquidity to reduce the cost of trading and split her order between the venues. However, suppose she does not. Then the price dislocation after a trade on one venue gives rise to a profitable trading opportunity, which comes at the expense of the liquidity providers. Namely, for any number x of tokens A bought on one venue, an arbitrageur can sell $0 < \tilde{x} < x$ on that venue, buy the amount on other venue, and earn a positive profit. Over-trading on one venue thus gives rise to "ping-pong" trading where arbitrageurs trade back and forth among the two venues. By point of contrast, this arbitrage opportunity does not arise with canonical pricing.

In the last part of the paper I discuss how mining (and trading) fees can somewhat mitigate both undesirable features of constant-product pricing because they reduce arbitrage profits, but the problem persists in particular for orders that are large enough and for contracts that are no sufficiently liquid. To establish some empirical stylized facts, I examine a small dataset from UniSwap. For very liquid contracts, the loss for investors from CPAMM pricing relative to canonical pricing is negligible. For less liquid trading pairs, however, the loss is substantial: for the trading of "wrapped" Bitcoin to USDT, in January 2021, more than 10% of investors stood to lose (theoretically) 53 bps or more of their trade size by using a swap exchanges such as UniSwap compared to the canonical pricing scheme.[4]

Decentralized finance is still a young field, despite the overall value that already circulates in the system. There are now several studies that explore the relationship of trading when there is both a CPAMM swap exchange and a centralized exchange. Parlour and Lehar (2021) describe the returns to liquidity provision and contrast CPAMM to limit order book trading. Aoyagi and Ito (2021) study the interaction of trading and prices when centralized and decentralized swap exchanges co-exist in a model of asymmetric information. Capponi and Jia (2021) focus on the complications that arise for liquidity providers of automated, constant product swap exchanges when the underlying token value, determined by prices on centralized venues, is volatile.

The contribution of my work is to highlight that the currently used CPAMM pricing approach is conceptually flawed because it creates intrinsic arbitrage and front-running opportunities. In practice, it is clear the current pricing approach creates negative externalities. However, the development in DeFi is fast-paced, projects are often short-lived, and the field is fluid. Since the field is, arguably, still at an experimental stage, it is imaginable (and desirable) that swap exchanges update their pricing approach in

[4] A side product of the calibration exercise is that I show how the various model parameters from the canonical Biais-model translate into the arguably more straightforward CPAMM formulation.

the future to remedy the current shortcomings.

I. The Trading of Blockchain Tokens

Prior to the summer of 2020, most crypto-tokens that had been issued on the various decentralized platforms could only be traded on so-called centralized venues such as Polionex, Binance, Kraken, Coinbase, or the now defunct QuadrigaX. These transactions were usually trades of crytocurrencies such as Bitcoin and Ether, the cryptocurrency of the Ethereum network, in exchange for either fiat currencies or other tokens that represented fiat currencies, so called "stablecoins" such as Tether. To trade, users have to register with the platform and transfer their blockchain asset to the "wallet" of an exchange, before they can use the exchange's system to make their trades. The implication is that as part of the process, custody of the asset moves from the user to the exchange. This arrangement is risky, as demonstrated by the numerous hacking and fraud incidents such as Mt. Gox, QuadrigaX, or Thodex. Moreover, it is almost comical that the blockchain community needed to rely on centralized exchanges for the trading of its decentralized tokens.

Blockchain based trading, however, has always been possible — after all, a limit order, the standard way to trade securities, is nothing but a simple contingent (smart) contract. And decentralized limit order books have indeed been around, e.g. EtherDelta.[5]

[5] The presence of these venues raises interesting legal questions. Formally, the contracts are a feature of the blockchain and they are available to interact with for anyone who has access to the internet. It is therefore not clear whether any jurisdiction has power of this functionality. Of course, the world of equity trading is highly regulated, and there are many rules of how a trading system can and cannot be operated. In practice, the websites that users access to trade on the decentralized exchange are operated by individuals or businesses. This allowed the S.E.C. to charge the founder of EtherDelta with operating an illegal national securities exchange. See: https://www.sec.gov/news/press-release/2018-258 Simply put: regulators have very little power over the operation of a blockchain, but they can

The idea of these decentralized "venues" is simple. Trades are always exchanges of two blockchain tokens, and so user simply have to register their limit orders as a smart contract on the blockchain, "locking" the token that can be sold subject to someone sending the desired amount of the other token to the contract. When that happens, the contract initiates an atomic swamp: the buyer receives the bought tokens and the seller receives the payment tokens. This atomic swap is processed by the blockchain miners and happens "in one go" so that there are never any failures to deliver — the trade is the settlement. A decentralized exchange (DEX) simply provides users with an interface to enter and sign orders using the cryptographic capabilities of their existing blockchain wallet. Moreover, the system keeps track of these orders (using information from the blockchain, not its own system) and displays available orders on a website.

Although such a mechanism is workable, it relies on individuals to continuously monitor and re-submit orders to supply liquidity; the required monitoring is a costly activity. Moreover, liquidity provision would also involve the perpetual submission of new liquidity providing orders, a process that absorbs computational power and thus mining fees.[6] Most importantly, this approach does not use much of the billions of value in aggregate capital that crypto-asset owners hold, very much in contrast to the legacy financial world where assets are used whenever possible.

This is where so called swap exchanges come in. A swap exchange is a smart contract that holds deposits of *pairs* of token from numerous liquidity providers.[7] Liquidity

exert power over people who are involved with blockchain projects.

[6] This is in contrast to most centralized exchanges or today's stock exchanges, where order submissions are usually free. The exception is Canada where users have to pay a nominal, sub-penny fee for every order that they submit; see Malinova, Park, and Riordan (2013). There are systems that keep the limit order book offline and only execute the trades on-chair.

[7] Balancer is more general and can hold portfolios of more than two tokens. If a particular pair is not available as a separate contract, some swap exchanges such as SushiSwap offer cross-contract

seekers then interact with this pool of liquidity and offer to add one type of tokens to this contract in exchange for the other part of the pair. The contract then determines the exchanged quantity based on a mechanical rule. Importantly, liquidity providers do not provide a priced contract, they can be entirely passive. These contracts involve just over 200 lines of code (in the meta-languages Reach it's about 20 lines).

The price of the swap, however, is only implicit in the sense that it may change at any time until the trade settles on the blockchain (i.e., gets processed by a miner who adds it to a block).[8] As I outlined in the introduction, this creates a problem because of the way that public blockchains are organized. The possibility of front-running is a fact of the organization of public blockchains because verified but non-settled transactions need to wait for inclusion in a new block in publicly visible "mem-pools." As Figure 1 illustrates, this allows a front-runner to send an identical transaction to the contract albeit with a higher mining fee. Such a transaction would receive a higher execution priority, thus enabling front-running.[9]

Miner Extractable Value. The problem of front-running in the CPAMM setup has been recognised as early as March 2018. Daian, Goldfeder, Kell, Li, Zhao, Bentov,

trading. For instance, suppose someone whats to trade USDC for USDT but there is no such contract. If, however, there is a contract for ETH and USDT and ETH and USDC, then the system would arranage a swap from USDC to USDT by trading USDC→ETH→USDT.

[8] Users can specify a maximum "slippage" or price impact that they'd accept.

[9] Within the DefI community, this type of front-running is often likened to the activities of so-called high-frequency traders (HFTs) on traditional stock exchanges. One of the most prominent activities of these HFTs is the "speed game" which refers to the practice of reducing the latency between different trading venues with ultra-fast data lines and having fast access to the central matching engine of the venues though co-location, i.e., physically placing their servers as closely as possible to the server of the matching engine. The purpose is to be able to react to market movements or new pieces of information as quickly as possible. Formally, however, in stock exchanges these HFTs cannot front-run a trade in the same way as in public blockchains because HFTs would not know about a trade of other traders until it appears on the tape — at which point it is, for all practical purposes, final. Although often brought up, this analogy is, therefore, inaccurate. That being said, known HFTs from equity and futures markets are active in DeFi markets; see Jump Trading's podcast.

Breidenbach, and Juels (2019)[10] describe the high-frequency mining fee auctions that often occur when front-running bot identify profitable transactions, including (but not only) CPAMM trades. UniSwap alone, however, usually accounts for a very large fraction of the gas (aka computational cycles) of the Ethereum blockchain.[11] For instance, the blog post Ethereum is a Dark Forest[12] describes a case where the authors identified a smart contract vulnerability. They tried to remedy the problem by moving funds out of the contract, but a bot identified their transaction, outbid them, and extracted the value. The overall background is that automated arbitrage bots scan the mempool for profitable, not-yet-settled transactions. If a bot spots such a transaction —including the arbitrage opportunities that I describe in this piece— it submits an identical transaction with higher fees to outbid the original one. If other bots do the same, a so-called priority gas auction ensues. Miners can thus extract revenue from profitable transactions (*Miner Extractable Value*, MEV) either by earning higher fees or by submitting the transaction themselves. A Paradigm Research blog post by Noyes (2021) describes this issue in more detail and estimates that in December 2020, realized MEV amounted to about $120M per day; the author states that UniSwap arbitrage is the most common form, though the data source and data generating process is not clear from the post.

There are a number of projects that try to remedy this issue; Aune, O'Hara, and Slama (2017), for instance, propose a technique to guarantee time priority in private blockchains.

[10] See also the Flashbot Project.

[11] Running data is available at ETH Gas Station; UniSwap's V2's main contract address is 0x7a250d5630b4cf539739df2c5dacb4c659f2488d.

[12] "Dark Forest" refers to an environment, described in a SciFi book by Cixin Liu of the same title, in which detection means certain death at the hands of advanced predators.

II. Market Making Models and Pricing

A. Overview of Standard Market Making Models

The main model in the finance literature on market microstructure is the Kyle (1985) model in which a single insider has perfect knowledge of the fundamental value of an asset. This insider (together with some "noise" traders) trades with a market maker an must take into account how his trades will affect the price. The Kyle model explicitly assumes that the market maker is subject to perfect competition and in the model, this results in linear price and demand functions. The most common approach to pricing in this type of model is to assume a competitive market for liquidity provision such that prices coincide with the fundamental value of the underlying security, subject to the information revealed should a trade take place at the given price. This model, however, assumes that there are information asymmetries (i.e., that there are insiders who genuinely know the fundamental value of an asset).

There are also models of limit order pricing where traders decide between submitting market and limit orders. The driving forces in these models, e.g., Parlour (1998), is the execution risk of limit orders, or the risk that the fundamental value changes so that one's limit order becomes stale, e.g. Foucault (1999) or, more recently, Budish, Cramton, and Shim (2013).

In this paper, I want to avoid the complications associated with asymmetric information. Limit order models are not of interest for this paper either, because swap exchanges are not limit order markets.

The most applicable class of models to compare to swap exchanges, in my opinion, are those that assume that market maker prices are driven by inventory risk: when accepting a trade, liquidity providers need to absorb a risky inventory, and prices are set to compensate liquidity providers for their risk. Prominent models in this literature originate from Huang and Stoll (1997), and I use a variation of Biais (1993), which is a very tractable version with prices that are linear in quantities.

B. Market Making with Inventory Risk

The model that I develop in this section is in the tradition of Biais (1993): an investor wants to trade quantity x of token A and receive (or make a payment in) token B. For simplicity, I assume that token A is a risky security and security B is riskless (i.e., it a nummeraire such as cash).[13]

We assume here that the risky token A has a fundamental value that is normally distributed with mean V and variance σ^2. Information regarding the distribution of the fundamental value is public knowledge. The asset is infinitely divisible.

All trades go through a group of N intermediaries, who require compensation in exchange for taking a risky inventory in token A. Namely, we assume that the intermediaries are risk averse and have negative exponential utility of wealth w (which displays constant absolute risk aversion (CARA)) with risk aversion coefficient $\kappa > 0$, and inventory of A tokens I_i, with $\sum_i I_i =: X$. With CARA utility intermediaries have utility of terminal wealth $u(w) = -e^{-\kappa w}$, where $w = -(v - p)x_i + I_i \cdot v$ and v denotes the value of an A token measured in B tokens, x_i is the quantity of A tokens that they sold at price p.

[13] If both tokens would be risky, e.g., users would trade Bitcoin against ETH, then we would measure each against a riskless asset.

In its simplest version, all tokens trade at a single price. The underlying assumption is that a trader would arrive at the market once and makes a single trade.[14] Some of the ideas that I will discuss below, however, have an implicitly dynamic flavor in the sense that we worry about subsequent trades. I will assume here instead that market makers post a pricing function $p(\cdot)$ and that when purchasing quantity x, investors pay $\int_0^x p(t)dt$. As I will show below, a linear pricing function is an equilibrium, and therefore the total cost function is quadratic. With a uniform price and a linear price function, investors would have an incentive to split their trades into infinitesimal orders sizes and trade "along" the price curve; this would allow them to reduce the total cost by half compared to trading the whole quantity in one go.[15]

For simplicity we assume that the supply of the riskless token B is ubiquitous, i.e., that intermediaries can borrow these tokens at an interest rate of 0. Intermediaries observe orders submitted to them, respond with a supply schedule that maximizes their utility, and then markets clear. Namely, they each specify for each price p of an A token measured in the B token, how many tokens they are willing to buy (or sell) $\forall p \in \mathbb{R} \;\; x_i(\cdot) : p \rightarrow x_i(p)$. Markets clear at a uniform price such that $\sum_{i=1}^{N} x_i(p) = x$.

Lemma 1 (Equilibrium price in Standard Market Making): *To buy x units of token A from the N intermediaries investors pay a marginal price*

$$p(x, I) = V + \frac{2\kappa\sigma^2}{N}(x - I). \tag{1}$$

[14] One way to think about this is that there is an implicit assumption that traders are not anonymous so that market makers can detect repeated trading accept trades only at prices that reflect the full size of an order.

[15] See van Kervel, Kwan, and Westerholm (2020) for a model where order splitting across time is intrinsically beneficial, even in the absence of asymmetric information.

We then define parameter $\ell := \kappa\sigma^2$; it signifies the liquidity or the price impact cost in this intermediated market. This pricing formula is, in principle, entirely mechanical and can be programmed as a smart contract (subject to sufficient availability of B tokens in the contract).

Proof. Intermediaries choose quantities x_i given price p, to maximize their expected utility, $\max_{x_i} \mathsf{E}U[-vx_i(p) + \int_0^{x_i} p(t)dt + I_i \times v]$. For CARA-normal frameworks, this task reduces to maximizing the certainty equivalent:

$$\max_{x_i}[I_i \times V - (Vx_i - \int_0^{x_i} p(t)dt] - \frac{\kappa}{2}\sigma^2[-x_i + I_i]^2,$$

where V denotes the expected value of the asset. The maximization problem results in the following first order condition:

$$V - p(x_i) - \kappa\sigma^2 \times I_i + \kappa\sigma^2 x_i = 0.$$

Solving for x_i yields the (inverse) demand schedule

$$x_i(p) = -\frac{V - p}{\kappa\sigma^2} + I_i.$$

The market clearing condition

$$\sum_{i=1}^{N} x_i(p) = x$$

implies, substituting for x_i, and simplifying, that

$$\sum_{i=1}^{N} \left(-\frac{V-p}{\kappa\sigma^2} + I_i \right) = x$$

which we solve for p to be

$$\Leftrightarrow \quad p(I,x) = V + \frac{\kappa\sigma^2}{N} \; (-I + x), \tag{2}$$

where I denotes the combined inventory of the intermediaries: $I = \sum_{i=1}^{N} I_i$. □

Price changes in this model occur for two reasons: changes in the fundamental value V, and trades which require intermediaries to absorb a risky inventory. When an investor approaches the intermediaries who hold total inventory I in order to buy x units, and the investor pays, in abuse of notation, the total cost for buying x is

$$p^{\mathsf{tmm}}(x,I) := \int_0^x V + \frac{\ell}{N}(t-I) \; dt = x \left(V - \frac{\ell}{N} I \right) + \frac{\ell}{2N} \; x^2,$$

where I use superscript tmm to signify a “tradtional” market maker pricing approach. Although the marginal price is linear, costs are convex functions of the quantity.

C. Constant Product Automated Market Makers

In a constant product market maker pricing model (CPAMM), liquidity providers make deposits for a trading pair of tokens A and B in a smart contract. Specifically, they provide aggregate quantities X units of tokens A and Y units of tokens B. Pricing is such for any demand $x \in (-\infty, X]$, value y is such that $(X-x) \times (Y+y) = c$ for some constant c (and the reverse for $y \in [-Y, \infty)$), and in particular $XY = c$. Therefore, one can think of this exchange rate as the fundamental value of an A token measured in B tokens. The direct implication is that the marginal price of an A token measured in B tokens is Y/X, and the price $p^{\mathsf{cmm}}(x)$ per unit when purchasing quantity x is y/x, where I use superscript cmm to signify the CPAMM pricing schedule.

Lemma 2 (CPAMM Pricing): *To trade x units of token A, an investor pays*

$$y(x) = x \times \frac{Y}{X - x} := p^{\mathsf{cmm}}(x, X, Y). \tag{3}$$

The proof is a merely algebraic rearrangement of the equation $(X - x) \times (Y + y) = c$ and thus omitted. After the trade, the contract contains $X - x$ units of token A, and $XY/(X - x)$ units of B.

III. Properties of the Pricing Functions

A. A Brief Comparison

The two pricing functions are obviously different: one is linear in token A demand x, the other is convex (as the second derivative of the price function is positive). The cost function for standard market making, however, is convex, too. For standard market maker pricing, the liquidity factor ℓ and the number of market makers N determine the slope of the linear function, the inventory I shifts the intercept, as does the true value V. However, an inventory in the standard market maker model is always considered to be costly whereas presumably some users agree to provide liquidity to use their buy-and-hold assets while earning additional yield through lending out their securities (akin to mutual funds). In that sense, any inventory I should be seen as *excess* amount of inventory beyond a target amount. To compare the two pricing functions, I set set the target amount to X and thus $I = 0$ and assume that market makers deposit single units of A tokens so that $N = X$.

Next, the constant product market maker model is not concerned with a "true" value for the asset. To compare CPAMM and standard market making, bar any trading, the implied marginal price for A tokens must coincide with the true value, i.e. $V = Y/X$. Then $Y = VX$ is the total cash amount that market makers contribute and the cash amount is equivalent to the cash value of their inventory of A tokens.

The two price functions coincide for two values of x. One is $x = 0$, by construction. The other is

$$x^* = \frac{X}{\ell}(\ell - 2V).$$

Figures 3 and 4 plots the two cost functions p^{cmm} and p^{tmm} for $V = 1$ and for various values of X and ℓ as indicated in the respective plot legends.

Figure 3 provides a plot for the entire range of feasible values, Panel A in Figure 4 focuses on the subset of plot for which $x^* < 0$, and Panel B in Figure 4 focuses on the subset of plot for which $x^* > 0$. Dotted lines are for p^{cmm}, dashed lines are for p^{tmm}. Negative values of x signify a sale and the cost, therefore, indicates revenue earned. The nature of the quadratic function cost function p^{tmm} is such that for low enough values of x, the investor still has to pay because the market makers require compensation for the inventory risk that they take. For the CPAMM case, on the other hand, in the limit for $x \to -\infty$, $p^{\mathsf{cmm}} \to -VX$ or $p^{\mathsf{cmm}} \to -Y$, meaning that the investor can at most extract all the cash that has been deposited by the liquidity providers. Likewise, as $x \to X$, the cost rises without bound.

The two cases highlight the different underlying ideas: for standard market making, the cost of a buy increases quadratically, but there is no explicit limit and the assumption is that market makers can provide or absorb any quantity of the desired token. Likewise, they are willing to buy arbitrarily large quantities, but at some point, they are no longer paying the seller, they require to be paid to accept further assets. The CPAMM model, on the other hand puts no bounds on the quantities that can be bought or sold, which is useful for smart contract automation.

The plots further highlights that the two cost functions intersect at x^* and merely touch at 0. Overall, the following result summarize the relationship of the two pricing functions.

Proposition 1 (Comparison of Prices): *For $x > x^*$, constant product pricing is worse for investors, and for $x < x^*$ it is better.*

Proof. With two points of intersection, there are two scenarios: $x^* < 0$ and $x^* > 0$.

- When $x^* < 0$, then $\forall x > x^*$, $p^{\mathsf{tmm}}(x) < p^{\mathsf{cmm}}(x)$, i.e., investors pay less in the standard model for $x > 0$ and they receive more for $x < 0$. Furthermore, $x < x^*$, $p^{\mathsf{tmm}}(x) > p^{\mathsf{cmm}}(x)$ such that investors receive less in the standard model.

- When $x^* > 0$, then for all values $x \in (0, x^*)$, $p^{\mathsf{tmm}}(x) > p^{\mathsf{cmm}}(x)$, which means that CPAMM provides a “better” price for investors because investors receive a higher price for their sales. For $x > x^*$, $p^{\mathsf{tmm}}(x) < p^{\mathsf{cmm}}(x)$ and standard pricing is better for investors. Finally, for $x < 0$, $p^{\mathsf{tmm}}(x) > p^{\mathsf{cmm}}(x)$ which means that investors receive less for what they sell in the standard model.

□

This finding shows that from a pricing perspective it is not straightforward to order the two pricing models, except for the knife edge case when $X\ell = 2V$. Generally, the results are intuitive however: the more liquid the standard market is (ℓ is small), the larger is the region of prices for which standard pricing is better. Likewise, the more liquidity investors are willing to provide for the CPAMM model (X is large), the larger is the region of prices for which CPAMM prices are superior for investors. The values X and ℓ, however, have an intuitive, non-direct connection. Namely, we assume that the starting inventory of market makers is $I = 0$ and that the market makers assess inventory other than their "comfort zone" to be costly. When X is large, market makers are willing to hold the A token, which presumably should also imply that ℓ is small. In other words, X and ℓ should be negatively related in practice.

B. *Desirable Properties of Pricing Functions*

A first important property is *additivity* which implies that splitting an order into two consecutively traded parts costs the same as sending an order of the same size "in one go." If this condition does not hold for a theoretical model, then the formulation is potentially dynamically inconsistent. Moreover, in a blockchain world, this feature would create excessive network usage without economic gain. In limit order markets, for instance, additivity often does not hold automatically if the market adjusts quotes after trades. In practice, traders commonly break large "parent" orders into many small "child" orders to reduce costs, but such behavior is often driven by considerations other than the pricing function.

Second, I ask whether *front-running is intrinsically profitable.* Namely, front-running refers to a situation where a trader Alice sees the trading intention of another trader Bob, mimics Bob's trade but acts before him, and then does an offsetting trade to close her original position. Front-running is usually costly for the person who is front-run, but it is not clear whether the possibility of front-running will always lead to a positive probability of front-running. For that, it would have to be intrinsically profitable. If that's the case, then the underlying pricing function is highly undesirable for a setting with a public blockchain. The reason is that in public blockchains it is impossible to prevent front-running because orders are always visible in the mempools prior to settlement.[16] We consider only cases where the original amount x is less than $X/2$, so that front-running is possible (for the total quantity bought would be $2x$).

The next set of properties applies to situations when there are multiple trading venues with the same pricing model. The first property relates to the splitting of liquidity (e.g., because a new venue opens for business). The first question to ask is whether *liquidity is divisible*, i.e., are trading costs the same when liquidity is concentrated on an existing system compared to it being split across two different ones? Modern markets are often fragmented and it is important to understand whether the pricing function itself generates an explicit benefit or cost when liquidity is fragmented (beyond the obvious fact that splitting trades involves twice the miner fees).

The second question relates to (accidental) single-venue over-trading. Namely, when liquidity is split across multiple venues, a trader should use all venues for a trade.[17] In practice, in the DeFi world there are order routing services such as 1inch and Paraswap

[16] This area is currently an active research field, the hope being, that cryptographic tools can be used to mask trades in the mempool.

[17] Generally speaking if liquidity is divisible, then traders should split their trades between venues in proportion to their liquidity. For simplicity I look at the case where the liquidity is the same on two venues, so the trader should have traded half her volume on each venue.

that perform this service and break up large orders across protocols.[18] Suppose, however, that a trader doesn't do this and instead uses only one of two identical venues (for instance, because s/he is unsophisticated or impatient). This will move the price for subsequent trades on one venue due while leaving the other unchanged. Multi-venue arbitrage occurs when the selling of an amount larger than half of the last trade on the first venue and buying it back on the other venue is intrinsically profitable.[19]

C. Properties of Standard Market Making Prices

Without loss of generality, in the proofs for standard pricing, we set $V = 0$ and $I = 0$. All results are based on buys; a symmetric argument holds for sells. I use $p^{\mathsf{tmm}}(x, 0)|_{N \to N'}$ to signify the pricing when the N market makers are reduced to $N' \leq N$ market makers (e.g., because N is split between two parallel systems).

Proposition 2 (Properties of Standard MM Pricing):
For $x \in (0, X)$, $\alpha \in (0, 1)$ and $k > 0$:

1. *Non-Profitable Order Splitting: for $\alpha \in (0,1)$: $p^{\mathsf{tmm}}(\alpha x, I) + p^{\mathsf{tmm}}((1-\alpha)x, I + \alpha x) = p^{\mathsf{tmm}}(x, I)$.*

2. *Unprofitable Front-running: $-p^{\mathsf{tmm}}(x, I) + p^{\mathsf{tmm}}(-x, I + 2x) = 0$.*

3. *Additivity with split liquidity, $k \cdot p^{\mathsf{tmm}}(x/k, I)|_{N \to N/k} = p^{\mathsf{tmm}}(x, I)|_{N \to N}$.*

4. No *Multi-venue arbitrage: $\forall \alpha > 1/2$, $p^{\mathsf{tmm}}(-\alpha x, I + x) - p^{\mathsf{tmm}}(\alpha x, I) < 0$.*

[18] I thank Julien Prat for pointing out these services.

[19] Why more than half the original order? Effectively, the original trader over-traded with one venue. Trading half the original amount as described would align liquidity on both venues to be as it should have been had the original trader split the trade equally among the two venues.

In the standard model, market makers intuitively price "along" the marginal price curve. Splitting an order x into quantities αx and $(1-\alpha)x$, therefore, simply means that after trading αx, the market maker accepts quantity $(1-\alpha)x$ at an inventory of αx, or, in plain English, she continues to price where she left.[20]

Proof. of 1.: After selling αx to an investor, the inventory changes by $-\alpha x$. Then,

$$\begin{aligned} p^{\mathsf{tmm}}(\alpha x, I) + p^{\mathsf{tmm}}((1-\alpha)x, I+\alpha x) &= \int\limits_0^{\alpha x} (V + \ell(t-I))dt + \int\limits_0^{(1-\alpha)x} (V + \ell(t - I + \alpha x))dt \\ &= \int_0^x \times (V + \ell(t-I))dt. \end{aligned}$$

Proof of 2.: As a front runner, the investor buys x, sees the other party buy x from the market makers, and then sells x. Payoffs are the amount received from selling minus the amount paid for buying. As before, without loss of generality, we set the initial inventory to $I = 0$. The front runner pays $p^{\mathsf{tmm}}(x, 0)$ for the initial position and receives $p^{\mathsf{tmm}}(-x, 2x)$ when liquidation (the inventory has increased by $2x$ from the front-runner's as well as from the front-run's trade). Then

$$\begin{aligned} -p^{\mathsf{tmm}}(x, 0) + p^{\mathsf{tmm}}(-x, 2x) &= -\int_0^x \ell t dt + \int_0^x \ell(-t + 2t)dt \\ &= \int_0^x \ell(-t(-t+2t)dt = 0. \end{aligned}$$

Proof of 3.: Additivity with split liquidity follows directly from the market setup because the model assumes a single market clearing price, and it is irrelevant where

[20] The standard Biais (1993) model has a single, uniform price, because the model makes the implicit assumption that trading is not anonymous and that, therefore, market makers would detect such behavior and refuse future trades.

market makers are. Formally, note first, that when the N market makers are split into k equal groups then $\ell = \kappa\sigma^2/N \to \kappa\sigma^2/(N/k) = k \cdot \ell$. Therefore

$$\begin{aligned} & k \cdot p^{\mathsf{tmm}}(x/k, I)|_{N \to N/k} = k \cdot \frac{\ell k}{2}(x/k)^2 = \frac{\ell}{2}x^2 \\ = \quad & p^{\mathsf{tmm}}(x, I)|_{N \to N} . \end{aligned}$$

Proof of 4.: The result follows from straightforward algebra:

$$\begin{aligned} & p^{\mathsf{tmm}}(-\alpha x, x) - p^{\mathsf{tmm}}(\alpha x, 0) \\ = \quad & \alpha x \frac{\ell}{2}(-\alpha x + x) - \alpha x \frac{\ell}{2} \alpha x = \frac{\ell}{2}\, \alpha(1 - 2\alpha)x^2. \end{aligned}$$

The last expression is negative if $\alpha > 1/2$. □

D. *Properties of Constant Product Automated Market Making Prices*

As a next step, we examine the same relationships for constant product market maker pricing. Recall that $y(2x, \cdot, \cdot)$ signifies the amount of token B that one has to pay for $2x$ many of the A tokens.

Proposition 3 (Properties of CPAMM Pricing):
For $x \in (0, X)$, $\alpha \in (0, 1)$ and $k > 0$:

1. *Additivity:* $p^{\mathsf{cmm}}(\alpha x, X, Y) + p^{\mathsf{cmm}}((1-\alpha)x, X - \alpha x, Y + y(\alpha x, X, Y)) = p^{\mathsf{cmm}}(x, X, Y)$.

2. *Profitable Front-running:* $-p^{\mathsf{cmm}}(x, X, Y) + p^{\mathsf{cmm}}(-x, X - 2x, Y + y(2x)) > 0$.

3. *Additivity with split liquidity,* $p^{\mathsf{cmm}}(x, X, Y) = k \cdot p^{\mathsf{cmm}}(x/k, X/k, Y/k)$.

4. *Multi-venue arbitrage:* $p^{\mathsf{cmm}}(-\alpha x, X-x, Y-y(x)) - p^{\mathsf{cmm}}(\alpha x, X, Y) > 0\ \forall \alpha \in (0,1)$.

Proof. of 1.: Since $y(x) = xY/(X-x)$, we have that after purchase of x, there are $XY/(X-x)$ of the type B tokens. Therefore

$$p^{\mathsf{cmm}}((1-\alpha)x, X-\alpha x, Y+y(\alpha x)) = \frac{(1-\alpha)xXY}{(X-\alpha x)(X-x)}.$$

Together we get

$$\begin{aligned}
&p^{\mathsf{cmm}}(\alpha x, X, Y) + p^{\mathsf{cmm}}((1-\alpha)x, X-\alpha x, Y+y(\alpha x)) \\
=\ & \frac{\alpha x Y}{X-\alpha x} + \frac{(1-\alpha)xXY}{(X-\alpha x)(X-x)} \\
=\ & \frac{xY(\alpha(X-x)+(1-\alpha)X)}{(X-\alpha x)(X-x)} = \frac{xY}{X-x} \\
=\ & p^{\mathsf{cmm}}(x, X, Y).
\end{aligned}$$

Proof of 2.: The front runner pays $y(x, X, Y)$ and later sells x where there have been two purchases of x each before (one by the front-runner, and one by the person having been front-run). For selling x, the front runner then receives

$$y(-x, X-2x, Y+y(2x)) = \frac{xXY}{(X-x)(X-2x)}.$$

Then we simplify to get

$$y(-x, X-2x, Y+y(2x)) - y(x, X, Y) = \frac{2x^2Y}{(X-x)(X-2x)} > 0. \tag{4}$$

Proof of 3.: We have

$$ky(x/k, X/k, Y/k) = \frac{kx/kY/k}{X/k - x/k} = \frac{xY}{X - x} = y(x, X, Y).$$

Proof of 4.: After a trade of x on venue 1, an arbitrageur sells $\alpha x < x$ on venue 1 and buys αx on venue 2. The cost for this trade is

$$\begin{aligned} & y(-\alpha x, X - x, Y - y(x)) - y(\alpha x) \\ = \; & \frac{\alpha x Y}{(X - \alpha x)(X - (1 - \alpha)x)(X - x)} \\ & \qquad \times \big(2X(X - x) + x^2(1 - \alpha)\big) > 0. \end{aligned}$$

□

The possibility of profitable front-running as well as multi-venue arbitrage (and subsequent ping-pong trading) is therefore an intrinsic concern under the given pricing model. This problem of front-running is indeed well-known, which is why all swap trading venues include an option for traders to limit the "slippage", i.e., when asking to arrange a trade, traders can limit the price impact/change that their order will face. As Proposition 2(2) shows, however, a different pricing system can eliminate this problem altogether.

IV. Impact of Trading Fees for CPAMM

A. Mining and Trading Fees

Trading Fees are best thought of as compensation for liquidity providers. The slope of the pricing curve in the canonical pricing model is an exact representation of the compensation that liquidity providers obtain for taking on risk under market clearing. No other compensation is warranted in a competitive market.

For CPAMM pricing it is not clear whether the shape of the curve reflects "fair" compensation for taking on risk. In fact, as I showed in the last section, assuming that the liquidity providers are the same under both regimes, CPAMM pricing either over or under-compensates liquidity providers for taking risk. In addition to the pricing function, in practice CPAMM users also pay a fee to liquidity providers and so far I have abstracted from this fee to simplify the exposition. However, the presence of this fee makes trading more costly and this implies that there is an additional cost to frontrunning and arbitrage trades. This fee therefore reduces the set of trades to which Proposition 3 (2) and (4) apply, which means that no all trades create an arbitrage opportunity, only sufficiently large ones do.

The same insight applies to **mining fees**: uses need to pay miners for each transaction so that it (or, rather the underlying code) gets executed on the blockchain. Mining fees therefore also reduce the set of trades for which front-running and arbitrage is profitable. The difference between mining and trading fees is that the latter are somewhat under the control of the trader herself. In the next subsection, I argue how a trader can use fees "defensively" to prevent front-running (the formulation is conceptually agnostic as to whether these fees are mining or trading fees).

For this defense, however, there is a bigger question relating to the workings of mining. Namely, the idea of a defensive fee is predicated on the idea that someone would have to pay a fee to outbid the original trader and then pay a similar fee to reverse the transaction. The implicit assumption is that this "someone" must pay the miner. But what if the front-runner is the miner itself? Then that the defensive fees are irrelevant because the miner can choose the order of transactions in the block and put its own transactions ahead of the front-run trade without paying a fee. In other words, mining fees cannot work as a defense against malicious miners.

Now, it is important to emphasize that even though the word suggests otherwise, miners are not people! Mining follows an open-source algorithms,[21] there is no secret in what miners are doing, their actions are transparent and hard-coded, and they don't take decisions pondering each arbitrage case. If they are malicious, it would be plainly visible (including the fact that their blockchain addresses are known). In principle, it may be possible for a protocol to avoid broadcasting transactions to mempools that are operated by malicious miners, though this is likely technologically tricky.

As I argue in this paper, there is an altogether better approach by using a pricing function that doesn't enable exploitative behavior in the first place.

B. Defensive Fees

Equation (4) specifies the front-running profits. These profits do not account for the mining fees or other transaction fees. Clearly, to be successful, the front-runner

[21] See, for instance, this link for the part of the Parity mining protocol that describes the ordering of transactions by fees.

has to outbid the original trader for the mining fees and she has to pay mining fees for the return transaction, too. The mining fees for the return transaction therefore have to be high enough to ensure that the return trade gets included in the same block (otherwise, the front-runner faces a possibly significant execution risk, because of 4.) but not higher to ensure that the front-run trade gets accounted for first. By choose a sufficiently high fee, the original traders can therefore make front-running unprofitable.

Proposition 4: *For each quantity x there exists a mining fee f that the original trader can submit to render front-running unprofitable; the fee is increasing in x.*

Proof. If the original trader pays f, the front-runner will pay $f + \epsilon_1$ for the first leg and $f - \epsilon_2$ for the return trip. For front-running to be profitable, equation (4) with mining fees has to satisfy

$$\frac{2x^2Y}{(X-x)(X-2x)} > 2f + (\epsilon_1 - \epsilon_2). \tag{5}$$

For ϵ_1, ϵ_2 arbitrarily close to zero, there exists an $\bar{f}$ such that for any mining fee $f > \bar{f}$, front running is unprofitable. Furthermore, the left hand side of (5) is obviously an increasing function of x. □

In other words, although front running is intrinsically profitable, when taking fees into account, traders do not have to accept arbitrarily large prices because of the possibility of front running. The left hand side of (5), however, is a convex function of x and therefore for large quantities, "protective" fees increase more than proportionally.

A second source of fees are the explicit trading fees that the liquidity providers receive. There is a subtle difference to transaction/mining fees in that these fees are usually proportional to the amount traded (in UniSwap V2, they were 30bps of the transaction amount; the details differ between protocols). However, the principle is the same: these fees can prevent front-running as they make each leg of the front-running trade more expensive.

Fees have no impact on the "desirable" features for canonical pricing: there is still no multi-venue arbitrage, and there is still no profitable front-running. The fees that users would have to pay to avoid front running are, in fact, a clear example of Miner Extractable Value (MEV) because the miners themselves could perform the front-running. In economic terms, this type of MEV amounts to rent extraction which leads to socially suboptimal outcomes.

V. A Simple Calibration Exercise

The analysis thus far has been abstract and it is unclear whether the problem that I describe is empirically relevant. In this section, I establish a few stylized facts to link the model to market data.

First, the standard market maker model requires a choice for the risk aversion coefficient κ. Using a meta-analysis, Babcock, Choi, and Eli Feinerman (1993) (Table 1) report that the CARA coefficient κ falls anywhere between .00001 and 0.5, where most of the values in the literature are at the small end of the spectrum; in my subsequent calibration, I will use $\kappa = 0.0005$.

Second, I will compare calibrated trading costs for three common trading pairs: Bitcoin–USD, Ether-USD, and USDT-USDC, where the latter is the exchange rate of two common stablecoins, USDT, issued by Tether Ltd., and USDC, issued by Coinbase

Inc. Here, USD and USDC are the nummeraire tokens Y.

Table I lists the estimates that I use for the calibration. For the fundamental values of these tokens V I will use the January 17, 2020 approximate prices of \$36,000, \$1,200, and \$1 for BTC, ETH, and USDT. To compute their standard deviations I collected data from CoinDesk (for BTC and ETH) as well as Nomics for USDT-USDC for the time from January 16, 2020 to January 16, 2021.

To determine the values X, the capital at risk, I use information on the liquidity provision in UniSwap trading pairs, currently the most heavily used CPAMM protocol, where I used either Tether's USDT or Coinbase's USDC as the baseline token Y, and I use so-called wrapped Bitcoin wBTC as a proxy for Bitcoin. On January 17, 2021, the liquidity provision contract for the USDT-ETH pair contained roughly 80K ETH, the USDC-wBTC contained 120 wBTC, and the USDC-USDT contract contained 17M USDT (and a similar amount of USDC). Therefore, the three contracts reflect medium, high, and low liquidity as well as low, medium and high volatility.

For all these parameters, the value x^* for which canonical and CPAMM costs coincide is negative (and quite large in absolute value). Therefore, for a very large set of reasonable trade sizes, canonical pricing offers "better" terms to investors. Using these figures, I then compute the excess trading costs as well as the mining fee in (5) that investors have to submit to avoid being front-run. I compute these items for three standard-sized trades of \$1,000, \$5,000, and \$10,000.

Table I summarizes the amounts. For small trade sizes (\$1,000) the excess cost of CPAMM are, arguably, negligible. The same holds generally for the highly liquid contract ETH-USD. However, for the less liquid contracts, costs can be substantial:

Table I
Calibration of Costs and No-Arbitrage Fees

	trade size	USDC-USDT	ETH-USD	BTC-USD
value V		1	1.2K	36K
σ		0.01	220	6.6K
X		17M	80K	120
X in USD		17M	100M	4.3M
x^*		$-3.4e^{15}$	-39.6M	-1.9K
excess cost	1K	0.1	0.0	0.3
CPAMM	5K	1.5	0.4	6.4
	10K	5.9	1.5	25.4
arbitrage	1K	0.1	0.0	0.3
prevention	5K	1.5	0.4	6.8
fee	10K	5.9	1.5	27.2
total in bps	1K	1.2	0.3	5.2
of transaction	5K	5.9	1.5	26.3
value	10K	11.8	3.0	52.6

trading $10,000 or more of wBTC-USD comes at a combined excess cost of 53 bps for the trade — which is a large cost in the trading world. To put this into perspective, for this type of highly traded pair, this is a massive cost: at a price of $30K, even a $1 bid ask spread is less than a basis point, and on most centralized markets, spreads are smaller than $0.10.

Using the first 4,000 transactions on January 14 and thereafter from the UniSwap contract Etherscan for the relevant wBTC-USDC pair, about 14% of transactions exceed $10K in value, and another 13% have transaction value between $5K and $10K.

Although it is not clear whether these users have suffered from front-running, the possibility implies that they would have been better served with a standard market maker pricing protocol.

VI. Discussion and Conclusion

Ad-hoc, exogenous pricing functions are unsatisfactory for economists, because such prices are not market prices, even if they are driven by demand and supply. With such pricing, it is unclear if prices will ever reflect demand and supply or aggregate information, all of which are core functions of market prices.[22] In standard trading models from the financial market micro-structure literature, liquidity providers compete to provide the best price. This assumption ties down how prices are determined, and in this paper I present a variation of the standard Biais (1993) model, that demonstrates that a pricing model that's derived from primitives can display many desirable properties of pricing functions. In contrast, the ubiquitous constant product automated market making function that most decentralized swap exchanges use is an economically arbitrary pricing function. I show here how it has theoretically highly undesirable properties that matter in practice, at least for lower liquidity tokens (arguably, the part of the market for which the pooling of liquidity has the most promise).

[22] Notably, Aoyagi (2020) develops a theoretical model of investor and liquidity provider behavior around constant product pricing as an exogenous rule. In his model there are three main market participants: noise traders, who trade for personal reasons; informed traders, who know the true value of the underlying asset and who maximize their trading profits, taking account of the effect of their trade on prices (as in Kyle (1985)); and liquidity providers who maximize their profits taking into account the pricing rule and the behavior of the latter two types of traders. Aoyagi (2020) solves the model in terms of traders and liquidity providers' behavior and he shows that the scheme allows liquidity providers to extract rents. Moreover, although users can infer the security's fundamental value from prices, the price usually not coincide with it.

As for my results. There are two main differences between the market maker models that I discuss in this paper. First, multi-venue arbitrage in the standard model is limited to a smaller set of quantities. Normally, the second market that saw no trade should adjust after the other market moved (and the initial trade should have occurred on both venues). Reverting half of the original amount should generally be profitable because the original trader would have essentially made a mistake. However, for the CPAMM model, there is a much wider range of intrinsically profitable trading opportunities.

Second and more importantly, front running in the CPAMM model is always profitable whereas it is not in the standard market making model. This is a fundamental and intrinsic problem in the CPAMM market maker model. Fees can mitigate to problem, but each of these fee solutions make trading as a whole more expensive and generate redistribution of income away from traders to miners. As Proposition 1 shows, when there are sufficiently many liquidity providers that are not too risk averse, then linear pricing is much cheaper for investors or, put differently, liquidity providers simply receive too much, above-market compensation for the risks that they take.

These properties aside, there are other concerns that merit attention. For instance, prices on a swap exchange do not exist in isolation because blockchain securities can be traded on various venues. This implies that the price of the CPAMM is often wrong relative to the broader market, unless liquidity suppliers constantly update their quantities such that the marginal price is in line with what's available elsewhere. Capponi and Jia (2021) studies this issue extensively and they show how volatility prices can lead to surges in transactions fees for the entire blockchain –yet another negative externality– as well as to liquidity freezes. All in all, although swap exchanges have very desirable features, such as the pooling of liquidity, there are several major concerns that need to be addressed for these systems to unfold their potential.

REFERENCES

Adams, Hayden, Noah Zinsmeister, and Dan Robinson, 2020, Uniswap v2 core, Whitepaper Uniswap Foundation https://uniswap.org/whitepaper.pdf.

Angeris, Guillermo, and Tarun Chitra, 2021, Improved price oracles: Constant function market makers, Working paper Stanford University https://arxiv.org/abs/2003.10001.

Aoyagi, Jun, 2020, Liquidity provision by automated market makers, Discussion Paper, working paper U.C. Berkeley https://ssrn.com/abstract=3674178.

———, and Yuki Ito, 2021, Liquidity implications of constant product market makers, Working paper UC Berkeley https://ssrn.com/abstract=3808755.

Aune, Rune, Maureen O'Hara, and Ouziel Slama, 2017, Footprints on the blockchain: Information leakage in distributed ledgers, Working paper Cornell University https://ssrn.com/abstract=2896803.

Babcock, Bruce, E. Kwan Choi, and Eli Feinerman, 1993, Risk and probability premiums for cara utility functions, *Journal of Agricultural and Resource Economics* 18, 17–24.

Biais, Bruno, 1993, Price formation and equilibrium liquidity in fragmented and centralized markets, *The Journal of Finance* 48, 157–185.

Budish, Eric B, Peter Cramton, and John J Shim, 2013, The high-frequency trading arms race: Frequent batch auctions as a market design response, *Fama-Miller Working Paper* pp. 14–03.

Capponi, Agostino, and Ruizhe Jia, 2021, The adoption of blockchain-based decentralized exchanges, working paper Columbia University https://ssrn.com/abstract=3805095.

Daian, Philip, Steven Goldfeder, Tyler Kell, Yunqi Li, Xueyuan Zhao, Iddo Bentov, Lorenz Breidenbach, and Ari Juels, 2019, Flash Boys 2.0: Frontrunning, Transaction Reordering, and Consensus Instability in Decentralized Exchanges, Working paper Cornell Tech.

Foucault, Thierry, 1999, Order flow composition and trading costs in a dynamic limit order market1, *Journal of Financial Markets* 2, 99–134.

Huang, Roger D., and Hans R. Stoll, 1997, The components of the bid-ask spread: A general approach, *The Review of Financial Studies* 10, 995–1034.

KyberNetwork, 2019, Kyber: An on-chain liquidity protoco, Whitepaper Kyber Network Foundation https://files.kyber.network/Kyber_Protocol_22_April_v0.1.pdf.

Kyle, Albert S., 1985, Continuous auctions and insider trading, *Econometrica* 53, 1315–1336.

Malinova, Katya, Andreas Park, and Ryan Riordan, 2013, Do retail traders suffer from high frequency traders?, Working paper University of Toronto.

Martinelli, Fernando, and Nikolai Mushegian, 2019, A non-custodial portfolio manager, liquidity provider, and price sensor, Whitepaper Balancer Foundation https://balancer.finance/whitepaper/.

Parlour, C., 1998, Price dynamics in limit order markets, *Review of Financial Studies* 11, 789–816.

Parlour, Christine, and Alfred Lehar, 2021, Decentralized exchanges, Working paper University of Calgary link.

van Kervel, Vincent, Amy Kwan, and P. Joakim Westerholm, 2020, Order splitting and interacting with a counterparty, Working paper Pontificia Universidad Católica de Chile https://ssrn.com/abstract=3516199.

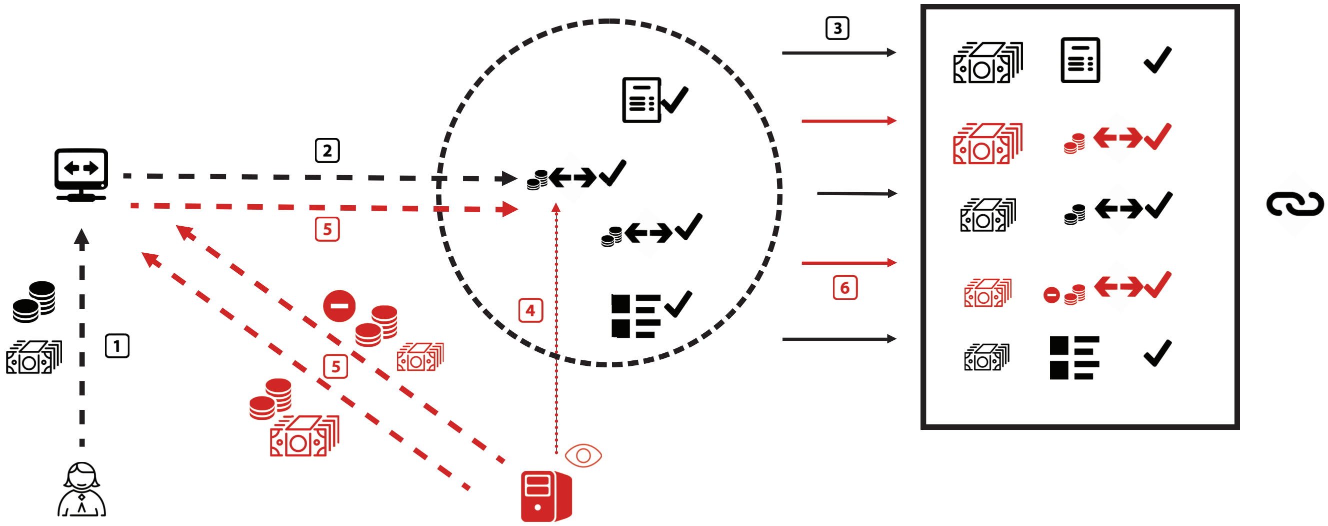


Figure 1
Illustration of Front-Running in Blockchains

The possibility of front-running is an intrinsic feature of blockchains. The schematic works as follows. A user who wants to perform a swap transaction submits the tokens she desires to exchange to the constant product market making contract (1). The contract submits an atomic swap to the blockchain network, and upon verification this transactions enters the mem-pool (2). Verified transactions get ordered in a block based on the fees that they offer (all else equal) (3). An attacker (likely a bot) observes the mempool and see a transaction that can be front-run profitably (4). The bot sends two off-setting swap transactions to the contract, when the front-running trade has a higher fee than the original, front-run transaction (5). In the block, the transactions now get re-ordered according to their mining fees and, upon inclusion on the chain, the transactions in the CPAMM contract get executed in the order of the fees (6).

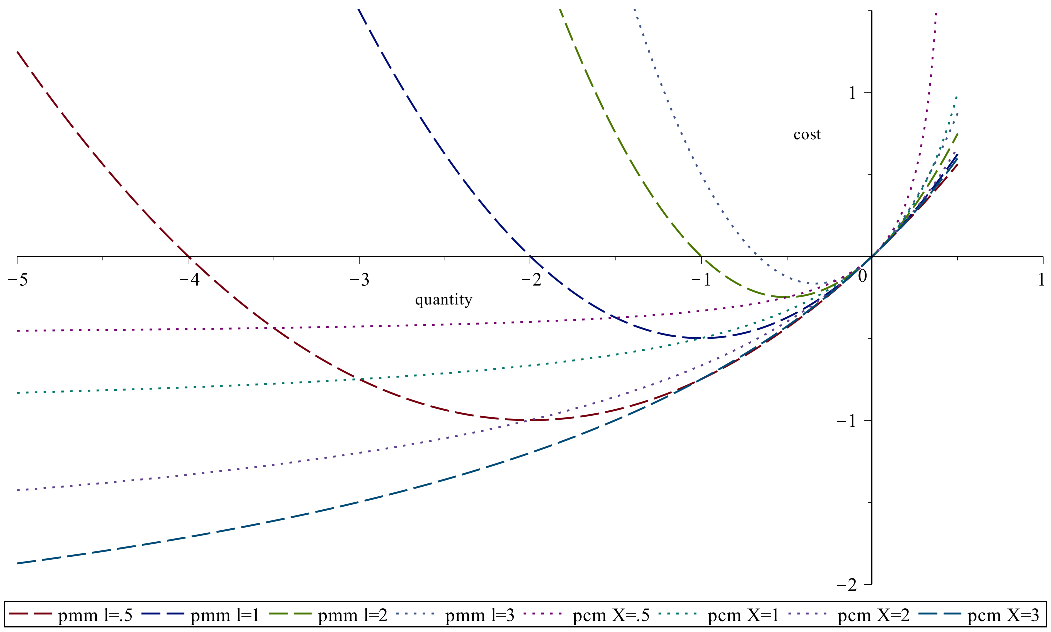
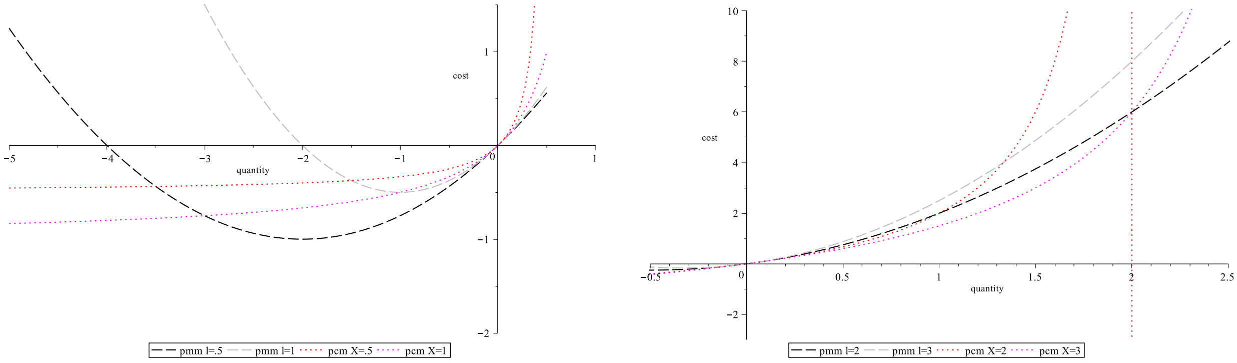


Figure 3
Comparison of the Cost Functions p^{tmm} and p^{cmm}



Panel A: $x^* < 0$

Panel B: $x^* > 0$

Figure 4
Comparison of the Cost Functions for $x^* < 0$ and $x^* > 0$