Optimal Trading Against Sandwich Attack Frontrunning on Constant Product Market Maker Exchanges

Abstract

Decentralized exchanges allow users to trade assets through automated pricing rules without intermediaries, but these markets are vulnerable to adversarial trading strategies that exploit transaction ordering. A prominent example is the sandwich attack, in which a trader places orders before and after a user's trade to profit from the induced price movement. This paper studies a dynamic trading game between a user and a strategic frontrunner on the popular cryptocurrency exchange form, a constant product market maker, emphasizing the role of fixed transaction costs such as fees required to obtain priority execution. I show that when these fixed costs are sufficiently large, a user can deter frontrunning by splitting a trade into smaller transactions, making each individual trade unprofitable to attack. The analysis characterizes the frontrunner's optimal response, derives conditions under which trade splitting dominates single-trade execution by reducing the frontrunning. The results highlight a simple, user-level mechanism for reducing adversarial trading that operates within existing automated market designs, without relying on private order flow, batch auctions, or protocol-level changes.

I - Introduction

Decentralized financial markets have grown rapidly in scale and economic relevance over the past decade, with decentralized exchanges processing billions of dollars in daily trading volume and supporting a wide range of financial activities. These markets are often motivated by the promise of lower transaction costs, greater transparency, and reduced reliance on intermediaries. In particular, transaction fees on blockchain-based payment and exchange systems are frequently cited as being substantially lower than those charged by traditional payment processors, such as credit card networks that typically impose fees on the order of several percentage points per transaction. As a result, blockchain-based market infrastructure has attracted increasing interest not only from retail users but also from established financial institutions exploring distributed ledger technology as a potential foundation for future trading, clearing, and settlement systems. Notably, major exchanges such as Nasdaq and the London Stock Exchange have publicly explored or piloted blockchain-based solutions for market infrastructure, raising the stakes of understanding how these systems perform under strategic behavior.

Despite their potential efficiency gains, decentralized markets introduce new vulnerabilities that arise from the way transactions are processed and executed. Decentralized finance (DeFi) platforms rely on smart contracts—self-executing pieces of code deployed on blockchains—to implement financial services without centralized control. Within DeFi, decentralized exchanges allow users to trade assets directly through automated pricing rules rather than centralized order books. The most widely used design is the constant product market maker, in which prices are determined algorithmically as a function of asset reserves supplied by compensated liquidity providers. Users submit transactions specifying trade size and acceptable price slippage, after which these transactions are broadcast to the network and included in blocks by validators.

A key feature of these systems is that transactions are not executed instantaneously or in a predetermined order. Instead, pending transactions are publicly visible before execution and are ordered by validators, typically according to transaction fees or priority payments. This institutional detail creates opportunities for strategic traders to exploit ordering discretion. In particular, a class of adversarial strategies known as sandwich attacks involves placing trades immediately before and after a user's transaction in order to profit from the price impact generated by the user's trade. The economic effect of such behavior is a transfer of surplus from ordinary users to

frontrunners and validators, increasing effective trading costs without providing additional liquidity or improving price discovery. This phenomenon is commonly referred to as Miner or Maximal Extractable Value (MEV) and has been shown empirically to account for substantial value extracted from users on major blockchain networks.

This paper studies whether users can mitigate sandwich attacks through execution strategy alone, without relying on protocol-level modifications or trusted intermediaries. I develop a dynamic trading model in which a user interacts with a strategic frontrunner on a constant product market maker. The central friction in the model is that frontrunning requires the payment of fixed transaction costs, such as network fees or priority bids, in order to secure favorable transaction ordering. While these costs are easily justified when attacking a large trade, they must be paid repeatedly when trades are split into smaller components. The analysis shows that when fixed costs are sufficiently high—such as during periods of network congestion—a user can optimally deter frontrunning by splitting a trade into smaller transactions, rendering each individual attack less or completely unprofitable. I characterize the frontrunner's optimal response, derive conditions under which trade splitting dominates single-trade execution, and identify fee regimes in which extractive trading incentives are reduced endogenously. It does not require precommittment and it could be implemented by users manually or more likely and easily algorithmically, or it can be used by existing protocols alongside their current exchanges. The results highlight a simple, user-level mechanism for reducing adversarial trading that operates within existing automated market maker designs and may be relevant for the broader adoption of blockchain-based market infrastructure.

II - Literature

The existence and economic significance of Miner Extractable Value (MEV) was first systematically documented by Daian et al. (2019), who show that transaction reordering in permissionless blockchains enables profitable frontrunning, arbitrage, and liquidation strategies that extract surplus from users. Using Ethereum transaction data, they characterize MEV as arising from a generalized first-price auction over transaction ordering and show that competition among extractors transfers a substantial share of surplus to validators while imposing negative externalities on users via higher transaction fees and worse execution prices. Note that my original design was from 2021 for clarity on influences versus what is now clear in the literature below.

Subsequent work has focused on specific MEV strategies and their empirical prevalence. Zhou et al. (2020) provide the first detailed empirical study of sandwich attacks on constant product market makers (CPMMs), using the popular Uniswap transaction data to identify profitable frontrunning opportunities and estimate attacker revenues. They document that the majority of Ethereum clients order transactions primarily by gas price (fixed transaction validation fee), enabling sandwich attacks whenever trades are sufficiently large relative to pool liquidity and user-specified slippage bounds. They further show that competition among frontrunners erodes profits but does not eliminate the incentive to attack, especially for high-value trades.

More recent literature has explored protocol-level (the website with some smart contract algorithm for users) solutions aimed at reducing or eliminating MEV. One branch focuses on fair transaction ordering, proposing mechanisms that constrain reordering power. Protocols such as Aequitas (Kelkar et al., 2022) and Themis (Zhang et al., 2022) attempt to enforce ordering rules based on receive times or cryptographic commitments. While these approaches can theoretically nearly eliminate certain forms of MEV, they require substantial changes to blockchain consensus and face practical challenges related to network latency and incentive compatibility. Determining the time of incoming transactions can be difficult with many nodes that need consensus as compared to single node systems prevalent in normal financial markets. Such protocols would need transparent smart contracts to participate, given that users often desire trustlessness.

Another strand of research investigates batch auctions and intent-based execution mechanisms. In this strain of idea first there were Flashbots. Flashbots introduced private transaction relays and proposer–builder separation, which reduce failed frontrunning attempts and raise the effective cost of priority transaction inclusion. As a result,

while MEV extraction becomes more efficient, the fixed cost of profitable sandwich attacks increases, reinforcing user-side strategies that rely on making frontrunning unprofitable. If your orderer of transactions will share some of the potential MEV then it is a reduction. Newer CowSwap and related designs aggregate trades over discrete intervals and clear them at a uniform price, effectively removing the temporal advantage required for sandwich attacks. Empirical and theoretical analyses (e.g., Obadia et al., 2022) show that such mechanisms substantially reduce sandwiching, though they introduce new solver-based intermediaries and rely on off-chain coordination. These designs represent a departure from continuous-time CPMMs rather than a modification within them.

A third line of work studies privacy-based mitigations, including encrypted mempools (incoming unordered transactions), trusted execution environments, and commit–reveal schemes, which hide transaction details until execution by separating commitment and revelation. Examples include MEV-SGX (Breidenbach et al., 2022) and Flashbots' SUAVE architecture, which aim to prevent frontrunners from observing transaction details prior to execution. While effective in limiting information leakage, these approaches introduce additional trust assumptions and infrastructure complexity. Furthermore, Angeris et al. (2021) show that privacy mechanisms alone are insufficient to prevent frontrunning in constant function market makers, as adversaries can infer transaction size and direction from observable price and reserve updates.

Recent work has emphasized slippage and execution quality as primary channels through which MEV affects users of automated market makers. Adams et al. (2023) provide a detailed empirical analysis of swaps on Uniswap v3, decomposing realized execution costs into benign price impact and adversarial components attributable to frontrunning and sandwich attacks. They document that even in highly liquid pools such as WETH–USDC, average trading costs are on the order of 20–25 basis points, while less liquid or more volatile pairs can exhibit costs exceeding 100 basis points per trade. Moreover, execution costs increase sharply with trade size: their estimates imply that an additional $ 1 million in trade value is associated with roughly 10–15 basis points of incremental slippage beyond mechanical price impact, consistent with increased exposure to MEV strategies. These findings motivate mechanism-level interventions that reshape how large trades interact with liquidity. Protocols such as SlowSwap and related slippage-limited or time-smoothing CFMM designs aim to reduce MEV by spreading the price impact of large trades across time or virtual liquidity, thereby lowering the marginal profitability of sandwich attacks. Such approaches mitigate MEV by modifying the pricing mechanism itself, in contrast to strategies that operate at the transaction-ordering or user-decision level. It does have some other notable differences such as it precommits beforehand and assumes equal size trades are best, it incorporates information only before it submits the first trade and can't be canceled.

Motivated by these findings, a growing line of work proposes modifying automated market maker designs to reduce the marginal profitability of large, fast-executing trades. Protocols such as SlowSwap and related slippage-limited or time-smoothing CFMM mechanisms alter how price impact is realized by effectively spreading large trades across time or virtual liquidity, thereby dampening the immediate price movements that enable sandwich attacks. By slowing the execution of large orders, these designs reduce the ability of frontrunners to extract value from short-term price impact while preserving continuous liquidity provision. Such approaches mitigate MEV by modifying the pricing mechanism itself, in contrast to strategies that operate at the transaction-ordering layer or through user-side execution decisions.

The existing literature has largely neglected user-side strategic responses to MEV within standard CPMM exchanges. This paper contributes by formally analyzing a simple but practical strategy available to traders on existing decentralized exchanges: splitting trades over time to reduce the profitability of sandwich attacks when fixed transaction costs are sufficiently high. Rather than eliminating information leakage or reordering power, the mechanism exploits the fixed cost structure of frontrunning—primarily gas fees and priority bids—to push attacker profits below zero for sufficiently small trades. This approach requires no protocol changes, no trusted intermediaries, and no modification of the CPMM pricing rule. You could create a smart contract that does this

automatically, including calculating all the needed sizes of trades.

By modeling the interaction between a trader and a frontrunner as a dynamic game under CPMM pricing, this paper characterizes the conditions under which trade splitting dominates single-trade execution and identifies parameter regimes—particularly high gas-price environments—where MEV extraction becomes unprofitable. In this sense, the paper complements protocol-level MEV mitigations by demonstrating how user behavior alone can partially neutralize frontrunning incentives within existing decentralized exchange infrastructures.

This paper starts in part III by describing the game. In part IV the front-runners profit maximization and best response will be described. In part V the victims best responses will be described. In part VI I will do dynamic statics and compare splitting trades. In part VII alternative responses of the victim and setups are discussed. Part VIII concludes.

III - Game setup

This game is aiming to describe a person who is trying to submit a trade between any two tokens against a frontrunner who is aiming to earn profit at their expense by paying transaction validators extra fees to place trades before and after the other person which gives the other person worse prices to purchase at. This game describes generally trading some token a for b on the most commonly structured decentralized exchanges using automatic market makers, this a and b could be tokenized dollars or real world assets or some cryptocurrency being traded for any other token. This works for most blockchains with smart contracts, including popular ethereum and solana.

There is a frontrunning paper Brunnermeier Pedersen 2004 which characterizes predatory trading on traders who need to deleverage after some shock and then other traders who are aware begin exiting their position and buy back afterwards to profit which influenced the setup for this paper but the pricing mechanism is exogenous and accomplished with a long term demand equation with traders consistently buying, whereas this paper will be focusing more on the frontrunning aspect he describes in a subsection of the paper with the appropriate differences in the blockchain environment.

I will be modeling the pricing with the most commonly used pricing algortihm for decentralized exchanges. I will be following a similar notation and pricing setup as was used in Angeris et al. 2019. These exchanges take in deposits from market makers using smart contracts and then algorithmically quote prices to potential users. The users tell the site how much they would like to trade and give it a maximum price that they would accept (price slippage) and then a smart contract is generated and sent to validators to execute. This form of pricing is more generally referred to as a constant function market making (CFMM) algorithm and this particular commonly used form is called the constant product market maker (CPMM) which is used by exchanges such as Uniswap V2, Pancakeswap, Raydium, etc. This model would not be difficult to adapt to other constant function market making methods such as those used by other popular exchanges like Balancer or Curve. However there are some more complicated exchanges such as newer forms of Uniswap which this method should work with but it would need a more complicated adjustment because for that example liquidity is prodvided over ranges instead of any price for market makers and this affects the size of price changes following trades.

This method of pricing has two reserves for the two tokens R_a, R_b. The USD value of the two at the time of depositing into reserves is equal but they will be representing the number of each tokens in the one exchange being used. In the CPMM the product of the two $R_a R_b = k$ is constant and the current spot price in token b is $\frac{R_a}{R_b}$. When a trader trades Δa for Δb the reserves update to $R_a + \gamma \Delta a$ and $R_b - \Delta b$ after accounting for the exchange's transaction fee γ. The product remains constant across time so to figure out the Δb that is returned for a given Δa we solve for Δb with the new reserve balances from that trade.

$$(R_a + \gamma \Delta a)(R_b - \Delta b) = k = R_a R_b \iff \Delta b = R_b - \frac{R_a R_b}{R_a + \gamma \Delta a}$$

The trader is quoted at price $\frac{\Delta a}{\Delta b}$ if they want to spend Δa. The option that the trader uses to specify the highest price they are willing to accept for the contract to execute is called the slippage. The slippage option guarantees the worst price the trade executes at is $\frac{\Delta a}{\Delta b}(1+s)$. The s is some price difference it can execute at compared to the price that was quoted. It can differ because others might buy and sell it before your trade actually submits to the blockchain record of transactions. If the price exceeds s percent above the quoted amount then the smart contract will not execute which eliminates the chance for frontrunning profitably. This is chosen by the user however in practice it's left as the default option, which is also observed in the Zhou et al 2020 paper. I consider it fixed for now and in a later section I share a specification that I think represents the new trader's problem with it as a choice variable.

The game is modeling a basic sandwich attack in which there is someone who wants to trade some amount of token a at a decentralized exchange and they are quoted for some amount of token b but before it goes through a frontrunner observes it and exogenously bids on their transaction fees to get the right to place a trade before and after the victim. Then when the trades finally submit to the blockchain ledger the frontrunner has placed their optimally sized trade before the other person's transaction which now should have a higher price because by buying ahead of them the frontrunner pushed up the price. Then the frontrunner exits their position once the other person has purchased the token b. When the victim's trade executes it pushes up the value of token b which the frontrunner now holds and then the frontrunner sells their holdings of that token, netting the difference between this value and what money they used to enter that position.

The game has discrete time periods which represent at least the time necessary to execute one new bundle (block) of transactions processed. The future periods payoffs are discounted by β. The victim could do a single trade or split up their trade over multiple periods. They may prefer a single trade if they are impatient or the fixed network transaction fee g is high. Potentially you could structure a split trade such that the profit is not sufficient for a frontrunner to participate because the required bid b is too high and that could outweigh these other effects. I am assuming the trades are not failing on account of excessive slippage unless the FR pushes the price past it, that is there are no other agents shifting prices nor probabilistic shifts in it. This is less reasonable the longer the the time periods are assumed to be. I assume there are enough reserves to accommodate the trades. I am currently assuming the frontrunning works each time which isn't necessarily true since some validator clients may not order transactions perfectly by network transaction fees as mentioned in Zhou et al. 2020.

I am adding in a mathematical tool to represent the fact that there are multiple exchanges of varying reserves and prices, the arbitragers force prices to match amongst them. This is not accounted for in the Angeris et al 2019 result, by assuming one exchange they are essentially assuming the exchange is always at the market price and that the exchange has identical reserves to the entire market reserves. The arbitrager is not a player. Between periods reserves update from the previous trade to be the equal to the total market reserves split evenly amongst all the D equally competitive exchanges so their prices match. If there is worse market power you could just reduce the number of exchanges and the updates will be less important. Arbitrage between periods would make splitting up trades more appealing than the case where there is a single exchange. This is because with multiple exchanges the reserves are smaller and there is less liquidity for the trade which means that there is a greater price change when trading and a worse quoted price. So they could execute one big trade with a large price change or if they split it up the price changes for each period are partially offset when the price is reduced a little from the arbitrager shifting reserves around to bring the price back down to the market price. This need not be done if more simplicity is desired.

The interaction between the trader (the victim) and the frontrunner is modeled as a dynamic game with discrete periods corresponding to transaction blocks. In each period, the victim may submit a trade to exchange an amount of token Δa for token b on a constant product market maker. The victim's objective is to maximize the total amount of token b received net of fixed transaction fees, taking into account the possibility of frontrunning.

Formally, the victim chooses a sequence of trades to maximize $\sum_{t=1}^{T}(\Delta b_{a,t} - g)$, where $\Delta b_{a,t}$ denotes the amount of token b actually received after frontrunning in period t, and g is a fixed exogenous transaction fee paid each time a trade is submitted. Both the victim and the frontrunner pay a proportional exchange fee $1 - \gamma$ on their input tokens.

The timing within each period is as follows. First, the victim submits a transaction specifying a trade of size Δa, which becomes publicly observable prior to execution. Second, a strategic frontrunner observes the pending transaction and decides whether to engage in a sandwich attack. If so, the frontrunner chooses an amount Δa_f to trade before the victim, receiving Δb_f according to the market maker pricing rule. The frontrunner's objective is to maximize profit $\Delta a_{f2} - \Delta a_f - b$, where Δa_{f2} is the amount of token a recovered when the frontrunner exits their position after the victim's trade, and b represents the fixed cost required to obtain favorable transaction ordering. This cost captures priority fees or bidding costs in the ordering auction and is treated as exogenous, reflecting competitive conditions among frontrunners.

If the frontrunner attacks, they pay the cost b and their initial trade executes ahead of the victim's transaction, yielding Δb_f. The victim's transaction then executes at a worse price due to the induced price impact, yielding $\Delta b_a < \Delta b$, where Δb denotes the counterfactual amount absent frontrunning. Finally, the frontrunner exits their position by trading Δb_f back into token a, receiving Δa_{f2}. If the frontrunner does not attack, the victim's trade executes at the quoted price net of exchange fees. This process repeats across periods until the victim exhausts their total budget A, or until the remaining amount of token a is insufficient to cover the fixed transaction fee g.

The model assumes that the frontrunner attacks at most one victim transaction per period and fully unwinds their position within the same period. An alternative strategy of entering a large position before the victim's first trade and exiting only after the victim completes all trades could be profitable if the frontrunner infers the victim's total budget A. As discussed in Section VII, such strategies can violate the slippage constraint and collapse the equilibrium to a single frontrun trade. I abstract from this case because delaying exit exposes the frontrunner to price risk, ties up capital that could be deployed elsewhere, and would require a Bayesian formulation due to heterogeneity in victims' discount factors β. Moreover, split trade execution need not involve commitment, allowing the victim to stop early, and frontrunners face a population of heterogeneous and often naive traders.

Under these assumptions, the equilibrium concept is Nash equilibrium. For each trade submitted by the victim, the frontrunner has a best response Δa_f^* that solves their profit maximization problem. Anticipating this response, the victim chooses the sequence $\Delta a_t{}_{t=1}^{T}$ to maximize total net receipts of token b after fees and frontrunning. The resulting outcome reflects mutually optimal behavior given the trading technology and cost structure of the market.

IV - Front-runner's problem

The Front-runner, FR optimizes their profits $\Delta a_{f2} - \Delta a_f - b$ and chooses Δa_f. The chosen variable Δa_f is the amount of token a they use in the first trade before the victim. The Δa_{f2} is the amount of token a they get back after the frontrunning is complete. b is equal to $g +$ premium where this premium is determined exogenously by things like competition between front-runners or if there are other profitable arbitrage opportunities it becomes costlier to put the transactions where desired for the frontrunning. The b which is the expected bid required for the sandwich attack could be estimated using recent past transactions found on a blockchain explorer website like etherscan. There are constraints associated with the pricing mechanism, a non-negativity constraint and one for the price slippage of the victim. The optimization is as follows

$$\max_{\Delta a_f} \pi = \max_{\Delta a_f} \Delta a_{f2} - \Delta a_f - b$$

$$st : \frac{\Delta a}{\Delta b}(1+s) \geq \frac{\Delta a}{\Delta b_a} \tag{1}$$

$$\Delta b_f = R_b - \frac{R_a R_b}{R_a + \gamma \Delta a_f} \tag{2}$$

$$\Delta b = R_b - \frac{R_a R_b}{R_a + \gamma \Delta a} \tag{3}$$

$$\Delta b_a = R_b - \Delta b_f - \frac{(R_a + \gamma \Delta a_f)(R_b - \Delta b_f)}{R_a + \gamma \Delta a_f + \gamma \Delta a} \tag{4}$$

$$\Delta a_{f2} = R_a + \gamma \Delta a + \gamma \Delta a_f - \frac{(R_a + \gamma \Delta a_f + \gamma \Delta a)(R_b - \Delta b_f - \Delta b_a)}{R_b - \Delta b_f - \Delta b_a + \gamma \Delta b_f} \tag{5}$$

$$\Delta a_f, \Delta b_f \geq 0 \tag{6}$$

The FR does not pay b if they choose to not front-run $\pi = \Delta a_f = \Delta a_{f2} = 0$. (1) is The slippage setting that terminates the trade when the price has moved too high. Slippage is not endogenous so it isn't needed further. (2)-(5) are constraints that come from the constant product pricing mechanism used on the exchanges, you can see it above in the setup section 3. They make sure there is a constant product of reserves in the exchange to price the token b in terms of publicly known parameters and inputs. (2) is the amount of token b that FR gets from trading Δa_f. (3) is the amount of token b that the victim would get from trading Δa absent a frontrunner, but it doesn't execute. (4) is the amount of token b that the victim actually gets after being front-runned, they trade Δa for Δb_a. (5) is the amount of token a that FR gets after exiting their original position, they trade Δb_f for Δa_{f2}. (6) is a positive trade constraint, you can't trade negatively here.

I am going to find the best response of the front-runner using first order conditions of this problem and then with an unrestrictive assumption on reserves you can derive a formula for Δa_f for a given victim's Δa that is found with the

Proposition 1 Assume reserves $R_a \geq 1, R_b \geq 1$. Assume Δa is sufficiently large with sufficiently high reserves of R_a and R_b. Alternatively the 2nd assumption is a trivial case if $\gamma = 1$. Then for either, profits π are increasing in Δa_f and the interior solution for the optimal choice of Δa_f is

$$\Delta a_f = \frac{\sqrt{\gamma^2(2R_a + \gamma \Delta a)^2 + 4\gamma^2 s R_a(R_a + \gamma \Delta a)} - (2R_a + \gamma \Delta a)\gamma}{2\gamma^2}$$

Proof. First substitute into the payoff equation the constraints at equality (2),(3),(4),(5), leaving only constraints (1),(6). We now look for conditions of monotonicity. Take a derivative with respect to Δa_f on it to get

$$\frac{\partial \pi}{\partial \Delta a_f} = (\gamma - 1) - \frac{\partial}{\partial \Delta a_f}\left(\frac{(R_a + \gamma \Delta a_f + \gamma \Delta a)(R_b - \Delta b_f - \Delta b_a)}{R_b - \Delta b_f - \Delta b_a + \gamma \Delta b_f}\right) = (\gamma - 1) - R_a R_b \frac{\partial}{\partial \Delta a_f}\left(\frac{1}{R_b - \Delta b_f - \Delta b_a + \gamma \Delta b_f}\right)$$

$$= (\gamma - 1) + R_a R_b \left(\frac{1}{R_b - \Delta b_f - \Delta b_a + \gamma \Delta b_f}\right)^2 \frac{\partial}{\partial \Delta a_f}((\gamma - 1)\Delta b_f - \Delta b_a)$$

$$= (\gamma - 1) + R_a R_b \left(\frac{1}{R_b - \Delta b_f - \Delta b_a + \gamma \Delta b_f}\right)^2 R_a R_b \gamma \left[\frac{\gamma}{(R_a + \gamma \Delta a_f)^2} - \frac{1}{(R_a + \gamma \Delta a_f + \gamma \Delta a)^2}\right]$$

The second equality is substituting in the constraints (4) and then (2). Evaluating the last part of the third equality's derivative gave

$$(\gamma - 1)\frac{\partial \Delta b_f}{\partial \Delta a_f} = \frac{(\gamma - 1)\gamma R_a R_b}{(R_a + \gamma \Delta a_f)^2} < 0$$

$$(-1)\frac{\partial \Delta b_a}{\partial \Delta a_f} = \frac{R_a R_b \gamma}{(R_a + \gamma \Delta a_f)^2} - \frac{R_a R_b \gamma}{(R_a + \gamma \Delta a_f + \gamma \Delta a)^2}$$

This shows Δb_f is increasing in Δa_f, but obviously since it has more of an input token. The Δb_a is decreasing in Δa_f if we simply assume $R_a + \gamma \Delta a_f \geq 1$, that at least 1 token is in the reserve. This means when FR spends more then the victim receives less due to a worse price.

The second term in the final equality part with $R_a R_b (\frac{1}{R_b - \Delta b_f - \Delta b_a + \gamma \Delta b_f})^2 R_a R_b \gamma$ is all positive. So all that matters for monotonicity and all that requires any assumptions is the part $[\frac{\gamma}{(R_a+\gamma \Delta a_f)^2} - \frac{1}{(R_a+\gamma \Delta a_f+\gamma \Delta a)^2}]$, it will be positive if either $\gamma \approx 1$ or if $\gamma \Delta a$ is sufficiently high to make the second part being subtracted away smaller than the first part. If $\gamma \approx 1$ then $(\gamma - 1)$ in the final equality is approximately zero so it would imply profits increase in Δa_f since the second term was positive. Or if Δa was large then the $\gamma - 1$ is something between -1 and zero and so the large second term needs to be larger than 1.

For this whole derivative to be positive, after rearranging the above with first derivative > 0, it depends on the parameters being such that

$$(\frac{1}{R_b - \Delta b_f - \Delta b_a + \gamma \Delta b_f})^2 [\frac{\gamma}{(R_a + \gamma \Delta a_f)^2} - \frac{1}{(R_a + \gamma \Delta a_f + \gamma \Delta a)^2}] < .5 R_a^2 R_b^2 \gamma$$

It is known $(\frac{1}{R_b - \Delta b_f - \Delta b_a + \gamma \Delta b_f})^2 \gamma [\frac{\gamma}{(R_a+\gamma \Delta a_f)^2} - \frac{1}{(R_a+\gamma \Delta a_f+\gamma \Delta a)^2}] < \frac{1}{R_b^2} [\frac{\gamma}{(R_a+\gamma \Delta a_f)^2} - \frac{1}{(R_a+\gamma \Delta a_f+\gamma \Delta a)^2}]$, but something more ambiguous would be that we need

$$\frac{1}{R_b^2} [\frac{\gamma}{(R_a + \gamma \Delta a_f)^2} - \frac{1}{(R_a + \gamma \Delta a_f + \gamma \Delta a)^2}] < .5 R_a^2 R_b^2 \gamma$$

The $.5 R_a^2 R_b^2 \gamma$ is almost always above 1 since reserves are usually large and the part in the brackets will between zero and 1. The $\frac{\gamma}{(R_a+\gamma \Delta a_f)^2}$ is some number $\gamma \in [0, 1]$ divided by some squared number already assumed to be above 1 by assuming reserves are above 1. So it is between zero and 1 and the whole bracketed terms are positive by the pricing equation for Δa. The right side of the inequality is greater than zero but growing in either reserves, the left side of the equality is decreasing in either reserve when $[\frac{\gamma}{(R_a+\gamma \Delta a_f)^2} - \frac{1}{(R_a+\gamma \Delta a_f+\gamma \Delta a)^2}] > 0$ which holds under the assumptions. So for sufficiently large reserves this inequality holds and profits increase in Δa_f. These reserves are worth much more than this unless its an edge case. The other potential assumption of a sufficiently large Δa is maybe restrictive but γ is about .998 or higher with many of these exchanges.

So under the parameter assumptions necessary to establish monotonicity in Δa_f you would purchase as much as is possible or zero if the max profit was below zero. The slippage constraint will become closer to binding as Δa_f increases because Δb_a declines in it. So the FR purchases Δa_f such that the slippage constraint (1) binds. This equality yields

$$\begin{aligned}
\Delta b_a(1+s) = \Delta b \iff (1+s)(R_b - R_b + \frac{R_a R_b}{R_a + \gamma \Delta a_f} - \frac{R_a R_b}{R_a + \gamma \Delta a_f + \gamma \Delta a}) &= R_b - \frac{R_a R_b}{R_a + \gamma \Delta a} \\
(1+s)(\frac{R_a R_b \gamma \Delta a}{(R_a + \gamma \Delta a_f + \gamma \Delta a)(R_a + \gamma \Delta a_f)}) &= \frac{R_b \gamma \Delta a}{R_a + \gamma \Delta a} \\
\frac{(1+s) R_a R_b \gamma \Delta a (R_a + \gamma \Delta a)}{\gamma \Delta a R_b} &= (R_a + \gamma \Delta a_f + \gamma \Delta a)(R_a + \gamma \Delta a_f) \\
\gamma^2 \Delta a_f^2 + \Delta a_f \gamma^2 (2R_a + \gamma \Delta a) - s R_a (R_a + \gamma \Delta a) &= 0 \\
\Delta a_f = \frac{\sqrt{\gamma^2 (2R_a + \gamma \Delta a)^2 + 4\gamma^2 s R_a (R_a + \gamma \Delta a)} - (2R_a + \gamma \Delta a)\gamma}{2\gamma^2}
\end{aligned}$$

In the first step (2) is substituted to reduce the numerator and in the final step here I ignore the negative potential root for this quadratic since we require it to be positive by constraint (6). This increases in Δa, the FR purchases more if the victim does too. All of this is to say that we assume the parameters and reserves preserve monotonicity of profits then the front runner will bring the execution price for the victim as close to the maximum allowed price through slippage choices as safely possible. **This completes the proof**.

This fits Zhou et al. 2020 finding that a lower slippage will make profits worse for FR. For an idea of what a profitable FR might require for Δa on a relatively liquid pair of tokens at the time of Zhou et al. analysis, they state for a popular pair Dai and ethereum it required a trade of at least 15-25 ethereum valued at 197 dollars at the time of their analysis. Then for that particular pair if Dai (dollars) is token a and Ethereum is token b, then our transactions must be at least size $\Delta a \geq 2955$. This minimum input is likely lower now since there is more liquidity in these markets now.

When the proposition assumptions do not hold then the FR would set the first derivative to zero $\frac{\partial \pi}{\partial \Delta a_f} = 0$ and solve for the roots of the equation and test each of them to see which is the largest profits. If none are positive they take the corner solution where no trade occurs. This potential method makes the optimal victim's choices much more difficult to calculate as it would require a numerical solution . The next section describes what the victim's best response is in reaction to the shown frontrunner's best response.

V The victim's problem

The trader has the option of spending all of their budget in 1 period or splitting the trade into multiple trades. FR will frontrun each individual trade if it is positive profits and the period ends once FR exits their position. The victim wants to maximize their total output in token b. There can't be infinite periods with tiny Δa due to the fixed network transaction costs g. I choose linear utility for practical reasons but any concave utility function would do. So the victim will take the output of Δb after doing a single trade or we maximize the following problem which may be done by the victim if it gives more of token b.

$$\max_{\Delta a_t} \sum_{t=1}^{T} \beta^{t-1}(\Delta b_{a,t} - g)$$

$$St. \Delta b_t = R_{b,t} - \frac{R_{a,t}R_{b,t}}{R_{a,t} + \gamma \Delta a_t} \tag{7}$$

$$\Delta b_{f,t} = R_{b,t} - \frac{R_{a,t}R_{b,t}}{R_{a,t} + \gamma \Delta a_{f.t}} \tag{8}$$

$$\Delta b_{a,t} = R_{b,t} - \Delta b_{f,t} - \frac{(R_{a,t} + \gamma \Delta a_{f,t})(R_{b,t} - \Delta b_{f,t})}{R_{a,t} + \gamma \Delta a_{f,t} + \gamma \Delta a_t} \tag{9}$$

$$\sum_{t=1}^{T} \Delta_{a,t} \leq A \tag{10}$$

$$R_{a,t} = \frac{R_{atot,t-1} + \gamma \Delta_{a,t-1} + \gamma \Delta_{af,t-1} - \Delta_{af2,t-1}}{D}, R_{b,t} = \frac{R_{btot,t-1} - \Delta_{ba,t-1} + \gamma \Delta_{bf,t-1} - \Delta_{bf,t-1}}{D} \tag{11}$$

$$R_{a,1} = \frac{R_{atot,1}}{D}, R_{b,1} = \frac{R_{btot,1}}{D} \tag{12}$$

$$\Delta a_{f,t} = \arg\max \pi_t \tag{13}$$

$$\Delta_{a,t}, \Delta_{b,t} \geq 0 \tag{14}$$

The β is a discount factor which could be determined by simply asking the victim how quickly they would like it completed in. We maximize the net $\Delta b_a - g$ instead of Δb because Δb_a accounts for how much of the token b you

get after the front-running. The constraints (7)-(9) are straightforwardly describing the constraints that keep the product of reserves constant as is required by the pricing mechanism. (10) is the budget constraint. (11) and (12) describe the transitions of reserves from each period and the initial value. Each period the a reserves add in the input of victim and FR and then remove some from when FR exits their position. The b reserves subtract what FR and the victim initially take out in b and then when FR exits their position they add some back in. This part in 11 with showing the global reserves versus individual is maybe not essential in a basic form but its worse liquidity brings bigger price movements in a single exchange trade. The price would go beyond the market price across it all if they could trade as such and would require arbitrage. This 1/D is coming from the arbitrage between periods. It was assumed that we are evenly distributing the reserves between D exchanges. This makes liquidity and prices better if you wait which helps make splitting trades more attractive. (13) is assuming FR is making their best response and is necessary to keep a Nash equilibrium. (14) is just non-negativity of the trades used.

Proposition 2 Assume the same assumptions of Proposition 1. The optimal Δa_t in a split purchases response with no frontrunning are the highest Δa_t that set the FR's profit to zero and the payoff is

$$B_{split} = \sum_{t=1}^{T} \beta^{t-1}(R_{b,t} - g) - \frac{\sum_{t=1}^{T} \beta^{t-1}(R_{a,t}R_{b,t}) \prod_{l \neq t}(R_{a,l} + \gamma \Delta a_l^*)}{\prod_{t=1}^{T}(R_{a,t} + \gamma \Delta a_t^*)}$$

Proof The $\Delta b_{a,t}$ was found to be increasing in Δa_f while proving theorem 1 under its assumptions. The Δa_f increases in Δa as was also seen in proving theorem 1. The profit of FR was increasing in Δa_f so the response would be to trade the highest Δa each time period that has no frontrunning. There is no frontrunning when $\pi = 0$ while using the optimal Δa_f^* which is a function of Δa. When $\pi = 0$ using the best response Δa_f^*, we can solve for the optimal Δa_t for each period. $\pi = 0$ when

$$0 = \pi = R_{a,t} + \gamma \Delta a_t + (1-\gamma)\Delta a_{f,t} - \frac{R_{a,t}R_{b,t}}{R_{b,t} - \Delta b_{a,t} + (\gamma - 1)\Delta b_{f,t}} - b$$

$$= R_{a,t} + \gamma \Delta a_t + (1-\gamma)\Delta a_{f,t} - b - \frac{R_{a,t}R_{b,t}}{\frac{\gamma R_{b,t}(R_{a,t}+\gamma\Delta a_{f,t})(R_{a,t}+\gamma\Delta a_{f,t}+\gamma\Delta a_t) - R_{a,t}R_{b,t}[(R_{a,t}+\gamma\Delta a_{f,t})(1-\gamma)-\gamma^2\Delta a_t]}{(R_{a,t}+\gamma\Delta a_{f,t})(R_{a,t}+\gamma\Delta a_{f,t}+\gamma\Delta a_t)}}$$

$$= (R_{a,t} + \gamma \Delta a_t + (1-\gamma)\Delta a_{f,t} - b)\{(R_{a,t} + \gamma \Delta a_{f,t})[\gamma(R_{a,t} + \gamma \Delta a_{f,t} + \gamma \Delta a_t) - R_{a,t}(1-\gamma)] - \gamma^2 \Delta a_t R_{a,t}\}$$
$$- R_{a,t}(R_{a,t} + \gamma \Delta a_{f,t})(R_{a,t} + \gamma \Delta a_{f,t} + \gamma \Delta a_t)$$

This will be a rather high degree polynomial in Δa due to the square root in the optimal Δa_f so it isn't easily analyzed. The victim will select the highest real Δa that solves this equation in each period when Δa_f is a best response. For showing the expression for the split trades we see if T=1 then

$$B_{split} = R_{b,1} - \frac{R_{b,1}R_{a,1}}{R_{a,1} + \gamma \Delta a_1^*} - g$$

and if T=2 then

$$B_{split} = R_{b,1} - \frac{R_{b,1}R_{a,1}}{R_{a,1}+\gamma\Delta a_1^*} - g + \beta R_{b,2} - \frac{\beta R_{b,2}R_{a,2}}{R_{a,2}+\gamma\Delta a_2^*} - \beta g$$

$$= \sum_{t=1}^{2}\beta^{t-1}(R_{b,t}-g) - \frac{\beta(R_{a,2}R_{b,2})(R_{a,1}+\gamma\Delta a_1^*)+(R_{a,1}R_{b,1})(R_{a,2}+\gamma\Delta a_2^*)}{(R_{a,1}+\gamma\Delta a_1^*)(R_{a,2}+\gamma\Delta a_2^*)}$$

$$= \sum_{t=1}^{T}\beta^{t-1}(R_{b,t}-g) - \frac{\sum_{t=1}^{T}\beta^{t-1}(R_{a,t}R_{b,t})\prod_{l\neq t}(R_{a,l}+\gamma\Delta a_l^*)}{\prod_{t=1}^{T}(R_{a,t}+\gamma\Delta a_t^*)}$$

This completes the proof

The number of periods T required to complete all of the trades can be found if you determine each optimal Δa_t in order. Eventually you will hit a point where the budget A is used up to the point where the final Δa_T will be equal or less than what is suggested by solving the above equation for it's roots. If this is very small then it could be less than the transaction fee g and you wouldn't be able to use the entire budget unless it is better to accept being frontrun in the previous period by trading $\Delta a_{T-1} + \Delta a_T$ instead of trading just the Δa_T.
The equation from theorem 1 for Δa_f is in most cases increasing in R_a as will be shown in the section VI. $R_{a,t}$ grows when

$$R_{a,t-1} < \frac{R_{atot,t-1}+\gamma\Delta a_{t-1}+\gamma\Delta a_{f,t-1}-\Delta a_{f2,t-1}}{D} = \frac{R_{a,t-1}D+\gamma\Delta a_{t-1}+\gamma\Delta a_{f,t-1}-\Delta a_{f2,t-1}}{D}$$

$$\iff 0 < \gamma\Delta a_{t-1}+\gamma\Delta a_{f,t-1}-\Delta a_{f2,t-1} \iff \gamma\Delta a_{t-1} > \Delta a_{f2,t-1}-\gamma\Delta a_{f,t-1} > b$$

When there is no front running this is always satisfied and so in a no frontrunning equilibrium R_a always grows every period because the victim continues to give away token a for b. But when frontrunning occurs then if $\gamma\Delta a_{t-1} > \Delta a_{f2,t-1} - \gamma\Delta a_{f,t-1} > b$ then R_a will increase each period. This means with frontrunning R_a increases when the victim trades are bigger than the bids the frontrunner needs to bid. When R_a increases each period the Δa_f does too. This pushes up profits and because the FR is able to more profitably trade the victim has a smaller Δa since they increase together. So as time goes on you should expect for the optimal Δa to shrink.

If the victim chooses to do a single trade and get front-runned then they will get

$$B_{st} = \Delta b_{a,1}(A) - g = \frac{R_{a,1}R_{b,1}}{R_{a,1}+\gamma\Delta a_{f,1}} - \frac{R_{a,1}R_{b,1}}{R_{a,1}+\gamma\Delta a_{f,1}+\gamma A} - g = \frac{R_{a,1}R_{b,1}\gamma A}{(R_{a,1}+\gamma\Delta a_{f,1})(R_{a,1}+\gamma\Delta a_{f,1}+\gamma A)} - g$$

The best response of the victim is to choose whichever payoff is greater that incorporates optimal play using the FR's best response. They choose the higher of B_{st} or the B_{split} if these are their choices. The split trades haven't been concluded to be growing, equal, or shrinking over time and solutions would need real world numerical estimations.

VI Comparative statics

First I will take a look at various factors which could drive up front-running Δa_f under the more sufficient liquidity scenario that increases profits as in theorem 1. First slippage's effect.

$$\frac{\partial\Delta a_f}{\partial s} = \frac{R_a(R_a+\gamma\Delta a)}{\sqrt{\gamma^2(2R_a+\gamma\Delta a)^2+4\gamma^2 sR_a(R_a+\gamma\Delta a)}} > 0$$

This has an increasing effect which is to be expected since the Δa_f was chosen to bind the slippage constraint and this would loosen the constraint. So as was shown in Zhou et al. 2020, this shows that higher slippage allows for

more front-running. Now for γ's effect.

$$\frac{\partial \Delta a_f}{\partial \gamma} = \frac{2\gamma(2R_a+\gamma\Delta a)^2 + 8\gamma s R_a(R_a+\gamma\Delta a) + 4\gamma^2 s R_a \Delta a}{4\gamma^2\sqrt{\gamma^2(2R_a+\gamma\Delta a)^2 + 4\gamma^2 s R_a(R_a+\gamma\Delta a)}} - \frac{(2R_a+\gamma\Delta a)+\Delta a\gamma}{2\gamma^2} + \frac{(2R_a+\gamma\Delta a)}{2\gamma^2}$$
$$- \frac{\sqrt{\gamma^2(2R_a+\gamma\Delta a)^2 + 4\gamma^2 s R_a(R_a+\gamma\Delta a)} - (2R_a+\gamma\Delta a)\gamma}{\gamma^3}$$

This is not known for every set of parameters but the first and final fraction there will be the predominate part of what determines increasing or decreasing and with $\gamma \approx 1$ it appears that it will be decreasing due to the large denominator on the first fraction. The effect of R_a

$$\frac{\partial \Delta a_f}{\partial R_a} = \frac{\frac{1}{2}\sqrt{\gamma^2(2R_a+\gamma\Delta a)^2 + 4\gamma^2 s R_a(R_a+\gamma\Delta a)}(4(2R_a+\gamma\Delta a)\gamma^2 + 4\gamma^2 s R_a + 4\gamma^2 s(R_a+\gamma\Delta a)) - 2\gamma}{2\gamma^2}$$

This whole thing is larger than Δa_f with the assumption from before that $R_a \geq 1$ and if $1-\gamma$ is in a realistic range of under 1%. Under those very common circumstances it is increasing, this shift would be making token b more relatively valuable which means that FR will get more of Δa_{f2} when they trade. Now for the statics with B_{st}, B_{split}. Since I do not have an exact form for Δa^* I can't find them for parameters in it except for on a single trade. Checking the derivative for g we see

$$\frac{\partial B_{st}}{\partial g} = -1, \frac{\partial B_{split}}{\partial g} = -\sum_{t=1}^{T} \beta^{t-1}$$

This shows that as the network transaction fee increases the victim tends towards allowing their trades to be front-run. Checking for the β derivative yields

$$\frac{\partial B_{split}}{\partial \beta} = \sum_{t=1}^{T}(t-1)\beta^{t-2}(R_{b,t}-g) - \frac{\sum_{t=1}^{T}(t-1)\beta^{t-2}(R_{a,t}R_{b,t})\prod_{l\neq t}(R_{a,l}+\gamma\Delta a_l^*)}{\prod_{t=1}^{T}(R_{a,t}+\gamma\Delta a_t^*)}$$
$$\geq \sum_{t=2}^{T}\beta^{t-2}[(R_{b,t}-g) - \frac{(R_{a,t}R_{b,t})\prod_{l\neq t}(R_{a,l}+\gamma\Delta a_l^*)}{\prod_{t=1}^{T}(R_{a,t}+\gamma\Delta a_t^*)}]$$

The second part being multiplied in the brackets is known to be positive if B_{split} is. So then we have that a split becomes better if victims are more patient. If there is discounting the optimal split shouldn't be equal sized splits because further out transactions would be worth less to you .When my environment already ignores reserves changing from future periods from outsiders this discount fact does help price in the user risk preferences. This β should be close to 1 unless the trade is very urgent given that these periods are blocks of transactions processed and this is done in seconds. Now I will check Δa's effect on profits.

$$\frac{\partial \pi}{\partial \Delta a} = \gamma + (\gamma-1)\frac{\partial \Delta a_f}{\partial \Delta a} - \frac{R_a R_b}{(R_b - \Delta b_a + (\gamma-1)\Delta b_f)^2}(\frac{\partial \Delta b_a}{\partial \Delta a} + (1-\gamma)\frac{\partial \Delta b_f}{\partial \Delta a})$$
$$\frac{\partial \Delta b_a}{\partial \Delta a} = \frac{R_a R_b}{(R_a+\gamma\Delta a_f+\gamma\Delta a)^2}(\gamma\frac{\partial \Delta a_f}{\partial \Delta a}+\gamma) - \frac{R_a R_b}{(R_a+\gamma\Delta a_f)^2}(\gamma\frac{\partial \Delta a_f}{\partial \Delta a}) < 0$$
$$\frac{\partial \Delta b_f}{\partial \Delta a} = \frac{R_a R_b}{(R_a+\gamma\Delta a_f)^2}(\gamma\frac{\partial \Delta a_f}{\partial \Delta a}) > 0$$

The derivative with Δb_f and Δb_a results are as they are due to the increasing derivative on Δa_f. This final $(\frac{\partial \Delta b_a}{\partial \Delta a} + (1-\gamma)\frac{\partial \Delta b_f}{\partial \Delta a})$ will be equal to

$$\frac{R_a R_b}{(R_a+\gamma\Delta a_f+\gamma\Delta a)^2}(\gamma\frac{\partial \Delta a_f}{\partial \Delta a}+\gamma) + \frac{R_a R_b \gamma}{(R_a+\gamma\Delta a_f)^2}(\gamma\frac{\partial \Delta a_f}{\partial \Delta a}) > 0$$

This means that the $\frac{\partial \pi}{\partial \Delta a} < 0$ if all of the terms following the first γ which are all negative exceed it, profit declines in Δa. So then since profits and Δa move inversely then when b increases and profits decline then the Δa should rise and then you could weakly reduce the amount of periods required in the trade split and the amount you obtain would increase.

VII Additional considerations

The equilibrium in part V is focusing on a specific equilibria but there may exist other types of equilibria for the victim that were not found. They may opt to trade one or more periods without front-running and they trade the remaining token a in a single period while being front-runned. The victim trade's size may be declining over time because they are impatient and at some point it is preferable to take some frontrunning on a portion rather than wait because the remainder of their budget is too small. If so then an example of the trading structure is as follows. If the victim were to prefer doing 1 period of trades without front-running and then 1 period in which they spend the remainder of their budget and get front-runned to a fully split set of trades then $B_{split,1} \geq B_{split} \geq B_{st}$. β is known to improve splitting and with the new trading there are fewer future payoffs so it likely increases less in β than splitting. Here I use T=3 because the number of roots in this example grows in T, however, if this is done with a parameter like g then there is no such problem. If $B_{split} \geq B_{st}$ then

$$\begin{aligned} B_{split} = R_{b,1} - g + \beta(R_{b,2} - g) + \beta^2(R_{b,3} - g) - \frac{R_{a,1}R_{b,1}(R_{a,2} + \gamma\Delta a_2^*)(R_{a,3} + \gamma\Delta a_3^*)}{(R_{a,1} + \gamma\Delta a_1^*)(R_{a,2} + \gamma\Delta a_2^*)(R_{a,3} + \gamma\Delta a_3^*)} \\ - \frac{\beta R_{a,2}R_{b,2}(R_{a,1} + \gamma\Delta a_1^*)(R_{a,3} + \gamma\Delta a_3^*) + \beta^2 R_{a,3}R_{b,3}(R_{a,1} + \gamma\Delta a_1^*)(R_{a,2} + \gamma\Delta a_2^*)}{(R_{a,1} + \gamma\Delta a_1^*)(R_{a,2} + \gamma\Delta a_2^*)(R_{a,3} + \gamma\Delta a_3^*)} \\ = \sum_{t=1}^{3} \beta^{t-1}(\Delta b_t - g) \geq \frac{R_{a,1}R_{b,1}}{(R_{a,1} + \gamma\Delta a_{f,1})(R_{a,1} + \gamma\Delta a_{f,1} + \gamma\Delta a_1)} \\ \beta \geq \frac{\sqrt{(\Delta b_2 - g)^2 - 4(\Delta b_3 - g)(\Delta b_1 - g - B_{st})} - (\Delta b_2 - g)}{2(\Delta b_3 - g)} \end{aligned}$$

I ignore the negative root since $\beta \in [0, 1]$. Now the deviation after 1 period of safe trading.

$$\begin{aligned} \Delta b_1 - g + \frac{\beta}{R_{a,2}R_{b,2}}(R_{a,2} + \gamma\Delta a_{f,2})(R_{a,2} + \gamma\Delta a_{f,2} + \gamma(A - \Delta a_1) \geq \frac{R_{a,1}R_{b,1}}{(R_{a,1} + \gamma\Delta a_{f,1})(R_{a,1} + \gamma\Delta a_{f,1} + \gamma\Delta a_1)} \\ \beta \geq \frac{B_{st} - (\Delta b_1 - g)}{\frac{R_{a,2}R_{b,2}}{(R_{a,2}+\gamma\Delta a_{f,2})(R_{a,2}+\gamma\Delta a_{f,2}+\gamma(A-\Delta a_1))}} = \frac{[B_{st} - (\Delta b_1 - g)](R_{a,2} + \gamma\Delta a_{f,2})(R_{a,2} + \gamma\Delta a_{f,2} + \gamma(A - \Delta a_1))}{R_{a,2}R_{b,2}} \end{aligned}$$

If β satisfies both of these inequalities then they choose whichever of the two outputs is larger, if only one is satisfied then they would choose that. So while it is more straightforward to determine what is superior between splitting fully and a single trade there may be other equilibria. This problem has more potential options to consider as the number of necessary periods required to fully split grows and computation time used may be important for some traders.

This current model has left slippage to be exogenous but it is a choice to set it however you like when you trade on any exchange. Zhou et al 2020 note that most traders do leave it as the default 1% but they note that among every token pair observed the average slippage was 1.16% so many trades wouldn't execute at that. This is including many pairs so for higher liquidity commonly traded pairs this is lower and for lower liquidity token pairs this is going to be higher. If we want to include slippage as an argument to optimize victims utility we need to include a probability of failed transactions in the model which I had not included. It should be that the probability

of a failed transaction is decreasing in slippage $\frac{\partial P(\text{Failed transaction}|s)}{\partial s} < 0$. The optimization would look like this

$$\max_{\Delta a_t, s}(1-P(\text{Failed}|s))^T \sum_{t=1}^{T} \beta^{t-1}(\Delta b_{a,t} - g) + (1-P(\text{Failed}|s))^T \sum_{t=2}^{T+1} \beta^{t-1}(\Delta b_{a,t} - g)$$
$$+(1-P(\text{Failed}|s))(\Delta b_{a,1} - g) + (1-P(\text{Failed}|s))^{T-1} \sum_{t=3}^{T+2} \beta^{t-1}(\Delta b_{a,t} - g)...$$

The constraints are identical to before. This first term describes every transaction succeeding, the second is if the first is the only one to fail, the third is if the second fails and so on. This will have a first order condition that is not very tractable if T is high but this is doable numerically if you assume away very low probability events. The probability of failure may vary significantly based on market conditions so perhaps an estimated average based on previous data may not always be accurate. Though a frontrunner desires price changes they will have to also balance this with keeping it safely below the cutoff due to unforseen factors. If your β is high and g is low then these failed transactions would likely be less significant since these are the costs of a failed transaction.

A potential algorithm that would use this model would be as follows. First the user would state how long they would be willing to wait for the trades and the size of the trade. Then you calculate the payoff for the single trade and the estimated Δa_t for each period. With the Δa_t you know the number of periods required to complete all of the split trades. You can compare this time to the time requested by the user and if the required time is lower or about equal to the requested then beta should be pretty high. If the requested time falls lower below the required you could bring beta down in some function so that it brings the payoff value of the split trades lower. Then you can estimate parameters like b and D using past transaction data and current reserves of the exchange respectively. The b would be the average observed bid of frontrunners on similar transactions. The D would be found from comparing the exchange reserves to however much reserves the whole market would have if it were in a single exchange. Finally you can compare the payoffs of split trades versus the single trade and then generate a smart contract that would execute the better option. There are now a few options for designing smart contracts that allow for cash flows over time such as what is being done on superfluid.finance or using a verifiable delay function. Some options would force commitment of future transactions and some would allow for it to be canceled at any time before completion. This would likely change the frontrunners strategy if there is precommittment. .

VIII Conclusions

This paper serves as a formal description of the basic sandwich attack front-running game that is experienced by traders on decentralized exchanges and it could be translated into a fairly simple algorithm and some of the comparative statics could serve as potential guidelines. Even after coming back to further improve the paper years later, the literature still appears to have a place for this and the MEV problem is still unsolved and only marginally reducible by educated users only. The literature has focused much more on protocol or blockchain changes and this area of user mitigated MEV likely has potential since frontrunning is strategic behavior. This could serve as a framework or inspire a better framework for considering potential changes to reduce the phenomenon. There are other rarer potential front-running attacks to consider such as the liquidity provider version of this same sandwich attack described in Zhou et al. 2020 in which the market maker is doing the sandwich attack and after they purchase Δb_f they remove their reserves from the exchange so that the token is less liquid and there is a greater price impact from the victim's trades. Fair ordering rules, batches, or privacy attempts havent gotten to a point where its very largely used and its not clear all of the compromises are known and accounted for and with it comes user distrust. Incoming exchanges like Nasdaq or London stock exchange needing to consider all of this papers considerations are depending on how they implement their solution. If their network contains decentralized validators then there are room for attacks on the network to disrupt consensus and MEV. As a cautionary tale, one of the most popular

blockchains Solana attempted and later had to change its attempt at fair ordering. All of these could be potentially interesting extensions for further exploring this growing problem.

References

Adams, A., Chan, B. Y., Markovich, S., & Wan, X. (2023). Don't let MEV slip: The costs of swapping on the Uniswap protocol. *arXiv preprint arXiv:2309.13648*.

G. Angeris, H.-T. Kao, R. Chiang, C. Noyes, and T. Chitra, "An analysis of Uniswap markets," Cryptoeconomic Systems, to appear.

G. Angeris, A. Evans, and T. Chitra. A note on privacy in constant function market makers. arXiv preprint arXiv:2103.01193, 2021.

Breidenbach, L., Daian, P., Tramer, F., & Juels, A. (2022). MEV-SGX: A secure and private execution environment for decentralized finance. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*.

Brunnermeier, Markus K.,Lasse Heje Pedersen. 2005. "Predatory Trading." Journal of Finance, 60(4): 1825– 63.

Philip Daian, Steven Goldfeder, Tyler Kell, Yunqi Li, Xueyuan Zhao, Iddo Bentov, Lorenz Breidenbach, and Ari Juels. Flash Boys 2.0: Frontrunning, Transaction Reordering, and Consensus Instability in Decentralized Exchanges. arXiv preprint arXiv:1904.05234, 2019.

Kelkar, M., Deb, S., Long, S., Juels, A., & Kannan, S. (2022). Themis: Fast, strong order-fairness in Byzantine consensus. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*.

Obadia, A., Harz, D., Fritsch, L., & Gudgeon, L. (2022). An analysis of CowSwap's batch auctions. *arXiv preprint arXiv:2206.04128*.

Zhou, L., Qin, K., Torres, C.F., Le, D.V., Gervais, A.: High-frequency trading on decentralized on-chain exchanges. arXiv preprint arXiv:2009.14021 (2020)