Towards a Theory of Maximal Extractable Value I: Constant Function Market Makers

Kshitij Kulkarni
ksk@eecs.berkeley.edu

Theo Diamandis
tdiamand@mit.edu

Tarun Chitra
tarun@gauntlet.network

July 2022

Abstract

Maximal Extractable Value (MEV) represents excess value captured by miners (or validators) from users in a cryptocurrency network. This excess value often comes from reordering users transactions to maximize fees or inserting new transactions that allow a miner to front-run users' transactions. The most common type of MEV involves what is known as a sandwich attack against a user trading on a popular class of automated market makers known as CFMMs. In this first paper of a series on MEV, we analyze game theoretic properties of MEV in CFMMs that we call *reordering* and *routing* MEV. In the case of reordering, we show conditions when the maximum price impact caused by the reordering of sandwich attacks in a sequence of trades relative to the average price impact is $O(\log n)$ in the number of user trades. In the case of routing, we present examples where the existence of MEV both degrades and counterintuitively *improves* the quality of routing. We construct an analogue of the price of anarchy for this setting and demonstrate that if the impact of a sandwich attack is localized in a suitable sense, then the price of anarchy is constant. Combined, our results provide improvements that both MEV searchers and CFMM designers can utilize for estimating costs and profits of MEV.

Contents

1 Introduction

Blockchain systems, such as Bitcoin and Ethereum, allow users to submit financial transactions to a network of validators (often referred to as miners) who compete to earn fees from processing transactions. Validators aggregate users transactions into blocks of transactions and then come to consensus on whether the transactions in a block are valid according to a set of consensus rules. Cryptographically secure consensus protocols from distributed systems are used to ensure that the probability that a miner can profitably execute a strategy to earn a higher share of fees than the resources they lock into the network is negligible.

However, the majority of consensus protocols do not enforce any constraints on the precise ordering of transactions within a block. One reason for this is because it is effectively impossible to ensure that validators provide precise enough and non-manipulable timestamps to allow for agreement on a global first-in-first-out (FIFO) ordering. This means that individual validators or cartels of validators can agree upon a particular ordering of transactions that nets them the highest profit. One way of extracting profit is front-running: a miner observes a user's trade and submits their own transaction in front of the user's trade to force the user to have a worse execution price. Any type of excess profit that a miner can extract by adjusting execution of user transactions is known as Maximal Extractable Value (MEV).[1]

There are three principal agents involved in MEV: miners, network users, and MEV searchers. Miners (or validators in Proof of Stake networks) contribute resources to a network in order to win the chance to earn fees by validating transactions. Network users are ordinary users who submit financial transactions to miners to be validated and added to the blockchain. Finally, MEV searchers (or simply, searchers) are agents who find profitable opportunities from reordering, inserting, or eliding transactions.

Searchers design *strategies*, which are solutions to Knapsack-like problems that find the sequence of transactions that is most profitable yet fits within the block limit (*e.g.* maximum number of transactions per block). Once a searcher finds a profitable strategy, they enter an auction run by miners that allows them to bid on particular transaction slots. For instance, if a strategy involves front-running a particular transaction that is found in the public memory pool of transactions, a searcher can bid to be placed right behind this transaction. One might expect that the market would converge to an equilibrium where miners and searchers are the same agent. However, this has not borne out to be true as services such as Flashbots [Teaa] run the combinatorial auction for searchers who are not necessarily miners to bid on particular block positions. Participants in these auctions share their profits with the miner who guarantees the transactions they request are executed atomically and at the correct positions.

This paper is the first of a series that aims to present a number of different theoretical lenses that can be used for analyzing MEV. While previous work has focused on the computational description of MEV, all work known to the authors does not discuss the economic

[1] We note that attacks that incentivize unnecessary forking in blockchains, sometimes referred to as 'time-bandit attacks', are also a form of MEV [JSZ+21]. In this paper, we ignore these types of attacks but note that our framework can be generalized to include them.

properties of systems with MEV. By viewing MEV as a multi-agent game between miners, searchers, and users, one can attempt to compare the equilibria that occur between different forms of MEV.

We use this simple game theoretic observation to compare MEV in a number of applications including trading and lending and provide explicit examples of user utilities and payoffs. This series will conclude with a description of how MEV between different applications composes and how one can qualitatively understand such equilibria (which we suspect are computationally difficult, *e.g.* PPAD-complete, to compute directly).

Prior work on MEV. Since the first work that unearthed MEV was published in 2019 [DGK+19], there has been over $650 million extracted [Teaa]. Moreover the number of different types of miner strategies used to exploit the lack of transaction ordering safety has grown rapidly [QZLG20, ZQG21, QZG21, AEC21, BCLL21a, BCLL21b]. There has been a desire to understand the space of possible attacks and to quantify their profitability under different conditions. There have been a number of papers that try to quantify sandwich attack profitability but only for constant product market makers [BCLL21a, HW22]. However, they do not provide minimax, Price of Anarchy, or worst-case bounds for generic CFMMs and bundles or sequences of transactions. Recent work has focused on reducing strategy profitability via more complex ordering consensus mechanisms (so-called 'fair sequencing' [KZGJ20]) and by reducing the dimensionality of the strategy space using formal verification [BDKJ21].

However, none of this work provides theoretical guarantees or bounds on the profitability of MEV. Fair sequencing tries to reduce MEV by having validators come to consensus on relative transaction orderings — *e.g.* validators vote on whether transaction A came before transaction B as part of the consensus protocol rules. Such systems, as the authors of [KZGJ20] readily admit, cannot ever be deterministically secure due to the Condorcet paradox and Arrow's impossibility theorem. The same authors try to rectify this in [KDL+21], however, it requires a permissioned blockchain network which is far from the the realistic setting of blockchains like Ethereum and Solana. Moreover, both of their frameworks are application agnostic. Given that the profitability of MEV strategies varies dramatically from application to application [Teaa], this implies that the framework doesn't provide tight or strong bounds on how much it reduces MEV.

Subsequent work [BDKJ21] aimed to correct this deficiency by establishing a framework for numerically evaluating profitability. The tools utilized in this work involve using formal verification to search through the set of sequences of actions that satisfy a predicate (*e.g.* "strategy generates >$X of profits"). While this work is practically useful, it provides no theoretical insight into why certain applications have more or less MEV than others. Moreover, no prior provides any guidance on the types of economic equilibria that can occur in systems with MEV.

MEV and Privacy. One of the goals of this series of works is to provide formal, theoretical tools for analyzing MEV in a variety of different contexts. The focus on this paper will be on

MEV that occurs when users makes transactions with a particular type of automated market maker. However, any theoretical framework for MEV needs to be generic enough to describe MEV when applied to automated market makers [QZG21], lending protocols [SS21], non-fungible tokens [AH21], and more. Moreover, any theoretical foundation needs to account for the transaction level differences between forms of MEV. For instance, simple transactions, such as front running or sandwiching (§2.3), are 'local' in that they generally only impact a constant number of transactions (relative to the blocksize). On the other hand, complex flash loan attacks or block purchasing attacks [McS22] can be extremely non-local in nature.

Prior work [CAE22] has implicitly connected MEV profits in automated market makers to a loss of user privacy. A miner's ability to extract maximal value from a user (or group of users) stems from their ability to see all transactions publicly *before* committing to a validate a block. This allows the validator to simulate different strategies before committing to validating, as exemplified by Flashbot's transaction simulator [Teaa]. As such, we can view a blockchain user's utility function for a particular transaction as consisting of a positive component (*e.g.* price on an exchange is lower than the user's private valuation) and a negative component due to losses from privacy. Thus, the maximum loss to a user from lack of privacy can be viewed as a major factor in the calculation of the maximum profit that a validator can extract from their public transactions.

Privacy-utility trade-offs [MF13, SCDFSM17, AAC$^+$11, SRP13, DJW13, DJW14] are commonly studied in differential privacy and federated machine learning. They represent the idea that as a user begins to privatize their data (*e.g.* by adding noise to their inputs), predictive models trained on this data suffer from a loss in quality. A model's utility to its users generally goes down as user privacy is increased. The goal of analyzing such trade-offs is to show that the social welfare (*e.g.* the utility realized by all users) is optimally traded off for individual privacy.

In this work, we implicitly study a privacy-utility trade-off for users of CFMMs. Their utility is measured by the price impact a trade of size Δ causes — the less price impact, the higher the user's utility. On the other hand, the ability for an MEV searcher to extract a profit of size $\mathsf{PNL}(\Delta)$ from a user trade represents a measurement of their loss of privacy. Our results in §3 and §4 utilize this privacy-utility decomposition to quantify the trade-off under different liquidity conditions. This can also be viewed as extending the work of [CAE22] to include the impact of MEV on the privacy-utility trade-off.

MEV in CFMMs. The privacy-utility trade-off that arises from analyzing differential privacy in constant function market makers (CFMMs), like Uniswap, provides an explicit example of where MEV and privacy are related [CAE22]. In [CAE22] scenario, the authors capture the trade-off between the cost of privacy (*e.g.* splitting trades has a cost for the trader) and the level of privacy, measured in terms of (ϵ, δ)-differential privacy. The paper explicitly shows that splitting a trade Δ into two trades Δ_1, Δ_2 with $\Delta_1 + \Delta_2 = \Delta$, when combined with shuffling the order of transaction in a block, leads to better privacy for users while worsening their price impact. The (ϵ, δ)-differential privacy guarantee in the paper implies that splitting trades and reordering them reduces the worst-case profit from MEV

exponentially in the number of trades executed. On the other hand, earlier work [AC20] demonstrated that splitting trades and executing them on a CFMM leads to worse utility (*e.g.* the user pays a higher price for the same quantity of asset). Combining these two results demonstrates that CFMMs trade-off privacy (*e.g.* a reduction of worst-case MEV profit) for utility (*e.g.* best price execution).

The results of [CAE22] are qualitative in nature and do not provide explicit formulas for the privacy-utility trade-off. In §2, we analyze 'sandwich attacks', a highly localized form of MEV. Sandwich attacks are by far the most popular type of MEV, with over $500m extracted from users via sandwich attacks [Teaa]. We demonstrate that provided there is enough liquidity, the maximal profit attainable from a sandwich attack has a particular subadditivity property, akin to the subadditivity for privacy found in [CAE22]. This explicit calculation shows that applying the privacy methodology of [CAE22] reduces sandwich attacks and explicitly shows that MEV and privacy in CFMMs are inversely related (*e.g.* lower MEV leads to higher statistical privacy and vice-versa).

Prior work has focused on analyzing sandwich attacks only in one type of CFMM, the constant product automated market maker Uniswap [HW22, Züs21]. In §2, we generalize this to *any* CFMM as defined in [AAE$^+$21]. To do so, we first define a sandwich attack in terms of the price impact function of a CFMM. This allows us to utilize the notion of CFMM curvature [ACE22] to explicitly provide bounds on the profitability of a sandwich attack given a particular user trade. Unlike prior work [HW22, BCLL21a], we also consider MEV profitability for a searcher that receives a sequence of trades $\Delta_1, \ldots, \Delta_n$ rather than a single trade Δ. Finally, we note that MEV in CFMMs has implicitly been analyzed when studying privacy preserving mechanisms in CFMMs, including threshold cryptography [AO21, BO22], differential privacy [CAE22], and zero knowledge commitments [dV21].

Main Results. We ask two questions pertaining to CFMM MEV that shed light on the maximum value that a MEV searcher who sandwiches user trades can hope to extract:

In the case of a single CFMM trading two assets, how much does ***reordering*** *user trades affect the excess price impact caused by sandwich attacks?*

In the case of a network of CFMMs trading multiple assets, how much does the presence of sandwich attackers on the network affect the ***routing*** *of trades?*

To answer the first question, we construct an analogue of a competitive ratio or so-called *prophet inequality* [Hil83] that measures the ratio of the price impact of the worst case sandwich attack to that of the average sandwich attack given an order flow of n trades, $\Delta_1, \ldots, \Delta_n$. In §3, we show that under sufficient liquidity conditions this ratio, which we term the *cost of feudalism*, is $O(\log n)$. This somewhat counterintuitively suggests that provided that the CFMMs involved have sufficient liquidity, there is not a large asymptotic difference between the worst sandwich attack and the average case sandwich attack. We note that our liquidity constraints are a measurement of 'locality' of a sandwich attack, which ensures that the compounding price impact of a sequence of sandwiches is bounded sufficiently. We also

note that when this locality condition is satisfied, the constrained version of the sandwich problem that takes into account block size[2] is a generalized knapsack problem, whereas when it is not the problem is significantly harder than knapsack. This locality result generalizes the results of [BCLL21a], which only considers optimal sandwiches for Uniswap in the arbitrarily large block size limit.

We answer the second question about routing by adapting conventional Price of Anarchy (PoA) results [Rou05, Rou15, RST17] to CFMMs. Our setup is the following: suppose that we have a graph $G = (V, E)$, where V is a set of tokens and $E \subset \binom{V}{2}$ is a set of edges. Each edge $e \in E$ has an associated CFMM price impact function $g_e(\Delta)$ that returns the price when a trade of size Δ passes through e. Given an input trade of size $\Delta_{a,b}$ which is in units of token $a \in V$, how should we route our trade across G to maximize the output tokens b that we receive? Recent work [AECB22] has described how to solve this problem without transaction fees or sandwich attacks.

In §4, we define how to generalize the routing problem to include sandwich attacks. The presence of sandwich attacks shifts the notion of an optimal route between two tokens as the problem loses certain convexity properties. This leads to more complicated routing, where a trade can be split over multiple paths between the source and destination nodes. Unlike [AECB22], we instead define a notion of an equilibrium (§4, Def. 6) which states that in equilibrium, paths that have non-zero flow cause the same price impact. Using this notion of equilibria, we are able to construct a PoA for routing flows and utilize the (λ, μ)-smoothness results of [Rou15] to bound the PoA by a constant.

Counterintuitively, we find that the PoA for routing a trade Δ across a network graph of CFMMs is $O(1)$. This suggests the presence of sandwich attacks does not greatly impact routing quality in networks of CFMMs. Another interpretation of this result is that while individual trades that are sandwiched undoubtedly receive worse prices, sandwiches can cause more effective routing patterns as some flow avoids sandwiches edges. In §4 we provide an explicit example of a CFMM network that avoids Braess Paradox [Rou05] behavior because sandwiching makes the Braess edge more expensive. This suggests that sandwiching can sometimes a net benefit to routing quality, which is formally represented by via the constant PoA.

2 Sandwich Attacks

In front-running trades (colloquially termed *sandwich attacks* [QZLG20]), an adversary places orders before and after a user's order to result in a worse execution price for said order. Users specify both trade side and a limit price (in the form of a *slippage limit*) when placing an order. The slippage limit prevents the order from being executed at a price that is much worse than the current market price. After seeing the user's order, an adversary can submit a trade before the user's trade, pushing up the price to any value below the

[2]The constrained problem can be stated as follows: Suppose that we have a block that fits at most k trades. We want to find the subset $A = \{a_1, \ldots, a_k\} \subset [n], |A| = k$ of $\Delta_1, \ldots, \Delta_n$ and a permutation $\pi \in S_k$ such that the block containing $(a_{\pi(1)}, \ldots, a_{\pi(k)})$ has the highest profit

limit. Then immediately after the user's trade is executed, the adversary places a trade in the opposite direction to recover their initial investment and a profit due to the price impact caused by the user's trade. We will first describe sandwich attacks concretely for the most widely-used CFMM, Uniswap before describing their characteristics for general CFMMs.

2.1 Uniswap

Before considering generic CFMMs, we will first illustrate our results for Uniswap [AC20, AKC$^+$21, ZCP18]. Uniswap was the first CFMM to launch in production has had over 1 trillion dollars of trading volume flow through it since inception [@ha22]. It has a particularly simple structure: assume we have reserves of token A, R_A, and reserves of token B, R_B, and without loss of generality, that token B is the numeraire. Then, a user's trade of size Δ (by providing Δ units of token B for token A, or analogously, the CFMM or liquidity provider is swapping A for B) is valid if:

$$(R_A - \Delta')(R_B + \gamma\Delta) = R_A R_B \tag{1}$$

where $1-\gamma$ represents the percentage fee parameter that controls how much the liquidity provider charges for facilitating the trade. The amount Δ' is determined implicitly by (1) and will vary as the fee γ is changed. The price quoted by Uniswap is the ratio of the reserves, *e.g.* the price of A in terms of B is $p_{AB} = \frac{R_A}{R_B}$. For a trade of size Δ, we will define $p_{AB}(\Delta, R_A, R_B) = \frac{R_A - \Delta'}{R_B + \gamma\Delta}$. This is the *marginal price* of trade Δ. That is, by changing the reserves, the trade changes the price of the tokens. The amount of output token that the user receives can then be computed using the marginal price as [AKC$^+$21]:

$$G(\Delta) = \frac{1}{\gamma}\left(\frac{-R_A R_B}{R_A + \Delta} + R_B\right) \tag{2}$$

Note that when $\gamma = 1$, the quantity $k = R_A R_B$ is always constant. In the sequel, we will assume that $\gamma = 1$ but note that lower bounds from [ACE22, App. B] can be used to generalize the results of the paper to $\gamma < 1$.

Slippage Limits. A *slippage limit* $\eta \in [0, 1]$ represents how much of a price impact a user is willing to tolerate to execute their trade, as measured by the minimum amount of the output token they are willing to receive. One reason that a user has to provide a slippage limit is because they may not a priori know the reserves present. For instance, suppose there are two trades Δ_1, Δ_2 to be executed by two different users. Since miners or validators get to choose whether they execute Δ_1 first or second, the user does not know if their trade is executed with an initial price $p_{AB}(0, R_A, R_B)$ or at $p_{AB}(\Delta_2, R_A, R_B)$. The slippage limit η is a way for a user to say that they do not want to receive less than $1-\eta$ times the nominal amount they would receive in the absence of the other trade, *e.g.* if the trade is executed in the sequence Δ_2, Δ_1, the miner cannot execute the trade unless

$$G(\Delta_1 + \Delta_2) - G(\Delta_2) \geq (1-\eta)G(\Delta_1)$$

This condition is enforced by the Uniswap smart contract and allows users to ensure that their trade is executed at a favorable price.[3] However, this still places the onus of choosing the correct parameter η on the user. We define a *trade* to be a pair (Δ, η) of a trade size and slippage limit.

What does it mean for a user to choose a slippage limit that is too large? Explicitly, this means that there is a trade Δ' executed before (Δ, η) such that $G(\Delta + \Delta') - G(\Delta') > (1 + \eta)G(\Delta)$ is a strict inequality and not an equality. This allows for an 'attacker' (MEV searcher) to construct a trade Δ^{sand} such that

$$G(\Delta + \Delta^{\text{sand}}) - G(\Delta^{\text{sand}}) = (1 + \eta)G(\Delta)$$

By filling up the slack in the inequality constraint, the attacker is able to worsen the execution price of the user trade (Δ, η). Moreover, if the attacker submits a trade of size $\Delta^{\text{sand}'}$ after executing the trades Δ^{sand} and Δ, then they are able to profit given the convexity of the Uniswap invariant (see, e.g., [ACE22] and the example below). The triple of trades $(\Delta^{\text{sand}}, (\Delta, \eta), \Delta^{\text{sand}'})$ is called a *sandwich attack*.

Example of Sandwich Attack on Uniswap. We provide here a concrete example of sandwich attacks in the case of Uniswap, whose forward exchange function takes the form:

$$G(\Delta) = -\frac{k}{R + \Delta} + R' \tag{3}$$

for reserves R and R' of input and output asset, respectively, and $k = RR'$ [AKC+21]. Assume the user submits a trade (Δ, η) to this market maker, and a sandwich attacker wants to design Δ^{sand} to force the slippage limit to be tight. That is,

$$G(\Delta + \Delta^{\text{sand}}) - G(\Delta^{\text{sand}}) = (1 - \eta)G(\Delta)$$

and plugging in the functional form of $G(\cdot)$ for Uniswap, we have:

$$-\frac{k}{R + \Delta + \Delta^{\text{sand}}} + R' + \frac{k}{R + \Delta^{\text{sand}}} - R' = (1 - \eta)\left(-\frac{k}{R + \Delta} + R'\right)$$

Solving for Δ^{sand} (with the full calculation in Appendix A), we have:

$$\Delta^{\text{sand}} = \frac{-(\Delta + 2R) + \sqrt{(\Delta + 2R)^2 - 4(R^2 + R\Delta)\frac{-\eta}{1-\eta}}}{2} \tag{4}$$

Note that when $\eta = 0$, the above simplifies to $\Delta^{\text{sand}} = 0$, as desired, and that Δ^{sand} is an increasing function of η.

[3] The condition enforced by the Uniswap contract is a *minimum amount* of the output token that a user is willing to accept [Uni22]. For reasons of convenience, we choose to work in quantity space, but detail how quantity slippage limits can be converted into price slippage limits in Appendix B.

2.2 Constant function market makers

We now generalize sandwich attacks to more general contracts called *constant function market makers*, which hold some amount of *reserves* $R, R' \geq 0$ of two assets and have a *trading function* $\psi : \mathbf{R}^2 \times \mathbf{R}^2 \to \mathbf{R}$. Traders can then submit a *trade* (Δ, Δ') denoting the amount they wish to tender (if negative) or receive (if positive) from the contract. The contract then accepts the trade if

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

and pays out (Δ, Δ') to the trader.

Curvature. We briefly summarize the main definitions and results of [ACE22] here. Suppose that the trading function ψ is differentiable (as most trading functions in practice are), then the marginal price for a trade of size Δ is

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

Here ∂_i denotes the partial derivative with respect to the ith argument, while Δ' is specified by the implicit condition $\psi(R, R', \Delta, \Delta') = \psi(R, R', 0, 0)$; *i.e.*, the trade (Δ, Δ') is assumed to be valid. Additionally, the reserves R, R' are assumed to be fixed. Matching the notation of Section 2.1, the function g is known as the *price impact* function as it represents the final marginal price of a positive sized trade. When there are fees, one can show that $g^{fee}(\Delta) = \gamma g(\gamma \Delta)$ where $1 - \gamma$ denotes the percentage fee. We say that a CFMM is α-*stable* if it satisfies

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

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

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

for all $\Delta \in [0, K]$ for some positive K. One important property of g is that it can be used to compute Δ' [ACE22, §2.1]:

$$\Delta' = \int_0^{-\Delta} g(t) dt \tag{5}$$

Simple methods for computing α and β in common CFMMs are presented in [ACE22, §1.1] and [AAE$^+$21, §4]. We define $\Delta' = G(\Delta)$ to be the *forward exchange function*, which is equivalently the amount of output token received for an input of size Δ. The function $G(\Delta)$ was shown to be concave and increasing in [AAE$^+$21].

Slippage Limits. When a user submits an order to a CFMM, they submit two parameters: a trade size $\Delta \in \mathbf{R}$ and a *slippage limit* $\eta \in (0, 1)$. The slippage is interpreted as the minimum output amount that the user is willing to accept as a fraction of $G(\Delta)$. That is, the trade is accepted if the amount in output token the user receives is larger than or equal to

$(1-\eta)G(\Delta)$. This notion of slippage can also be mapped to the maximum price impact that a user is willing to tolerate, as a percentage, which we elucidate in Appendix B. However, the majority of this paper will focus on representing slippage limits in quantity space (e.g. in terms of $G(\Delta)$ not $g(\Delta)$).

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

$$|g(\Delta) - g(0)| \leq \alpha''|\Delta|,$$

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

$$|g(\Delta) - g(0)| \geq \beta''|\Delta|.$$

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

Note that any two-sided α-stable and β-liquid market maker is automatically η-stable and η-liquid for $\eta = \min(\alpha, \beta)$. An η-liquid and η-stable price impact function is *bi-Lipschitz* and admits an inverse $g^{-1}(p)$ that is also bi-Lipschitz [How97]. In particular, if g is η bi-Lipschitz, then $g^{-1}(p)$ is $\frac{1}{\eta}$ bi-Lipschitz:

$$\frac{1}{\alpha}p \leq |g^{-1}(p) - g^{-1}(0)| \leq \frac{1}{\beta}p$$

2.3 Sandwich Attacks

Sandwich attacks have been analyzed for constant product CFMMs (*e.g.* Uniswap) [ZQT+21, QZLG20], but have not been analyzed for general CFMMs. We will generalize prior work to CFMMs with two-sided price impact functions $g(\Delta)$. Our results will initially start in the path independent scenario (*e.g.* $\gamma = 1$) but we can use bounds from [ACE22, App. B] to generalize to the scenario with fees.

Recall that a user submits an order to a CFMM of the form $T = (\Delta, \eta) \in \mathbf{R} \times \mathbf{R}_+$, where η is the slippage limit. The slippage limit quantifies the minimum amount of output that the user is willing to receive from the CFMM. That is, the user is willing to receive no less than $(1 - \eta)G(\Delta)$ units of the output token. If a user submits an order that is not tight, then there exists a Δ^{sand} such that $G(\Delta + \Delta^{\text{sand}}) - G(\Delta^{\text{sand}}) > (1 - \eta)G(\Delta)$. That is, Δ^{sand} satisfies:

$$G(\Delta + \Delta^{\text{sand}}) - G(\Delta^{\text{sand}}) = (1 - \eta)G(\Delta) \tag{6}$$

Table 1: Sequence of reserve updates in a sandwich attack.

	Input Reserves	Output Reserves
Sandwich attack	$R \to R + \Delta^{\text{sand}}$	$R' \to R' - \Delta^{\mathsf{sand,out}}$
User submits trade	$R \to R + \Delta^{\text{sand}} + \Delta$	$R' \to R' - \Delta^{\mathsf{sand,out}} - \Delta^{\mathsf{out}}$
Sandwicher sells back	$R \to R + \Delta^{\text{sand}} + \Delta - \Delta^{\text{sand}'}$	$R \to R - \Delta^{\mathsf{out}}$

To see where this equation comes from, we enumerate the trade sequence of a sandwich attack in Table 1. Suppose initially that the CFMM has reserves R and R'. The sandwicher submits Δ^{sand} ahead of the user, which causes the reserves to be updated as $R \to R + \Delta^{\text{sand}}$ and $R' \to R' - \Delta^{\mathsf{sand,out}}$. Now, the user submits the trade Δ, after which the reserves are $R \to R + \Delta^{\text{sand}} + \Delta$ and $R \to R - \Delta^{\mathsf{sand,out}} - \Delta^{\mathsf{out}}$. Recall that the user is interested in receiving no less than $(1-\eta)G(\Delta)$ units of the output token. The amount the user receives after sandwiching, Δ^{out}, is given by seeing that $\Delta^{\mathsf{sand,out}} + \Delta^{\mathsf{out}} = G(\Delta^{\text{sand}} + \Delta)$, and from the reserve update equation for the sandwicher's trade, $\Delta^{\mathsf{sand,out}} = G(\Delta^{\text{sand}})$. Substituting, this allows us to solve for Δ^{out}, giving:

$$\Delta^{\mathsf{out}} = G(\Delta^{\text{sand}} + \Delta) - G(\Delta^{\text{sand}})$$

Setting the amount that the user gets out, Δ^{out}, equal to the minimum amount they would be willing to receive, $(1-\eta)G(\Delta)$, we have the defining equation for optimal sandwiches, Equation (6).

After Δ^{sand} and T are executed, the sandwich attacker sends a trade of $\Delta^{\text{sand}'}$ to recover their initial investment of Δ^{sand}. After sending the initial trade of Δ^{sand}, the sandwich attacker holds $G(\Delta^{\text{sand}})$ of output token. If the attacker is to stay risk neutral they have to sell back the amount of output token they hold, $G(\Delta^{\text{sand}})$. This amount can be calculated as follows in units of input token:

$$\Delta^{\text{sand}'} = \Delta^{\text{sand}} + \Delta - G^{-1}(G(\Delta + \Delta^{\text{sand}}) - G(\Delta^{\text{sand}})) \tag{7}$$

where $G^{-1}(\cdot)$ is the reverse exchange function, the inverse of G [AAE+21].

Therefore, the complete sandwich attack is a triplet of transactions: $\Delta^{\text{sand}}, (\Delta, \eta), \Delta^{\text{sand}'}$. We emphasize that both Δ^{sand} and $\Delta^{\text{sand}'}$ are in units of *input* token. If the sandwich attack is executed, the sandwicher can make a profit of:

$$\begin{aligned}\mathsf{PNL}(\Delta, \eta) &= \Delta^{\text{sand}'}(\Delta, \eta) - \Delta^{\text{sand}}(\Delta, \eta) \\ &= \Delta - G^{-1}(G(\Delta + \Delta^{\text{sand}}) - G(\Delta^{\text{sand}}))\end{aligned} \tag{8}$$

measured in input token, where we use the notation $\Delta^{\text{sand}}(\Delta, \eta)$ to refer to the solution of the implicit equation (6) and $\Delta^{\text{sand}'}(\Delta, \eta)$ refers to the quantity defined in (7). Note that when $\eta = 0$, $\mathsf{PNL}(\Delta, \eta) = 0$ for all Δ, as desired.

Note that this is the net profit and loss (PNL) because the first term of the right-hand side measures the amount of input-token received after the round-trip trades, whereas the second term subtracts the amount of input token that the sandwicher had to put up as capital to execute the attack. One can use the equation $G(\Delta + \Delta^{\text{sand}}) - G(\Delta^{\text{sand}}) = (1-\eta)G(\Delta)$ to numerically solve for the optimal $\Delta^{\text{sand}}(\Delta, \eta)$, *e.g.* by finding the roots of $G(\Delta+x) - G(x) - (1-\eta)G(\Delta) = 0$. Going forward, we abuse notation and simply refer to this solution as Δ^{sand}.

2.4 Bounds on Sandwich Attack Profitability.

In order to more holistically reason about the impact of sandwich attacks and to construct a social welfare function, we first need to reason about the expected size of a sandwich attack ${\Delta^{\text{sand}}}_i$ given a sequences of trades $\Delta_1, \ldots, \Delta_n$. We first show upper and lower bounds on the sandwich trade size as a function of curvature parameters and slippage limits. Subsequently, we show that sandwich attack profitability is often maximized by sandwiching each trade Δ_i independently. This represents the 'locality' of sandwich attacking — one doesn't need to combine subsets of trades to sandwich together, reducing the computational complexity and allowing us to bound the net price impact of sandwich trades. To construct these bounds, we first need one definition about the rate of growth of $G(\Delta)$:

Definition 1. A forward exchange function $G(\Delta)$ is (μ, κ)-*smooth* if there exists $M > 0$ such that for all $\Delta \in [0, M]$ there exist constants $\mu, \kappa > 0$ such that

$$\kappa\Delta \leq G(\Delta) - G(0) \leq \mu\Delta \tag{9}$$

Usually, one takes $G(0) = 0$ so this corresponds to a set of Bilipschitz bounds on G. Note that we can define an analogous notion of smoothness for the reverse exchange function (and bounds for $\Delta < 0$), but for simplicity, we will phrase all of our results in terms of the forward exchange functions as all of the proofs hold for reverse exchange functions.

Note 1. *Note that μ and κ in* (9) *are distinct from the definition of α-stability and β-liquidity in Section 2.2.*

Bounds on Δ^{sand}. We first provide bounds on Δ^{sand}, whose proofs will be in Appendix C.

Claim 1. *If $\eta \geq 1 - \frac{\kappa}{\mu}$ then we have $\Delta^{\text{sand}}(\eta, \Delta) = O(\eta)\Delta$*

This bound demonstrates that the size of a sandwich attack is linear in the slippage limit provided that the slippage limit is sufficiently larger than a curvature ratio. Such a bound can be used, for instance, by wallet designers to help users choose slippage limits that explicitly bound the maximum expected sandwich profit. For lower bounds on Δ^{sand}, we will need to make further assumptions. In particular, we require that the price impact function $g(\Delta) = G'(\Delta)$ grows sufficiently fast.

Claim 2. *Suppose that the price impact function $g(\Delta)$ is β-liquid in addition to G being (μ, κ)-smooth. Then there exists $\gamma = 1 + \Theta(\sqrt{1+\eta})$ such that*

$$\Delta^{\text{sand}} \geq \left(\frac{\mu}{\beta} - \Delta\right)\gamma$$

The intuition for why we need this extra assumption can come from analyzing a constant sum market maker which has $\beta = 0$ [AAE+21]. In a constant sum market maker, there is no sandwich profit as there is no price impact when one executes the sequence of trades $(\Delta^{\text{sand}}, \Delta, \Delta^{\text{sand}'})$. Therefore, in order for us to have any lower bound on sandwich size (and profit, which is linear in Δ^{sand} as per 8), we need some non-zero price impact. This is precisely captured by the β-liquid condition for price impact.

Bounds on $\Delta^{\text{sand}'}$. Now, we bound the round trip trade made by the sandwicher, $\Delta^{\text{sand}'}$, which satisfies equation (7) (note that $\Delta^{\text{sand}'}$ is in units of input token). We prove the following two claims in Appendix D:

Claim 3. *Suppose that $\eta \geq \frac{2-\frac{\kappa}{\mu}}{1+\frac{\mu}{\mu-\kappa}}$. Then $\Delta^{\text{sand}'} = O(\eta)\Delta$*

This claim demonstrates that under mild conditions on the slippage limit, we can control the roundtrip profit in terms of linear factors of the slippage limit.

Claim 4. *Suppose that $g(\Delta)$ is β-liquid. Then there exists $\gamma = 1 + \Theta(\sqrt{1+\eta})$ such that*

$$\Delta^{\text{sand}'} \geq \frac{\mu\gamma}{\beta} - \Delta\left(\gamma + \frac{\eta\mu}{\kappa}\right)$$

Upper Bound on Sandwicher Price Impact and Profit. Recall that the net price impact of a sandwich is controlled by $\Delta^{\text{sand}} + \Delta - \Delta^{\text{sand}'}$. To upper bound price impact, we first need to lower bound $\Delta^{\text{sand}'}$. Using Claims 1 and 4 we can now bound the total trade size that occurs in the input token when sandwiched (the *effective* size of the sandwich attack):

$$\Delta^{\text{sand}} + \Delta - \Delta^{\text{sand}'} \leq (O(\eta) + \gamma)\Delta - \frac{\mu\gamma}{\beta}$$

Similarly, this gives us a bound on the profit (8):

$$\mathsf{PNL}(\Delta, \eta) = \Delta^{\text{sand}'} - \Delta^{\text{sand}} \leq (O(\eta) + \gamma - 1)\,\Delta - \frac{\mu\gamma}{\beta} \leq C\max(\eta, \sqrt{1+\eta})\Delta - \frac{\mu\gamma}{\beta}$$

This implies that given all of the liquidity and slippage conditions of the claims are met, profit and price impact are linear in γ. Using the precise constants in Appendices C and D, one can also compute a 'hurdle rate' in terms of γ that describes minimal conditions for a sandwicher to be profitable (proved in Appendix D):

Claim 5. *If $\left(\eta\left((1+\frac{\mu}{\kappa}) - \left(2 - \frac{\kappa}{\mu}\right) + \gamma\right)\Delta \geq \frac{\mu\gamma}{\beta}$ then $\mathsf{PNL}(\Delta, \eta) \geq 0$*

This simple result can be used by both wallet designers (who are optimizing η for users) and protocol designers (who can control μ, κ) as a way to minimize expected sandwich profit.

Sandwich Profitability is Local. For the sandwich attacker, their net profit in units of input token is $\Delta^{\text{sand}'} - \Delta^{\text{sand}}$. That is, they put in Δ^{sand} input tokens in and receive $\Delta^{\text{sand}'}$ input tokens out. Let $\mathsf{PNL}(\Delta, \eta) = \Delta^{\text{sand}'}(\Delta, \eta) - \Delta^{\text{sand}}(\Delta, \eta)$. We will generally write $\mathsf{PNL}(\Delta)$ and suppress explicit mention of η unless explicitly necessary.[4]. We now introduce the following definition:

Definition 2. We say that sandwich attacks for a sequence of trades $T = \{(\Delta_1, \eta_1), \ldots, (\Delta_n, \eta_n)\}$ are *strongly local* if we have for any index set $\{i_1, \ldots, i_J\}$, with $i_1 < \cdots < i_J$, $i_1 \geq 1$, $i_J = n$, and $J \leq n$, we have:

$$\mathsf{PNL}(\Delta_1 + \cdots + \Delta_{i_1}) + \cdots + \mathsf{PNL}(\Delta_{i_{j-1}+1} + \cdots + \Delta_{i_J}) \leq \sum_{i=1}^{n} \mathsf{PNL}(\Delta_i)$$

where $\mathsf{PNL}(\Delta_1 + \cdots + \Delta_j) = \Delta^{\text{sand}'}(\Delta_1 + \cdots + \Delta_j, \eta) - \Delta^{\text{sand}}(\Delta_1 + \cdots + \Delta_j, \eta)$.

Informally, this says that it is never more profitable to sandwich bundles of transactions versus simply sandwiching them individually. This implicitly places a constants on the curvature constants, insofar as they cannot be too large (e.g. very low liquidity and high price impact). If this property is not true, one finds themselves with a much more difficult problem to solve. In particular, if we have a finite block size (e.g. can process at most k trades), then we have to both solve the Knapsack problem of picking the k most profitable trades to sandwich while also needing to determine the sequence $\Delta_{[1]}, \ldots, \Delta_{[k]}$ that has the largest amount of price impact. Computationally, this will be at least NP-hard and potentially inapproximable. In Appendix F, we prove the following statement showing conditions under which it is never optimal to sandwich pairs of transactions, which we call *pairwise locality*.

Proposition 1. *Suppose that we have trades $T = \{(\Delta_1, \eta), \ldots, (\Delta_n, \eta)\}$ passing through a CFMM with curvature constants μ, κ. Then, given sufficient conditions in Equation (44), sandwich attacks are pairwise local, that is for all $i \in [n]$:*

$$\mathsf{PNL}(\Delta_i + \Delta_{i+1}) \leq \mathsf{PNL}(\Delta_i) + \mathsf{PNL}(\Delta_{i+1}) \tag{10}$$

In the following, we always maintain the assumption that sandwich attacks are strongly local, following Definition 2.

3 Reordering MEV

We now introduce the notion of reordering MEV for sandwich attacks. Throughout this section, we deal with a block of size h (*e.g.* can process n trades), and a sequence of trades $T_n \coloneqq \{(\Delta_1, \eta_1), \ldots, (\Delta_k, \eta_n)\}$ with a default ordering $i = 1, \ldots, n$, where $3n < h$ (this is to ensure that there are enough slots in the block for a sandwich attacker to insert trades).

[4]When we reason about two trades $(\Delta_1, \eta_1), (\Delta_2, \eta_2)$, we assume that we are dealing with a single slippage limit $\eta = \min(\eta_1, \eta_2)$. That is, we only account for slippage that the most conservative user inputs.

We assume that the slippage limits η_i are lower bounded by a single η, i.e. $\eta_i \leq \eta$ for all $i = 1, \ldots, n$.

We aim to understand how much sandwiching can cause user trades to have worsened price execution by reordering by a permutation $\pi \in S_n$ by studying the quantity we call *cost of feudalism*[5], or CoF:

$$\mathsf{CoF}(T_n) = \frac{\mathbf{E}_{\pi \sim S_n}\left[\max_{i \in [n]} |\mathsf{PNL}_{\pi(i)}(T_n) - \mathsf{PNL}_i(T_n)|\right]}{\mathbf{E}_{\pi \sim S_n}\left[\frac{1}{n}\sum_{i=1}^{n} |\mathsf{PNL}_{\pi(i)}(T_n) - \mathsf{PNL}_i(T_n)|\right]} \tag{11}$$

where $\mathsf{PNL}_i(T_n) = {\Delta^{\text{sand}}}'_i - \Delta^{\text{sand}}{}_i$ for the ith trade in T_n and $\mathsf{PNL}_{\pi(i)}(T_n)$ is the $\pi(i)$th trade in T_n for a permutation $\pi \in S_n$. We will slightly abuse notation and elide the explicit mention of T_n by writing PNL_i for brevity.

The interpretation of this quantity is that it compares the profit captured from sandwiching for the worst affected user to the average user, over all permutations of the trades, relative to a fixed ordering. Therefore, it characterizes the maximum amount that any individual user's price execution might be affected over reorderings. This excess value is captured by searchers and miners, and hence we call it the cost of feudalism. In particular, we seek to *upper bound* the numerator and *lower bound* the denominator of (45) to get a bound for $\mathsf{CoF}(T_n)$. Our main result (informally) shows the following:

Given sufficient locality and liquidity conditions on $\mu, \kappa, \alpha, \beta, \eta$, $\mathsf{CoF}(T_n)$ is $O(\log n)$.

Now, for a fixed ordering $i = 1, \ldots, n$, we show useful bounds on $\Delta^{\text{sand}}{}_i$ and ${\Delta^{\text{sand}}}'_i$ in this sequence using the upper bound derived in the previous section. To lighten notation, define $\xi_j = \Delta^{\text{sand}}{}_j + \Delta_j - {\Delta^{\text{sand}}}'_j$. We know that $\Delta^{\text{sand}}{}_i$ and ${\Delta^{\text{sand}}}'_i$ satisfy the equations:

$$G\left(\Delta^{\text{sand}}{}_i + \Delta_i + \sum_{j=1}^{i-1} \xi_j\right) - G\left(\Delta^{\text{sand}}{}_i + \sum_{j=1}^{i-1} \xi_j\right) = (1-\eta)G(\Delta_i)$$

$${\Delta^{\text{sand}}}'_i = {\Delta^{\text{sand}}}'_i + \Delta - G^{-1}(G(\Delta^{\text{sand}}{}_i + \Delta_i) - G(\Delta^{\text{sand}}{}_i))$$

Note that for a fixed ordering, the sandwich attacks $\Delta^{\text{sand}}{}_i$ and ${\Delta^{\text{sand}}}'_i$ depend on both the trades Δ_j and the sandwich attacks $\Delta^{\text{sand}}{}_j$ and ${\Delta^{\text{sand}}}'_j$ that came *before* (for $j = 1, \ldots, i-1$). This crucially requires us to bound the sandwich attack on trade i in terms of the partial trade drifts up to that trade, which are defined as follows:

Definition 3. Define the *partial trade drifts* as $\tilde{u}_i = \sum_{j=1}^{i} \xi_j$.

Note 2. *In the subsequent, we always work in the regime $\tilde{u}_i \approx 0$. That is, the trades are roughly mean-reverting. Our results hold when this is not true, but there are a number of higher-order correction terms that cloud the intuition of the main results.*

[5]The term 'cost of feudalism' analogizes the structure of feudal kingdoms to miners and searchers, who exact a tax in the form of sandwich attacks from users by reordering their trades.

We next upper and lower bound PNL_i in terms of $\tilde{u}_i$ and Δ_i using the following proposition:

Proposition 2. *If a sequence of trades $T_n = \{(\Delta_1, \eta_1), \ldots, (\Delta_n, \eta_n)\}$ is strongly local, then there exist constants $d, e > 0$, $e < 1$ which only depend on $\mu, \kappa, \eta = \max_i \eta_i$, such that:*

$$\Delta^{\text{sand}}{}_i \leq (1+d)\Delta_i + \sum_{j=1}^{i-1} (1+e)^{i-j} \xi_j \tag{12}$$

We prove this using a series of lemmas that are stated and proved in Appendix E. We note that this bound allows us to suitably 'localize' sandwich attacks by upper bounding $\Delta^{\text{sand}}{}_i$ by a term *linear* in the trade Δ_i, and geometrically decaying terms in the drifts $\tilde{u}_j$ for $j = 1, \ldots, i-1$. This will allow us to bound $\mathsf{CoF}(T_n)$ as a function of the curvature constants $\mu, \kappa, \alpha, \beta$, the slippage limit η, and the trade drifts $\tilde{u}_i$.

To bound $\mathsf{CoF}(T_n)$, we first show the sandwich profit function $\mathsf{PNL}_i = \Delta^{\text{sand}'}{}_i - \Delta^{\text{sand}}{}_i$ has curvature dependent on $\mu, \kappa, \alpha, \beta, \eta, \tilde{u}_i$:

Proposition 3. *If a set of trades $T = \{(\Delta_1, \eta_1), \ldots, (\Delta_n, \eta_n)\}$ is strongly local, there exist polynomials p, q of constant degree $d = O(1)$, such that PNL_i satisfies*

$$q(\mu, \kappa, \alpha, \beta, \eta, \tilde{u}_i)\Delta_i \leq \mathsf{PNL}_i \leq p(\mu, \kappa, \alpha, \beta, \eta, \tilde{u}_i)\Delta_i$$

We prove this result in Appendix G. We combine Proposition 3 with [CAE22, Thm. 1] and Lemma 7 of Appendix E to the following result (which is proved in Appendix H):

Proposition 4. *If a set of trades $T = \{(\Delta_1, \eta_1), \ldots, (\Delta_n, \eta_n)\}$ is strongly local, we have:*

$$\mathop{\mathbf{E}}_{\pi \sim S_n} \left[\max_{i \in [n]} |\mathsf{PNL}_{\pi(i)} - \mathsf{PNL}_i| \right] = O(\log n)$$

where the constant depends on $\mu, \kappa, \alpha, \beta, \eta, \tilde{u}_i$.

Proposition 5. *If a set of trades $T = \{(\Delta_1, \eta_1), \ldots, (\Delta_n, \eta_n)\}$ is strongly local, we have:*

$$\mathop{\mathbf{E}}_{\pi \sim S_n} \left[\frac{1}{n} \sum_{i=1}^{n} |\mathsf{PNL}_{\pi(i)} - \mathsf{PNL}_i| \right] = \Omega(1) \tag{13}$$

Combining Propositions 4 and 5, we have our main theorem:

Theorem 1. *If a set of trades $T_n = \{(\Delta_1, \eta_1), \ldots, (\Delta_n, \eta_n)\}$ is strongly local, then $\mathsf{CoF}(T_n) = O(\log n)$.*

We note that our result can be viewed as a competitive ratio or so-called *prophet inequality* bound [Hil83], except that we fix the distribution over permutations to be uniform over the entire symmetric group [ADK21].

4 Routing MEV

We now introduce the notion of routing MEV for sandwich attacks. Intuitively, this is the excess value that a sandwich attacker can extract from a user's trade over a network of CFMMs. It has been demonstrated that if there exists a universe of n tokens and there is a set of CFMMs on pairs of tokens, the routing problem is generically convex (sans transaction fees) [AECB22]. As such, it is possible for MEV searchers to take in order flow, represented by a set of trades, and optimally route transactions through a network of CFMMs.

To measure the impact of the presence of sandwich attacks on a CFMM network, we first define 'equilibrium routing', in which a user's trade is routed over a network from a source token to a destination token such that the price over all paths that are used is equal. Then, we measure the excess price impact caused by sandwichers, and define a measure of social cost incurred by the network, which is the net profit extracted due to sandwich attacks. The main result of this section is to prove that the price of anarchy, the ratio of the social cost of the worst-case equilibrium to the minimizer of the social cost, is constant and bounded by constants related to the slippage limits defined by the user and the liquidity of the CFMM network.

Before introducing the formal definition of the above quantities, we provide two illustrative examples that show cases in which sandwich attacks can worsen, and counterintuitively *improve* routing on CFMM networks, which we call the CFMM Pigou and CFMM Braess examples, respectively.

4.1 The CFMM Pigou Example

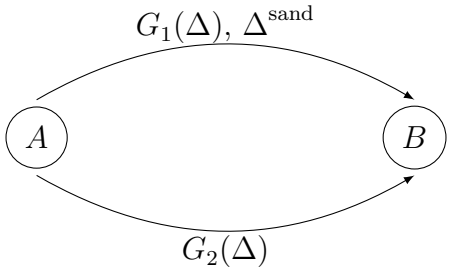

Figure 1: The CFMM Pigou Network

Optimal routing changes due to sandwiching Here we provide an explicit example of how optimal routing changes when there is a sandwich attack on the network in Figure 1. A trader desires to trade between tokens A and B and has two Uniswap pools to trade on. Suppose that we have two Uniswap constant product CFMMs with reserves (R_1, R_1') and (R_2, R_2') between the same pairs of assets (see Figure 1). Recall from §2.1 that Uniswap has a forward exchange function of the form $G_i(\Delta) = -\frac{k_i}{R_i+\Delta} - R_i'$ where $k_i = R_i R_i'$ is the prod-

uct of the reserves and a price function $g_i(\Delta) = \frac{k_i}{(R_i - \Delta)^2}$. Assume that $R_1 = R_2$ and $R_1' = R_2'$.

The user wants to trade a net quantity of Δ units of token A to B. In the absence of sandwich attacks on this network, in equilibrium, the user will split their trade in a fashion that equalizes the price they receive from each path that trades from A to B.[6] If we denote α as the fraction of Δ that trades on G_1, then the equilibrium, $\alpha^\star$ satisfies $g_1(\alpha^\star\Delta) = g_2((1-\alpha^\star)\Delta)$ whenever $1 > \alpha^\star > 0$. When $R_1 = R_2$ and $R_1' = R_2'$, it is seen that $\alpha^\star = \frac{1}{2}$. This equilibrium can also be computed as the maximizer of the function[7]:

$$F(\alpha) = G_1(\alpha\Delta) + G_2((1-\alpha)\Delta) \tag{14}$$

That is, $\alpha^\star = \arg\max_{\alpha\in[0,1]} F(\alpha)$ and F is a concave function of α (this can be seen by noting that G_i are concave in α and that for $f, g : \mathbb{R} \to \mathbb{R}$ concave, $f(\alpha x) + g((1-\alpha)x)$ is concave in α for $\alpha \geq 0$ [BBV04]). In the literature on routing games, F is called a *potential function* because for $\alpha \in (0,1)$ the optimality condition $\frac{\partial F}{\partial \alpha}(\alpha^\star) = 0$ gives us the equilibrium condition $g_1(\alpha^\star\Delta) = g_2((1-\alpha^\star)\Delta)$ [Rou07].

Now, assume there is a sandwich attacker Δ^{sand} present on G_1. The user submits a slippage limit η which quantifies the minimum output they are willing to receive from G_1. That is, η defines the sandwich attack Δ^{sand} by:

$$G_1(\alpha\Delta + \Delta^{\text{sand}}) - G_1(\Delta^{\text{sand}}) = (1-\eta)G_1(\alpha\Delta)$$

Recall from equation (4) that the optimal sandwich for this Uniswap pool therefore is:

$$\Delta^{\text{sand}}(\alpha,\eta) = \frac{-(\alpha\Delta + 2R_1) + \sqrt{(\alpha\Delta + 2R_1)^2 - 4(R_1^2 + R_1\alpha\Delta)\frac{-\eta}{1-\eta}}}{2}$$

With the presence of the sandwich, the function $F(\alpha)$ is modified to:

$$F(\alpha,\eta) = G_1(\alpha\Delta + \Delta^{\text{sand}}(\alpha,\eta)) - G_1(\Delta^{\text{sand}}(\alpha,\eta)) + G_2((1-\alpha)\Delta) \tag{15}$$

and the equilibrium is now:

$$\alpha^\star(\eta) = \arg\max_{\alpha\in[0,1]} F(\alpha,\eta) \tag{16}$$

That is, the net output is modified to reflect the reduced amount in output token the user receives as a result of sandwiching. The sandwich attacker is making it more expensive for the user to trade on G_1. As a result, for $\eta > 0$ the equilibrium ends up being shifted to $\alpha^\star < \frac{1}{2}$. So, the user gradually trades a smaller fraction of their trade on G_1. In Figure 2 we plot the function $F(\alpha,\eta)$ as η varies, for representative values $R = 1.5$, $R' = 2$, $\Delta = 0.5$. It can be seen that at $\eta = 0$, the equilibrium reduces to $\alpha^\star = \frac{1}{2}$. However, when η increases, the maximizer of F moves to less than $\frac{1}{2}$. In Figure 3, we plot how the equilibrium varies with slippage η, and it can be seen that the equilibrium is a decreasing function of η.

[6] The interpretation of this equilibrium condition is that the amount of Δ placed on any path cannot be marginally changed to improve the price that the user got on that path.

[7] In this example, the function F is also the *net output* the user receives from the network.

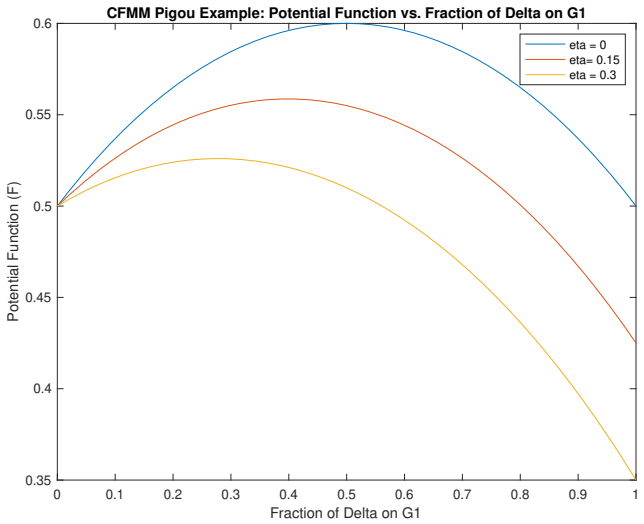

Figure 2: $F(\alpha, \eta)$ as η varies for $R = 1.5$, $R' = 2$, $\Delta = 0.5$. As can be seen, the equilibrium $\alpha^\star$ steadily reduces as η is increased. Therefore, the sandwicher is causing the equilibrium to be less than $\frac{1}{2}$. Further, because F also serves the role as the net output function, the amount of output token the user gets in equilibrium steadily worsens due to the presence of the sandwicher.

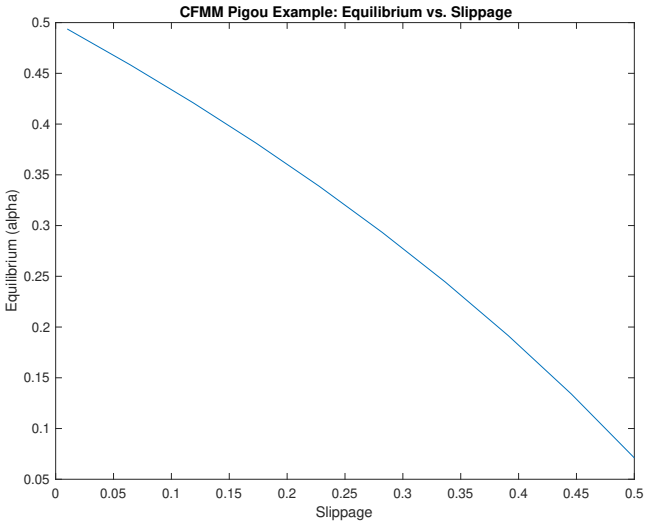

Figure 3: Equilibrium $\alpha^\star$ versus slippage in the CFMM Pigou example. As slippage increases, the equilibrium is such that the user prefers to increasingly not trade on G_1 due to the presence of the sandwicher.

4.2 The CFMM Braess Example

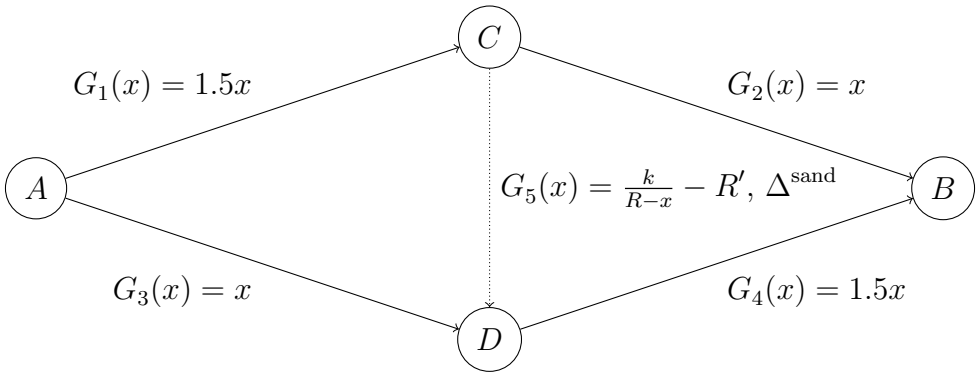

Figure 4: The CFMM Braess Network, for $R = 1$, $R' = 1$, $k = 1$, $\Delta = 1$.

Sandwichers can improve routing. As mentioned in the introduction, the constant PoA for routing can be interpreted as stating that sandwiches do not significantly degrade routing quality. We now construct an example representing an "inverse Braess paradox" in CFMMs in which the presence of sandwich attackers in fact *improves* the network flow, and the profit that the sandwicher can extract from the network is bounded. This demonstrates that while sandwiches necessarily worsen individual trade price impact on a given CFMM, they explicitly can improve the 'social welfare' (measured as the net output that users receive over all paths) for trades on a network. The paradox proceeds in three steps:

1. We begin with a diamond network of CFMMs trading pairs of tokens, as in Figure 4. Initially, there is no CFMM trading the pair C and D (i.e. $G_5(x) = 0$). A net trade of Δ units comes to trade between A and B. We set $\Delta = 1$. In the absence of the middle link, $G_5(x)$, the equilibrium is to split the trade halfway between paths $A \to C \to B$ and $A \to D \to B$. If we denote $\alpha^\star$ to be the fraction of Δ that, in equilibrium goes on path $A \to C \to B$, then the net output for $\alpha^\star = \frac{1}{2}$ is given by:

$$\begin{aligned} G^{\mathsf{net}}(\alpha^\star, \Delta) &= G_2\left(G_1\left(\frac{1}{2}\right)\right) + G_4\left(G_3\left(\frac{1}{2}\right)\right) \\ &= \frac{1}{2}1.5 + \frac{1}{2}1.5 = 1.5 \end{aligned}$$

2. A CFMM trading between assets C and D is added ($G_5(x) \neq 0$). The presence of the additional route $A \to C \to D \to B$ causes the equilibrium to shift such that the net output the user receives over all three paths *reduces* as compared to the case in which $G_5(x) = 0$. Formally, the equilibrium condition for $\alpha > 0$ (by symmetry) is now:

$$g_2(g_1(\alpha^\star + (1-2\alpha^\star)\Delta) - g_1((1-2\alpha^\star)\Delta)) = g_4(g_5(g_1((\alpha^\star + (1-2\alpha^\star)\Delta)) - g_1(\alpha^\star\Delta))$$

where as before, $g_i(\Delta) = G_i'(\Delta)$, is the price impact function on link i.

The net output for equilibrium $\alpha^\star$ satisfying the above equation is given by:

$$G^{\mathsf{out}}(\alpha^\star, \Delta) = G_2(G_1(\alpha\Delta)) + G_4(G_5(G_1(1-2\alpha^\star))) + G_4(G_3(\alpha\Delta)$$

Indeed, it can be verified numerically that for the parameters chosen in Figure 4, and where G_5 is the Uniswap forward exchange function, the equilibrium shifts from $(\alpha_1^\star, \alpha_2^\star) = \left(\frac{1}{2}, \frac{1}{2}\right)$ to $(\alpha_1^\star, \alpha_2^\star, \alpha_3^\star) \approx (0.29, 0.41, 0.29)$ and correspondingly the net output reduces.

3. Now, a sandwicher is added on the CFMM trading between assets C and D. The user specifies a slippage limit η over the path $A \to C \to D \to B$, and the sandwicher computes the optimal sandwich $\Delta^{\text{sand}}(\Delta, \alpha, \eta)$ according to:

$$\Delta^{\text{sand}}(\Delta, \alpha, \eta) = \frac{-((1-2\alpha)\Delta + 2R) + \sqrt{((1-2\alpha)\Delta + 2R)^2 - 4(R^2 + R(1-2\alpha)\Delta)\frac{-\eta}{1-\eta}}}{2}$$

The equilibrium under sandwiching, $\alpha^\star(\eta)$, for $\alpha > 0$ is modified to account for the excess price impact that the sandwicher causes:

$$\begin{aligned} &g_2(g_1(\alpha^\star + (1-2\alpha^\star)\Delta) - g_1((1-2\alpha^\star)\Delta)) \\ &= g_4(g_5(g_1((\alpha^\star + (1-2\alpha^\star)\Delta)) - g_1(\alpha^\star\Delta) + \Delta^{\text{sand}}(\Delta, \alpha, \eta)) - g_5(\Delta^{\text{sand}}(\Delta, \alpha, \eta))) \end{aligned}$$

It can be seen in Figure 5 that if $\Delta^{\text{sand}}(\Delta, \alpha, \eta)$ is increasing in η, then as the user's slippage η increases, the equilibrium will steadily shift such that the user no longer prefers to trade with middle link.

In Figure 5, we see the net quantity of output tokens received by the user if they submitted a slippage limit of η. We can see that the output increases in η until a certain slippage limit, as which point the user returns to the $(\alpha_1^*, \alpha_2^*, \alpha_3^*) = (\frac{1}{2}, \frac{1}{2}, 0)$ equilibrium. Unlike the Pigou example of the last section, this demonstrates that sandwicher profit is *not* strictly monotonically increasing in η as rational users avoid the link the sandwicher is on. This is the 'inverse' part of the Braess-like paradox described — instead of migrating all of one's flow to the middle link (as in the classic Braess Paradox), the presence of the sandwicher forces users to reallocate across the network.

We can see a similar effect in Figure 6 which depicts the profit $\mathsf{PNL}(\Delta, \eta) = \Delta^{\text{sand}'}(\Delta, \eta) - \Delta^{\text{sand}}(\Delta, \eta)$ for the sandwicher. Again, we see that the sandwich attacker's profit has a maxima and begins to decrease as users allocate away from the middle link in the network graph. In some ways, the sandwich attacker can be thought of as acting like a 'decentralized traffic controller', where users' best responses to the existence of a sandwich attacker lead to better flow (*e.g.* tokens outputted) across the network.

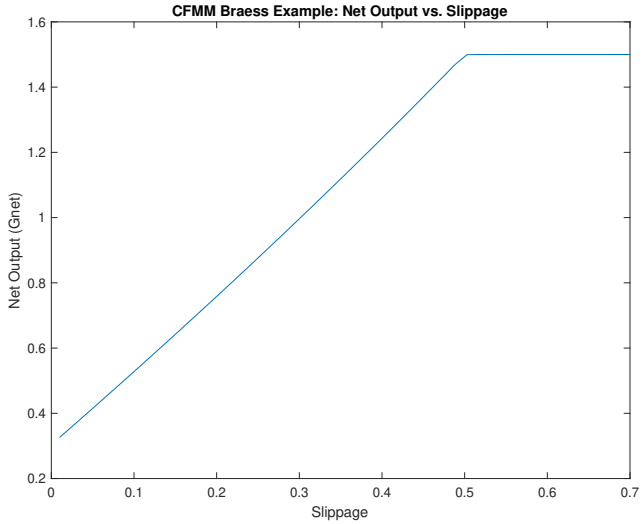

Figure 5: $G^{\mathsf{net}}(\alpha^\star(\eta), \eta)$ as η varies for the CFMM Braess network. It can be seen that as η increases, the equilibrium shifts such that the user no longer prefers to trade on the middle link. Therefore, the net output goes back up to its original level (without the middle link) of 1.5.

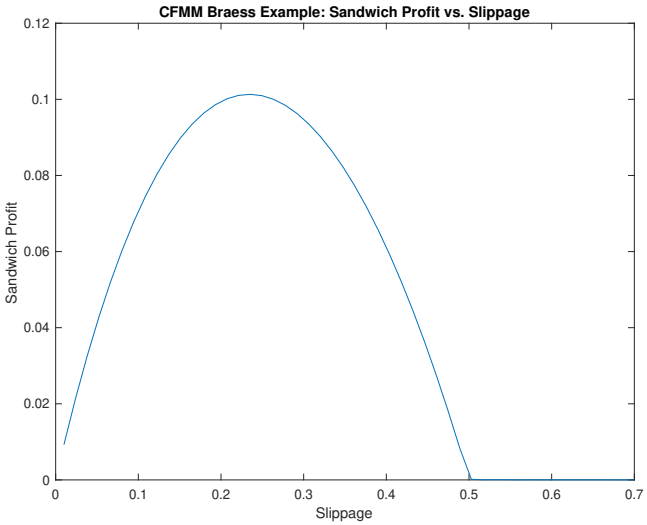

Figure 6: The sandwich profit $\Delta^{\text{sand}'} - \Delta^{\text{sand}}$ in equilibrium as slippage increases. As η increases, the sandwicher is able to steadily increase profit, but there is a maximum amount after which the user begins reducing their trade on the middle link. The sandwich profit therefore steadily goes down.

4.3 Price of Anarchy

We now provide the general formulation of sandwich attacks on networks of CFMMs. Suppose that we have a graph $G = (V, E)$ where each vertex $A \in V$ denotes a token and each edge $e = (A, B) \in E$ represents a CFMM for trading between tokens A and B. Associated to each $e \in E$ is a price function $g_e(\cdot)$ and a corresponding forward exchange function $G_e(\cdot)$ that executes a trade at the CFMM associated to edge e. We note that the argument to this function is the *total* trade Δ_e that trades over this CFMM (*i.e.* an edge $e \in E$ may belong to multiple paths that connect source and destination vertices on the network).

Equilibrium Splitting. The space of trades is defined as $\mathcal{T} = \mathbf{R} \times \mathbf{R}_+ \times V \times V$. Each trade $T \in \mathcal{T}$ where $T = (\Delta, \eta, A, B)$ specifies an amount, Δ to be traded from vertex A to vertex B along with a slippage limit η. We will abuse notation and utilize $(\Delta_{AB}, \eta_{AB}) \in T$ to refer to $(\Delta, \eta, A, B) \in T$. Define *Routing MEV* as any excess value that can be extracted from adjusting how transactions are executed on this graph. In particular, we will study the effect of sandwich attacks on this CFMM network. Given the above examples of this kind of MEV, we define notion of worst case profit that can be extracted by sandwich attackers on the network as compared to equilibrium routing inspired from price of anarchy in the game theory literature [Rou05]. Informally, we prove:

The price of anarchy of a network of CFMMs under sandwich attacks is bounded by a constant that depends on the slippage η and constants $\kappa, \mu, \alpha, \beta$ of the CFMMs.

We now introduce the notion of trade splittings. Consider a trade $T = (\Delta, \eta, A, B)$. Denote the set of paths from A to B as $\mathcal{P}$. Let $\alpha \in \{x \in \mathbf{R}^{|\mathcal{P}|} : \sum_i x_i = 1\} = \hat{S}_{|\mathcal{P}|}$ be the *splitting vector* that tells us how much of the Δ units of token A are routed onto each path. In order to generalize the optimization problem of eq. (14), we first define the notion of an equilibrium splitting directly in terms of prices:

Definition 4. A splitting vector $\alpha^\star \in \hat{S}_{|\mathcal{P}|}$ is said to be an *equilibrium splitting in price*[8] if for all $p \in \mathcal{P}$, $\alpha_p^\star > 0$ implies:

$$g_p(\alpha_p^\star) \leq g_{p'}(\alpha_{p'}^\star) \tag{17}$$

for every $p' \in \mathcal{P}$.

This definition says that over *all* paths on which there is a nonzero trade in equilibrium, the price experienced by the user must be equal.[9] There is an equivalent optimization in terms of a potential function, as in the Pigou and Braess examples; however, it is easier to prove Price of Anarchy bounds in terms of price. We will compute optimal slippage limits using a quantity output function, whereas we will compute the equilibrium splitting in price and rely on the equivalence of Appendix B to map between price and quantity slippage limits.

[8] The existence of such an equilibrium can be established using standard results in nonatomic routing games [Rou05].

[9] This can be viewed as a subgradient condition, as per [AAE+21]

Sandwich Attacks on Network Graphs. Let $g_e(\cdot)$ be the price impact function for each edge $e \in E$. Suppose that a user has submitted a trade (Δ_{AB}, η_{AB}). The slippage limit η_{AB} indicates that the user is willing to accept a minimum amount of output token B from the network. However, unlike the single-edge case of §2.3, we now need to construct slippage limits for *each edge* in a path. In order to bound the price impact of a sandwich attack, we first have to solve for the implied slippage limits on each edge in a path before solving for bounds on sandwich size.

We first argue that these slippage limits can be computed uniquely without the presence of sandwiches. Specifically, define $G_e(\Delta) = \int_0^\Delta g_e(-t)dt$ to be the *output function* of edge $e \in E$ and for a path $p = (e_1, \ldots, e_{|p|}) \in \mathcal{P}$. Then the (quantity) slippage limits $\eta_1, \ldots, \eta_{|p|} = \eta$ are defined recursively as

$$G_{e_{|p|-1}}(\Delta_{e_{|p|-1}}) = (1-\eta)G_{e_{|p|}}(\Delta_{|p|})$$
$$G_{e_{|p|-k-1}}(\Delta_{e_{|p|-k-1}}) = (1-\eta_k)G_{e_{|p|-k}}(\Delta_{|p|-k})$$

This recursion is effectively a dynamic program that can be solved and bounded using curvature. Moreover, the convexity of each G_e ensures that we can uniquely solve for an optimal η_k.

Next, suppose that there is a sandwich attacker present on every edge that computes an optimal sandwich attack ${\Delta^{\text{sand}}}_e$ over the cumulative trade that enters that edge. This sandwich attack mutates the above recursion into the following:

$$G_{e_{k-1}}(\Delta_{e_{k-1}} + {\Delta^{\text{sand}}}_{e_{k-1}}) - G_{e_{k-1}}({\Delta^{\text{sand}}}_{e_{k-1}}) = (1-\eta_k)G_k(\Delta_k) - G_{e_{k-2}}({\Delta^{\text{sand}}}_{e_{k-2}})$$

The left hand side of this equation represents the excess price impact that occurs at edge e_k when the path $p \in \mathcal{P}$ is traversed. The right hand side is contribution to the terminal impact (a boundary term) from the e_{k-1}th edge. This output recursion for solving for η_i is not guaranteed to be convex as it is the difference of convex functions [BBV04], which means there could be multiple sequences $\eta_1, \ldots, \eta_{|p|}$ and $\eta'_1, \ldots, \eta'_{|p|}$ that could lead to the same optimal output. However, using curvature bounds we can bound this recursion which subsequently allows us to provide bounds on ${\Delta^{\text{sand}}}_e$:

Proposition 6. *There exist functions $f(\kappa, \mu, \eta)$ and $g(\kappa, \mu, \eta)$ such that*

$${\Delta^{\text{sand}}}_e \leq f(\kappa, \mu, \eta)^{|p|}\Delta_e \tag{18}$$

and

$$\Delta_e \leq g(\kappa, \mu, \eta)^{|p|}{\Delta^{\text{sand}}}_e \tag{19}$$

for all $e \in E$ and paths $p \in \mathcal{P}$.

These bounds will be important in defining the excess price impact realized by the user and the sandwich profit realized by attacker. This proposition (and associated desiderata about analyzing the aforementioned recursions) is described in Appendix K.1.

Social Cost and the Price of Anarchy. To define a measure of social cost, we use the sandwich size $\Delta^{\text{sand}}{}_e$ on every edge. That is, we want to find a splitting α that minimizes the trade-weighted price impact that the sandwicher causes on the network[10]. To do this, we define the edge cost functions for $e \in E$:

$$c_e(\alpha, \Delta) = p_e(g_e(\Delta^{\text{sand}}{}_e + \Delta_e) - g_e(\Delta_e)) \tag{20}$$

where p_e is the price of the output token on edge $e \in E$ in terms of a numéraire in an infinitely liquid reference market, which, without loss of generality, whose label we refer to as \$. Note that the units of c_e are in $\frac{\$}{\text{input token}}$, and the edge costs represent the excess price impact that the presence of the sandwich attack $\Delta^{\text{sand}}{}_e$ causes on the user's trade. Then, we say that the cost of sandwiching on the network (measured in \$) is:

$$C(\alpha, \Delta) = \sum_e c_e(\alpha, \Delta)\Delta_e \tag{21}$$

This is measuring the trade-weighted price impact that the presence of a sandwicher on each edge has on the network. The above definition allows us to define the price of anarchy:

Definition 5. Let $\alpha^\star \in \hat{S}_{|\mathcal{P}|}$ be any equilibrium splitting. Then,

$$\mathsf{PoA}(\Delta) = \frac{C(\alpha^\star, \Delta)}{\inf_\alpha C(\alpha, \Delta)} \tag{22}$$

Our main theorem is a result that bounds the price of anarchy, which is the ratio of the equilibrium cost to the optimum as a function of the curvature and slippage parameters of the network. Notably, the price of anarchy is upper bounded by a *constant*. This is shown by connecting the price functions of the CFMMs to (λ, μ)-smoothness arguments from [Rou15].

Theorem 2. *Suppose that* $f(\kappa, \mu, \eta), g(\kappa, \mu, \eta) \in O((1 + (\alpha\beta\kappa)^{O(1)})^{1/\mathsf{diam}(G)}$. *Then there exists a function* $C(\kappa, \alpha, \beta, \mu, \eta)$ *that is constant in the size of the network graph* G *such that*

$$\mathsf{PoA}(\Delta) \leq C(\kappa, \alpha, \beta, \mu, \eta) \tag{23}$$

The proof of the propositions and the main theorem can be found in Appendix K. One can view the condition of bounds on f and g informally as saying that provided there is enough liquidity on each edge in the graph (measured by the μ, κ dependence in f, g), then the PoA from sandwiching is constant. We can interpret this as a weak generalization of the Braess example, demonstrating that sandwiches, provided there is enough liquidity, do not necessarily cause asymptotically worse performance in CFMM network graphs. We note that our result can likely be sharpened and that the constants are not tight.

[10]Note that this can be equivalently stated as finding a splitting α that maximizes the net output G^{net} that the user receives over the network.

5 Conclusion and Future Work

In this paper, we provided the first formal description of generic sandwich attacks against arbitrary CFMMs. Using this description, we were able to explicitly compute bounds that depend on curvature and liquidity on sandwich attack profitability. These bounds allowed us to analyze the two most prominent forms of CFMM MEV: reordering and routing MEV. For reordering MEV, we found the somewhat unexpected result that given an order flow of n trades, the worst case price impact received by a trader is only logarithmically worse than the average case price impact. Much more paradoxically, we found that for routing MEV, there can be scenarios where sandwich attacks *increase* efficiency and/or social welfare for users when n trades are routed across a network of CFMMs. We generalized this example to a larger set of network graphs of CFMMs and showed that the Price of Anarchy for routing MEV is constant given sufficient liquidity on the network graph. The key insight to this calculation was adapting CFMM price impact functions in a way that we could apply (λ, μ)-smoothness results from [Rou15].

Prior works on MEV [BCLL21a, ZQG21, BDKJ21, QZG21, HW22] have focused on illuminating either specific profitable attacks or methodologies for the numerical or empirical estimation of MEV. We believe that this is the first paper that adds more formal algorithmic game theory and probability results about the impact of MEV on users. Such results provide both asymptotic, theoretical insight into the nature of MEV but also provide some mitigating strategies for protocol developers. For instance, Theorem 1 (and the upper and lower bounds on sandwich profitability) provide guidance on how to choose and/or set slippage limits η as a function of CFMM curvature to reduce the impact of MEV. These bounds are relatively weak and can be improved if one specializes to a smaller set of CFMMs.

From a theoretical perspective, our main results involve bounds on competitive ratios. While we only demonstrated that the Cost of Feudalism is $O(\log n)$ for local trades, it is likely that existing work on competitive ratios can generalize this to longer-ranged (non-local) sequences of transactions. For instance, prophet inequalities are known for online Knapsack problems that are similar to those used in MEV [JMZ22]. As there are some basic results on composition of prophet inequalities [Luc17] that could allow for us to extend both routing and reordering bounds to non-local forms of MEV. Another direction for extending the results of these papers includes mapping long-ranged MEV strategies to bounded auctions and utilizing competitive ratio results for these auctions (*e.g.* [AMMN22]).

The remaining set of papers in this series will focus on other types of MEV and the interaction between MEV and privacy. Many of the results in this paper were inspired by [CAE22] which made a direct connection between MEV profit versus the cost of privacy. This connection is most easily analyzed in CFMMs, where the 'cost of privacy' can be directly interpreted via the excess price impact or fees that a user has to pay to ensure that MEV searchers have negligible profitability. Extending this analogy to more complex forms of MEV such as cross-chain MEV [OSS+21] and MEV related to liquidations [KCCM20] is future work. We note that many (if not most) of the routing MEV results can be ported to cross-chain MEV, as a number of protocols such as Synapse [Teab], use CFMMs to convert between synthetic and real assets. However, cross-chain MEV can be more rich in that

'long range' MEV strategies can be much more profitable than on a single chain. Our next paper will try to analyze the 'range' or 'distance' dependence of MEV strategies, with the understanding that results like Proposition 2 show that CFMM MEV is inherently local and not long ranged.

6 Acknowledgments

We thank Guillermo Angeris and Alex Evans for helpful comments and feedback.

References

[AAC+11] Mário S Alvim, Miguel E Andrés, Konstantinos Chatzikokolakis, Pierpaolo Degano, and Catuscia Palamidessi. Differential privacy: on the trade-off between utility and information leakage. In *International Workshop on Formal Aspects in Security and Trust*, pages 39–54. Springer, 2011.

[AAE+21] Guillermo Angeris, Akshay Agrawal, Alex Evans, Tarun Chitra, and Stephen Boyd. Constant function market makers: Multi-asset trades via convex optimization. 2021.

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

[ACE22] Guillermo Angeris, Tarun Chitra, and Alex Evans. When Does The Tail Wag The Dog? Curvature and Market Making. *Cryptoeconomic Systems*, 2(1), June 2022. https://cryptoeconomicsystems.pubpub.org/pub/angeris-curvature-market-making.

[ADK21] Makis Arsenis, Odysseas Drosis, and Robert Kleinberg. Constrained-order prophet inequalities. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2034–2046. SIAM, 2021.

[AEC21] Guillermo Angeris, Alex Evans, and Tarun Chitra. A note on bundle profit maximization. 2021.

[AECB22] Guillermo Angeris, Alex Evans, Tarun Chitra, and Stephen Boyd. Optimal routing for constant function market makers. In *Proceedings of the 23rd ACM Conference on Economics and Computation*, EC '22, page 115–128, New York, NY, USA, 2022. Association for Computing Machinery.

[Aga00] Ravi P Agarwal. *Difference equations and inequalities: theory, methods, and applications*. CRC Press, 2000.

[AH21] Anish Agnihotri and Hasu. A guide to designing effective nft launches, Oct 2021.

[AKC+21] Guillermo Angeris, Hsien-Tang Kao, Rei Chiang, Charlie Noyes, and Tarun Chitra. *Cryptoeconomic Systems*, (1), April 2021.

[AMMN22] Saeed Alaei, Ali Makhdoumi, Azarakhsh Malekian, and Rad Niazadeh. Descending price auctions with bounded number of price levels and batched prophet inequality. In *Proceedings of the 23rd ACM Conference on Economics and Computation*, EC '22, page 246, New York, NY, USA, 2022. Association for Computing Machinery.

[AO21] Sunny Agrawal and Dev Ojha. Vision for osmosis, May 2021.

[BBV04] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[BCLL21a] Massimo Bartoletti, James Hsin-yu Chiang, and Alberto Lluch-Lafuente. Maximizing extractable value from automated market makers. *arXiv preprint arXiv:2106.01870*, 2021.

[BCLL21b] Massimo Bartoletti, James Hsin-yu Chiang, and Alberto Lluch-Lafuente. A theory of automated market makers in defi. In *International Conference on Coordination Languages and Models*, pages 168–187. Springer, 2021.

[BDKJ21] Kushal Babel, Philip Daian, Mahimna Kelkar, and Ari Juels. Clockwork finance: Automated analysis of economic security in smart contracts. *arXiv preprint arXiv:2109.04347*, 2021.

[BO22] Joseph Bebel and Dev Ojha. Ferveo: Threshold decryption for mempool privacy in bft networks. Cryptology ePrint Archive, Paper 2022/898, 2022. `https://eprint.iacr.org/2022/898`.

[CAE21] Tarun Chitra, Guillermo Angeris, and Alex Evans. How liveness separates cfmms and order books. 2021.

[CAE22] Tarun Chitra, Guillermo Angeris, and Alex Evans. Differential privacy in constant function market makers. In *International Conference on Financial Cryptography and Data Security*. Springer, 2022.

[DGK+19] 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:1904.05234 [cs]*, April 2019.

[DJW13] John C Duchi, Michael I Jordan, and Martin J Wainwright. Local privacy and statistical minimax rates. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 429–438. IEEE, 2013.

[DJW14] John C Duchi, Michael I Jordan, and Martin J Wainwright. Privacy aware learning. *Journal of the ACM (JACM)*, 61(6):1–57, 2014.

[dV21] Henry de Valence. Sealed-bid batch auctions, 2021.

[@ha22] User: @hagaetc. Dex metrics dune analytics dashboard, Jun 2022.

[Hil83] TP Hill. Prophet inequalities and order selection in optimal stopping problems. *Proceedings of the American Mathematical Society*, 88(1):131–137, 1983.

[How97] Ralph Howard. The inverse function theorem for lipschitz maps. *Lecture Notes*, 1997.

[HW22] Lioba Heimbach and Roger Wattenhofer. Eliminating sandwich attacks with the help of game theory. *arXiv preprint arXiv:2202.03762*, 2022.

[JMZ22] Jiashuo Jiang, Will Ma, and Jiawei Zhang. Tight guarantees for multi-unit prophet inequalities and online stochastic knapsack. In *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1221–1246. SIAM, 2022.

[JSZ+21] Aljosha Judmayer, Nicholas Stifter, Alexei Zamyatin, Itay Tsabary, Ittay Eyal, Peter Gaži, Sarah Meiklejohn, and Edgar Weippl. Sok: Algorithmic incentive manipulation attacks on permissionless pow cryptocurrencies. In *International Conference on Financial Cryptography and Data Security*, pages 507–532. Springer, 2021.

[KCCM20] Hsien-Tang Kao, Tarun Chitra, Rei Chiang, and John Morrow. An analysis of the market risk to participants in the compound protocol. In *Third International Symposium on Foundations and Applications of Blockchains*, 2020.

[KDL+21] Mahimna Kelkar, Soubhik Deb, Sishan Long, Ari Juels, and Sreeram Kannan. Themis: Fast, strong order-fairness in byzantine consensus. *Cryptology ePrint Archive*, 2021.

[KZGJ20] Mahimna Kelkar, Fan Zhang, Steven Goldfeder, and Ari Juels. Order-fairness for byzantine consensus. In *Annual International Cryptology Conference*, pages 451–480. Springer, 2020.

[Luc17] Brendan Lucier. An economic view of prophet inequalities. *ACM SIGecom Exchanges*, 16(1):24–47, 2017.

[McS22] Michael McSweeney. Defi protocol inverse finance suffers exploit, loss of $15 million in crypto, Apr 2022.

[MF13] Ali Makhdoumi and Nadia Fawaz. Privacy-utility tradeoff under statistical uncertainty. In *2013 51st Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1627–1634. IEEE, 2013.

[OSS+21] Alexandre Obadia, Alejo Salles, Lakshman Sankar, Tarun Chitra, Vaibhav Chellani, and Philip Daian. Unity is strength: A formalization of cross-domain maximal extractable value. *arXiv preprint arXiv:2112.01472*, 2021.

[QZG21] Kaihua Qin, Liyi Zhou, and Arthur Gervais. Quantifying blockchain extractable value: How dark is the forest? *arXiv preprint arXiv:2101.05511*, 2021.

[QZLG20] Kaihua Qin, Liyi Zhou, Benjamin Livshits, and Arthur Gervais. Attacking the defi ecosystem with flash loans for fun and profit. *arXiv preprint arXiv:2003.03810*, 2020.

[Ree03] Bruce Reed. The height of a random binary search tree. *Journal of the ACM (JACM)*, 50(3):306–332, 2003.

[Rou05] Tim Roughgarden. *Selfish routing and the price of anarchy*. MIT press, 2005.

[Rou07] Tim Roughgarden. Routing games. *Algorithmic game theory*, 18:459–484, 2007.

[Rou15] Tim Roughgarden. Intrinsic robustness of the price of anarchy. *Journal of the ACM (JACM)*, 62(5):1–42, 2015.

[RST17] Tim Roughgarden, Vasilis Syrgkanis, and Eva Tardos. The price of anarchy in auctions. *Journal of Artificial Intelligence Research*, 59:59–101, 2017.

[SCDFSM17] Jordi Soria-Comas, Josep Domingo-Ferrer, David Sánchez, and David Megías. Individual differential privacy: A utility-preserving formulation of differential privacy guarantees. *IEEE Transactions on Information Forensics and Security*, 12(6):1418–1429, 2017.

[SRP13] Lalitha Sankar, S Raj Rajagopalan, and H Vincent Poor. Utility-privacy tradeoffs in databases: An information-theoretic approach. *IEEE Transactions on Information Forensics and Security*, 8(6):838–852, 2013.

[SS21] Stefan Stankovic and Stefan Stankovic. $136m lost as cream finance suffers another flash loan attack, Oct 2021.

[Teaa] Flashbots Team. Mev explore.

[Teab] Synapse Team. What is synapse?

[Uni22] Uniswap. Swaps, Jun 2022.

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

[Zin21] Noah Zinsmeister. Uniswap. `https://github.com/Uniswap/v2-periphery/blob/master/contracts/UniswapV2Router02.sol#L293`, 2021.

[ZQG21] Liyi Zhou, Kaihua Qin, and Arthur Gervais. A2mm: Mitigating frontrunning, transaction reordering and consensus instability in decentralized exchanges. *arXiv preprint arXiv:2106.07371*, 2021.

[ZQT+21] Liyi Zhou, Kaihua Qin, Christof Ferreira Torres, Duc V Le, and Arthur Gervais. High-frequency trading on decentralized on-chain exchanges. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 428–445. IEEE, 2021.

[Züs21] Patrick Züst. Analyzing and preventing sandwich attacks in ethereum. 2021.

A Uniswap Sandwich Example

The defining relation for Δ^{sand} for Uniswap is:

$$-\frac{k}{R+\Delta+\Delta^{\text{sand}}}+R'+\frac{k}{R+\Delta^{\text{sand}}}-R'=(1-\eta)\left(-\frac{k}{R+\Delta}+R'\right)$$

Cancelling R' and putting the left and hand sides over a common denominator:

$$\frac{-k(R+\Delta^{\text{sand}})+k(R+\Delta+\Delta^{\text{sand}})}{(R+\Delta+\Delta^{\text{sand}})(R+\Delta^{\text{sand}})}=(1-\eta)\left(\frac{-k+R'R+\Delta R'}{R+\Delta}\right)$$

Noting that $k=R'R$, and simplifying the left hand side:

$$\frac{k\Delta}{R^2+R\Delta+R\Delta^{\text{sand}}+R\Delta^{\text{sand}}+\Delta\Delta^{\text{sand}}+\Delta^{\text{sand}^2}}=(1-\eta)\frac{\Delta R'}{R+\Delta}$$

Which, after cancelling $R'\Delta$ on both sides, raising both sides of the equation to the -1 power, and multiplying by R gives us the equation:

$$\Delta^{\text{sand}^2}+\Delta^{\text{sand}}(\Delta+2R)+(R^2+R\Delta)=\frac{1}{1-\eta}(R^2+R\Delta)$$

and moving the right hand side to the left we have:

$$\Delta^{\text{sand}^2}+\Delta^{\text{sand}}(\Delta+2R)+(R^2+R\Delta)\left(1-\frac{1}{1-\eta}\right)=0$$

We solve the above quadratic to give us:

$$\Delta^{\text{sand}}=\frac{-(\Delta+2R)\pm\sqrt{(\Delta+2R)^2-4(R^2+R\Delta)\frac{-\eta}{1-\eta}}}{2}$$

Taking the positive root (which is the correct root to take, as when $\eta=0$, the positive root gives us $\Delta^{\text{sand}}=0$, and also gives us that Δ^{sand} is *increasing* in η), we have:

$$\Delta^{\text{sand}}=\frac{-(\Delta+2R)+\sqrt{(\Delta+2R)^2-4(R^2+R\Delta)\frac{-\eta}{1-\eta}}}{2}$$

B Price Slippage and Quantity Slippage are Equivalent

Uniswap enforces slippage limits [Zin21] in quantity space rather than price space. In particular, the interface for making a trade takes in an input quantity Δ and a minimum output quantity $\Delta'_{\min}$ and enforces that the output quantity received for Δ, Δ' always satisfies $\Delta' \geq \Delta'_{\min}$. In this section, we show that for Uniswap, there is a way to map slippage limits defined in terms of quantity to those defined in terms of price. For general CFMMs, this is also possible to show, but it is quite impractical to perform in practice as it involves inverting the CFMM invariant function.

Recall that the output quantity for a trade of size Δ, is defined via the forward transfer function $F(\Delta)$ [AAE$^+$21, §4.1]:

$$\Delta' = F(\Delta) = \int_0^{-\Delta} g(t)dt$$

Enforcing the Uniswap condition on quantity is equivalent to

$$F(\Delta + \Delta^f) - F(\Delta^f) \geq (1 - \eta^q)F(\Delta) \tag{24}$$

for any front running trade Δ^f. The left hand side of this equation is the output quantity received if one submits a trade of size Δ after someone has submitted a trade of size Δ^f ahead of my trade. The right hand side provides the notion of minimum quantity with η^q defined as a quantity space slippage limit. Put together, equation (24) states that the output quantity must at least be $(1 - \eta^q)$ times the quantity that was expected assuming no front running transactions take place. Our claim in this section is that one can compute $\eta^q(\eta)$ to go between quantity space slippage limits and price space slippage limits.

Recall that there exists $\gamma, \xi > 0$ such that the forward transfer function satisfies $F(\Delta) \geq \gamma\Delta$ and $F(\Delta) \leq \xi\Delta^2$. This implies that

$$\Delta' = F(\Delta + \Delta^f) - F(\Delta^f) \geq (1 - \eta^q)\gamma\Delta \tag{25}$$

Suppose that the Uniswap pool's initial reserves are (R, R') with an initial spot price of $p_0 = \frac{R'}{R}$ Then we can write a price condition akin to that of §2.2 as

$$\begin{aligned} g(\Delta) - g(0) = \frac{R' - \Delta'}{R + \Delta} - \frac{R'}{R} &= \frac{p_0 R - \Delta'}{R + \Delta} \\ &\leq \frac{p_0 R + (\eta^q - 1)\gamma\Delta}{R + \Delta} = \frac{p_0 R - \gamma\Delta}{R + \Delta} + \frac{\eta^q \gamma\Delta}{R + \Delta} \end{aligned}$$

where the inequality uses (25). Now we have

$$\frac{g(\Delta) - g(0)}{g(\Delta)} = \frac{p_0 R - \gamma\Delta}{R' - \Delta'} + \frac{\eta^q\gamma\Delta}{R' - \Delta'} \leq \frac{p_0 R - \xi\Delta}{R' - (1 - \eta^q)\xi\Delta} + \frac{\eta^q\xi\Delta}{R' - (\eta^q - 1)\xi\Delta}$$

The common denominator can be expanded as

$$\frac{1}{R' - (1-\eta^q)\xi\Delta} = \frac{1}{R'}\sum_{n=0}^{\infty}\left(\frac{(1-\eta^q)\xi\Delta^2}{R'}\right)^n \leq \frac{c(1-\eta^q)\xi\Delta^2}{R'^2}$$

By setting η to this quantity we match the bound from §2.2.

C Bounds on Δ^{sand}

C.1 Upper Bound (Claim 1)

Note that by construction, $G(0) = 0$ [AAE+21, §4.1]. These bounds have κ and μ in units of price. That is, the linear lower and upper bounds on the quantity output by a trade of size Δ implies a maximum and minimum price impact (lower and upper bounds on $g(\cdot)$).

We first bound the price impact of a cumulative trade $(\Delta^{\text{sand}}, (\Delta, \eta), \Delta^{\text{sand}'})$. To do this, we first assume the sandwich attack is optimal (making the output quantity the user gets tight with the specified slippage):

$$G(\Delta^{\text{sand}} + \Delta) - G(\Delta^{\text{sand}}) = (1-\eta)G(\Delta) \tag{26}$$

If we force the lower bound implied by (9) to be greater than the upper bound in equation (26), we have

$$G(\Delta + \Delta^{\text{sand}}) - G(\Delta^{\text{sand}}) \geq \kappa(\Delta + \Delta^{\text{sand}}) - \mu\Delta^{\text{sand}} \geq (1-\eta)\mu\Delta \geq (1-\eta)G(\Delta)$$

Rearranging the middle two terms gives us:

$$(\kappa - \mu)\Delta^{\text{sand}} \geq (1-\eta)\mu\Delta - \kappa\Delta = (\mu - \kappa)\Delta - \eta\mu\Delta$$

and dividing by $\kappa - \mu \leq 0$ we have:

$$\Delta^{\text{sand}} \leq \left(\frac{\eta\mu}{\mu - \kappa} - 1\right)\Delta \tag{27}$$

which provides an upper bound on Δ^{sand} as a function of the slippage and curvature. The bracketed term is positive when $\eta \geq 1 - \frac{\kappa}{\mu}$ which means that the slippage limit set by the user is larger than inverse of the curvature ratio. The smaller η is, i.e. the smaller a discount the user is willing to accept on the minimum output quantity they receive, the smaller the upper bound on the sandwich size will be.

C.2 Lower Bound (Claim 2)

Suppose we have a (μ, κ)-smooth forward exchange function $G(\Delta)$ whose derivative $g(\delta) = G'(\Delta)$ is β-liquid, *e.g.* $g(-\delta) - g(0) \geq \beta\Delta$. To construct a lower bound on Δ^{sand}, we will

bound the left side of (6) below and the right side above. We construct a quadratic lower bound for $G(\Delta)$ using g:

$$G(\Delta) = \int_0^{\Delta} g(-t)dt \geq \int_0^{\Delta} \beta t + g(0)dt = \frac{\kappa \Delta^2}{2} + g(0)\Delta$$

Using this bound, we can lower bound the left side of (6) as

$$G(\Delta^{\text{sand}} + \Delta) - G(\Delta^{\text{sand}}) \geq \frac{\beta(\Delta + \Delta^{\text{sand}})^2}{2} + g(0)\Delta - \mu\Delta^{\text{sand}}$$

Combining this with the bound $(1+\eta)G(\Delta) \leq (1+\eta)\mu\Delta$ gives the condition

$$(1+\eta)\mu\Delta \leq \frac{\beta(\Delta + \Delta^{\text{sand}})^2}{2} + g(0)\Delta - \mu\Delta^{\text{sand}}$$

Solving this quadratic equation in Δ^{sand} for when there is equality yields two roots

$$r_{\pm} = \left(\frac{\mu}{\beta} - \Delta\right)\left[1 \pm \sqrt{1 + \beta\Delta\left(1 + \frac{\eta\mu + p_0}{\mu - \kappa\Delta}\right)}\right]$$

Provided that $\Delta < \frac{\mu}{\beta}$ (note that μ has units of price where β has units of price over quantity), then $r_+ > 0$ and we have the condition

$$\Delta^{\text{sand}} \geq r_+ = \left(\frac{\mu}{\beta} - \Delta\right)\left[1 + \sqrt{1 + \beta\Delta\left(1 + \frac{\eta\mu + p_0}{\mu - \kappa\Delta}\right)}\right] > \left(\frac{\mu}{\beta} - \Delta\right)\gamma \tag{28}$$

where $\gamma > 1$.

D Bounds for $\Delta^{\text{sand}'}$

Once again using the linear lower and upper bounds on $G(\cdot)$ (and $G^{-1}(\cdot)$), we have:

$$\begin{aligned} G^{-1}(G(\Delta^{\text{sand}} + \Delta) - G(\Delta^{\text{sand}})) &\leq \frac{1}{\kappa}\left(\mu(\Delta^{\text{sand}} + \Delta) - \kappa\Delta^{\text{sand}}\right) = \frac{1}{\kappa}\left((\mu - \kappa)\Delta^{\text{sand}} + \mu\Delta\right) \\ &\leq \frac{1}{\kappa}\left((\mu - \kappa)\left(\frac{\eta\mu}{\mu - \kappa} - 1\right) + \mu\right)\Delta = \left(\frac{\eta\mu}{\kappa} + 1\right)\Delta \end{aligned} \tag{29}$$

Similarly, we have a matching lower bound

$$\begin{aligned} G^{-1}(G(\Delta^{\text{sand}} + \Delta) - G(\Delta^{\text{sand}})) &\geq \frac{1}{\mu}\left(\kappa(\Delta^{\text{sand}} + \Delta) - \mu\Delta^{\text{sand}}\right) = \frac{1}{\mu}\left(-(\mu - \kappa)\Delta^{\text{sand}} + \mu\Delta\right) \\ &\geq \frac{1}{\mu}\left((\mu - \kappa)\left(1 - \frac{\eta\mu}{\mu - \kappa}\right)\Delta + \mu\Delta\right) \\ &= \left(\left(1 - \frac{\kappa}{\mu}\right) + (1 - \eta)\right)\Delta \end{aligned} \tag{30}$$

These bounds furnish us bounds for $\Delta^{\text{sand}'}$:

$$\begin{aligned}\Delta^{\text{sand}'} &= \Delta^{\text{sand}} + \Delta - G^{-1}(G(\Delta^{\text{sand}} + \Delta) - G(\Delta^{\text{sand}})) \\ &\leq \frac{\eta\mu}{\mu - \kappa}\Delta - \left(\left(1 - \frac{\kappa}{\mu}\right) + (1 - \eta)\right)\Delta \\ &= \left(\eta\left(1 + \frac{\mu}{\mu - \kappa}\right) - \left(2 - \frac{\kappa}{\mu}\right)\right)\Delta\end{aligned}$$

and

$$\begin{aligned}\Delta^{\text{sand}'} &= \Delta^{\text{sand}} + \Delta - G^{-1}(G(\Delta^{\text{sand}} + \Delta) - G(\Delta^{\text{sand}})) \\ &\geq \left(\frac{\mu}{\beta} - \Delta\right)\gamma + \Delta - \left(\frac{\eta\mu}{\kappa} + 1\right)\Delta \qquad (31) \\ &\geq \frac{\mu\gamma}{\beta} - \Delta\left(\gamma + \frac{\eta\mu}{\kappa}\right) \qquad (32)\end{aligned}$$

The proof of claim 5 can be see simply by using the inequality

$$\Delta^{\text{sand}'} - \Delta^{\text{sand}} \geq \left(\eta\left(1 + \frac{\mu}{\mu - \kappa}\right) - \left(2 - \frac{\kappa}{\mu}\right)\right)\Delta + \Delta\gamma - \frac{\mu\gamma}{\beta}$$

E Bounds for $\mathsf{CoF}(T_n)$

E.1 Statements of Lemmas

The first two lemmas provide upper and lower bounds on ${\Delta^{\text{sand}}}_i$ using the partial trade drifts $\tilde{u}_{i-1}$ and the trade Δ_i:

Lemma 1.

$${\Delta^{\text{sand}}}_i \leq \tilde{u}_{i-1} + \left(\frac{\eta\mu}{\mu - \kappa} - 1\right)\Delta_i$$

Lemma 2.

$${\Delta^{\text{sand}}}_i \geq \tilde{u}_{i-1} + \left(\frac{\eta\kappa}{\mu - \kappa} - 1\right)\Delta_i$$

The next two lemmas provide upper and lower bounds on ${\Delta^{\text{sand}'}}_i$ in terms of $\tilde{u}_{i-1}$ and Δ_i. We note here that these bounds differ from the linear upper bounds in Lemmas 1 and 2 as they use a quadratic bound on ${\Delta^{\text{sand}}}_i$ to bound ${\Delta^{\text{sand}'}}_i$. These stronger conditions are necessary to prove Proposition 2. We also note that the constants γ and ν in the lemmas below are related to solutions of the aforementioned quadratic equation. In particular, $\nu < 0$ and $\gamma > 0$.

Lemma 3.

$$\Delta^{\text{sand}'}{}_i \geq -\left(\frac{\eta\mu}{\mu-\kappa}+\frac{\kappa}{\mu}(\gamma+1)-1\right)\Delta_i+\frac{\kappa}{\mu}\left(\frac{\mu}{\beta}-g(0)\right)-\left(1+\frac{\kappa\gamma}{\mu}\right)\tilde{u}_{i-1}$$

Lemma 4.

$$\Delta^{\text{sand}'}{}_i \leq -\left(\frac{\eta\kappa}{\mu-\kappa}+\frac{\mu}{\kappa}(\nu+1)-1\right)\Delta_i+\frac{\mu}{\kappa}\left(\frac{\kappa}{\beta}-g(0)\right)-\left(1+\frac{\mu\nu}{\kappa}\right)\tilde{u}_{i-1}$$

We now use these lemmas to derive lower and upper bounds on $\mathsf{PNL}_i = \Delta^{\text{sand}'}{}_i - \Delta^{\text{sand}}{}_i$. We first recall discrete Grönwall inequalities from [Aga00]:

Proposition 7 (Thm 4.1.1, [Aga00])**.** *Suppose that $u_k, q_k, f_k \in \mathbf{R}$ are non-negative sequences and $p_k \in \mathbf{R}$ is a sequence that collectively satisfy:*

$$u_k \leq p_k + q_k \sum_{\ell=a}^{k-1} f_\ell u_\ell$$

Then for all $k \geq a$, we have:

$$u_k \leq p_k + q_k \sum_{\ell=a}^{k-1} p_\ell f_\ell \left(\prod_{i=\ell+1}^{k-1}(1+q_i f_i)\right)$$

Proposition 8 (Thm 4.1.9, [Aga00])**.** *Let for all $k, r \in \mathbb{N}$ such that $k \leq r$ the following inequality be satisied:*

$$u_r \geq u_k - q_r \sum_{\ell=k+1}^{r} f_\ell u_\ell$$

where u_k is not necessarily nonnegative. Then, for all $k, r \in \mathbb{N}$, $k \leq r$

$$u_r \geq u_k \prod_{\ell=k+1}^{r}(1+q_r f_\ell)^{-1}$$

We now provide upper and lower bounds on $\mathsf{PNL}_i = \Delta^{\text{sand}'}{}_i - \Delta^{\text{sand}}{}_i$, which is the sandwich profit extracted by the sandwicher over the i-th trade, Δ_i. We make use of the above Grönwall inequalities to unroll the recursion and get bounds on PNL_i only in terms of Δ_j for $j = 1, \ldots, i$. Combining the bounds from Lemmas 4 and 2 we have the following upper bound:

Lemma 5. *We can upper bound $\Delta^{\text{sand}'}{}_i - \Delta_i - \Delta^{\text{sand}}{}_i$ as:*

$$\begin{aligned}\Delta^{\text{sand}'}{}_i - \Delta_i - \Delta^{\text{sand}}{}_i \leq &\left(-1-\frac{\mu}{\kappa}(\nu+1)\right)\Delta_i+\frac{\mu}{\kappa}\left(\frac{\kappa}{\beta}-g(0)\right)\\ &+\left(2+\frac{\mu\nu}{\kappa}\right)\left(\sum_{j=1}^{i-1}\Delta^{\text{sand}'}{}_i - \Delta_i - \Delta^{\text{sand}}{}_i\right)\end{aligned}$$

Now, using Proposition 7 we have:

Lemma 6. *For* $p_i = \left(-1 - \frac{\mu}{\kappa}(\nu+1)\right)\Delta_i + \frac{\mu}{\kappa}\left(\frac{\kappa}{\beta} - g(0)\right)$, *the sandwich profit* $\mathsf{PNL}_i = {\Delta^{\text{sand}}}'_i - \Delta^{\text{sand}}{}_i$ *can be upper bounded:*

$$ {\Delta^{\text{sand}}}'_i - \Delta^{\text{sand}}{}_i \leq p_i + \Delta_i + \left(2 + \frac{\mu\nu}{\kappa}\right)\sum_{\ell=1}^{i-1} p_\ell \left(3 + \frac{\mu\nu}{\kappa}\right)^{i-\ell-1} $$

We have similar lower bounds for PNL_i using the following lemmas:

Lemma 7. *We can lower bound* ${\Delta^{\text{sand}}}'_i - \Delta_i - \Delta^{\text{sand}}{}_i$ *as:*

$$ {\Delta^{\text{sand}}}'_i - \Delta_i - \Delta^{\text{sand}}{}_i \geq \left(-1 + \frac{\kappa}{\mu}(\gamma+1)\right)\Delta_i + \frac{\kappa}{\mu}\left(\frac{\mu}{\beta} - g(0)\right) \tag{33} $$

$$ + \left(2 + \frac{\kappa\gamma}{\mu}\right)\left(\sum_{j=1}^{i-1} {\Delta^{\text{sand}}}'_j - \Delta_j - \Delta^{\text{sand}}{}_j\right) \tag{34} $$

E.2 Proofs of Lemmas

Lemma 1.

$$ \Delta^{\text{sand}}{}_i \leq \tilde{u}_{i-1} + \left(\frac{\eta\mu}{\mu - \kappa} - 1\right)\Delta_i $$

Proof. We first note that by definition, $\Delta^{\text{sand}}{}_i$ satisfies the equation:

$$ G\left(\Delta^{\text{sand}}{}_i + \Delta_i + \sum_{j=1}^{i-1} \Delta^{\text{sand}}{}_i + \Delta_i - {\Delta^{\text{sand}}}'_i\right) $$
$$ - G\left(\Delta^{\text{sand}}{}_i + \sum_{j=1}^{i-1} \Delta^{\text{sand}}{}_i + \Delta_i - {\Delta^{\text{sand}}}'_i\right) = (1-\eta)G(\Delta_i) $$

Using the bounds $G(\Delta_i) \leq \mu\Delta_i$ and $G(\Delta_i) \geq \kappa\Delta_i$, we lower bound the left hand side and upper bound the right hand side to get:

$$ (\kappa - \mu)\,\Delta^{\text{sand}}{}_i + \kappa\Delta_i + (\kappa - \mu)\left(\sum_{j=1}^{i-1} \Delta^{\text{sand}}{}_j + \Delta_j - {\Delta^{\text{sand}}}'_j\right) \geq (1-\eta)\mu\Delta_i $$

Rearranging and dividing by $\kappa - \mu$, we have:

$$ \Delta^{\text{sand}}{}_i \leq \sum_{j=1}^{i-1} \Delta^{\text{sand}}{}_j + \Delta_j - {\Delta^{\text{sand}}}'_j + \left(1 - \frac{\eta\mu}{\mu - \kappa}\right)\Delta_i $$
$$ = \tilde{\mu}_{i-1} + \left(\frac{\eta\mu}{\mu - \kappa} - 1\right)\Delta_i $$

□

Lemma 2.

$$\Delta^{\text{sand}}{}_i \geq \tilde{u}_{i-1} + \left(1 - \frac{\eta\kappa}{\mu - \kappa}\right)\Delta_i$$

Proof. Once again, we begin with the definition of $\Delta^{\text{sand}}{}_i$:

$$G\left(\Delta^{\text{sand}}{}_i + \Delta_i + \sum_{j=1}^{i-1} \Delta^{\text{sand}}{}_i + \Delta_i - \Delta^{\text{sand}'}_i\right)$$
$$- G\left(\Delta^{\text{sand}}{}_i + \sum_{j=1}^{i-1} \Delta^{\text{sand}}{}_i + \Delta_i - \Delta^{\text{sand}'}_i\right) = (1-\eta)G(\Delta_i)$$

We now upper bound the left hand side and lower bound the right hand side to get:

$$(\mu - \kappa)\,\Delta^{\text{sand}}{}_i + \mu\Delta_i + (\mu - \kappa)\left(\sum_{j=1}^{i-1} \Delta^{\text{sand}}{}_j + \Delta_j - \Delta^{\text{sand}'}_j\right) \geq (1-\eta)\kappa\Delta_i$$

Rearranging and dividing by $\mu - \kappa$ we have:

$$\begin{aligned}\Delta^{\text{sand}}{}_i &\geq \sum_{j=1}^{i-1} \Delta^{\text{sand}}{}_j + \Delta_j - \Delta^{\text{sand}'}_j + \left(1 - \frac{\eta\kappa}{\mu - \kappa}\right)\Delta_i \\ &= \tilde{u}_{i-1} + \left(1 - \frac{\eta\kappa}{\mu - \kappa}\right)\Delta_i\end{aligned}$$

□

Lemma 3.

$$\Delta^{\text{sand}'}_i \geq -\left(\frac{\eta\mu}{\mu - \kappa} + \frac{\kappa}{\mu}(\gamma + 1) - 1\right)\Delta_i + \frac{\kappa}{\mu}\left(\frac{\mu}{\beta} - g(0)\right) - \left(1 + \frac{\kappa\gamma}{\mu}\right)\tilde{u}_{i-1}$$

Proof. Note that by definition, $\Delta^{\text{sand}'}_i$ satisfies the equation:

$$\Delta^{\text{sand}'}_i = \Delta^{\text{sand}} + \Delta - G^{-1}\left(G\left(\Delta^{\text{sand}}{}_i + \Delta_i + \sum_{j=1}^{i-1} \Delta^{\text{sand}}{}_j + \Delta_j - \Delta^{\text{sand}'}_j\right)\right.$$
$$\left. - G\left(\Delta^{\text{sand}}{}_i + \sum_{j=1}^{i-1} \Delta^{\text{sand}}{}_j + \Delta_j - \Delta^{\text{sand}'}_j\right)\right.$$

We first get a quadratic lower bound for ${\Delta^{\text{sand}}}_i$ in Δ_i and use it to lower bound ${\Delta^{\text{sand}'}}_i$:

$$\begin{aligned}(1+\eta)\mu\Delta_i \leq & \frac{\beta({\Delta^{\text{sand}}}_i + \Delta_i + \sum_{j=1}^{i-1} {\Delta^{\text{sand}}}_j + \Delta_j + {\Delta^{\text{sand}'}}_j)^2}{2} \\ & + g(0)\left({\Delta^{\text{sand}}}_i + \Delta_i + \sum_{j=1}^{i-1} {\Delta^{\text{sand}}}_j + \Delta_j + {\Delta^{\text{sand}'}}_j\right) \\ & - \mu\left({\Delta^{\text{sand}}}_i + \sum_{j=1}^{i-1} {\Delta^{\text{sand}}}_j + \Delta_j - {\Delta^{\text{sand}'}}_j\right)\end{aligned}$$

We now solve this quadratic equation in ${\Delta^{\text{sand}}}_i$ for when there is equality. In particular, we solve:

$$\begin{aligned}0 = & \underbrace{\frac{\beta}{2}}_{a} {\Delta^{\text{sand}}}_i^2 + \underbrace{(\beta\left(\frac{\Delta_i + \tilde{u}_{i-1} + g(0)}{2}\right) - \mu)}_{b} {\Delta^{\text{sand}}}_i \\ & + \underbrace{\frac{\beta}{2}(\Delta_i + \tilde{u}_{i-1})^2 + (g(0) - (1+\eta)\mu)\Delta_i + (g(0) - \mu)\tilde{u}_{i-1}}_{c}\end{aligned}$$

which gives us the roots:

$$r_\pm = \left(\frac{\mu}{\beta} - \Delta_i - \tilde{u}_{i-1} - g(0)\right) \pm \sqrt{(\Delta_i + \tilde{u}_{i-1} + g(0) - \mu)^2 - 2\beta c}$$

We can now take the positive root, $r_+ > 0$ and we have the condition:

$${\Delta^{\text{sand}}}_i \geq r_+ > \left(\frac{\mu}{\beta} - \Delta_i - \tilde{u}_{i-1} - g(0)\right)\gamma$$

for $\gamma > 0$. We can now use the definition of G^{-1} to construct a lower bound for ${\Delta^{\text{sand}'}}_i$:

$$\begin{aligned}{\Delta^{\text{sand}'}}_i &\geq {\Delta^{\text{sand}}}_i + \Delta_i - \frac{1}{\kappa}\left(\mu\left({\Delta^{\text{sand}}}_i + \Delta_i + \tilde{u}_{i-1}\right) - \kappa\left({\Delta^{\text{sand}}}_i + \tilde{u}_{i-1}\right)\right) \\ &\geq {\Delta^{\text{sand}}}_i + \Delta_i - \frac{1}{\kappa}\left(\mu\left(\frac{\mu}{\beta} - \Delta_i - \tilde{u}_{i-1} - g(0)\right)\gamma - \kappa\left(\tilde{u}_{i-1} + \left(1 - \frac{\eta\kappa}{\mu - \kappa}\right)\Delta_i\right) + \kappa\Delta_i\right) \\ &= {\Delta^{\text{sand}}}_i - \frac{1}{\kappa}\left(\left(-\kappa + \kappa\left(1 - \frac{\eta\mu}{\mu - \kappa}\right)\right)\Delta_i - \mu(\gamma+1)\Delta_i + \kappa\left(\frac{\mu}{\beta} - g(0)\right) - (\kappa + \mu)\tilde{u}_{i-1}\right) \\ &= {\Delta^{\text{sand}}}_i + \left(2 - \frac{\eta\mu}{\mu - \kappa} + \frac{\mu}{\kappa}(\gamma+1)\right)\Delta_i + \frac{\kappa}{\mu}\left(\frac{\mu}{\beta} - g(0)\right) - \left(1 + \frac{\mu}{\kappa}\right)\tilde{u}_{i-1}\end{aligned}$$

□

Lemma 4.

$${\Delta^{\text{sand}'}}_i \leq \frac{2(\mu - \kappa)}{\kappa}\tilde{u}_{i-1} + \left(\frac{\mu - \kappa}{\kappa} + \frac{(1-\eta)\mu}{\kappa}\right)\Delta_i \tag{35}$$

Proof. Once again, we begin with the defining relation of ${\Delta^{\text{sand}}}_i$. That is:

$$\begin{aligned}{\Delta^{\text{sand}'}}_i = {\Delta^{\text{sand}}}_i + \Delta_i - G^{-1}\left(G\left({\Delta^{\text{sand}}}_i + \Delta_i + \sum_{j=1}^{i-1} {\Delta^{\text{sand}}}_j + \Delta_j - {\Delta^{\text{sand}'}}_j\right)\right. \\ \left. - G\left({\Delta^{\text{sand}}}_i + \sum_{j=1}^{i-1} {\Delta^{\text{sand}}}_j + \Delta_j - {\Delta^{\text{sand}'}}_j\right)\right.\end{aligned}$$

Now, we use the linear upper bound on ${\Delta^{\text{sand}}}_i$ from Lemma 1 and the curvature of G and G^{-1} to upper bound ${\Delta^{\text{sand}'}}_i$ as:

$$\begin{aligned}{\Delta^{\text{sand}'}}_i &\leq {\Delta^{\text{sand}}}_i + \Delta_i - \frac{1}{\mu}\left((\kappa-\mu){\Delta^{\text{sand}}}_i + \kappa\Delta_i + (\kappa-\mu)\tilde{u}_{i-1}\right) \\ &\leq {\Delta^{\text{sand}}}_i + \Delta_i - \frac{\kappa-\mu}{\mu}\left(\tilde{u}_{i-1} + \left(\frac{\eta\mu}{\mu-\kappa} - 1\right)\Delta_i\right) + \frac{\mu}{\kappa}\Delta_i + \frac{\mu-\kappa}{\kappa}\tilde{u}_{i-1} \\ &\leq \left(1 - \frac{\kappa-\mu}{\mu}\right)\left(\frac{\eta\mu}{\mu-\kappa} - 1\right)\Delta_i + \left(\frac{\mu}{\kappa} + 1\right)\Delta_i + \frac{2(\mu-\kappa)}{\mu}\tilde{u}_{i-1} \\ &= \frac{2(\mu-\kappa)}{\kappa}\tilde{u}_{i-1} + \left(\left(1 - \frac{\kappa-\mu}{\mu}\right)\left(\frac{\eta\mu}{\mu-\kappa} - 1\right) + \frac{\mu}{\kappa} + 1\right)\Delta_i\end{aligned}$$

□

We also prove a quadratic upper bound on ${\Delta^{\text{sand}'}}_i$:

Lemma 5.

$${\Delta^{\text{sand}'}}_i \leq -\left(\frac{\eta\kappa}{\mu-\kappa} + \frac{\mu}{\kappa}(\nu+1) - 1\right)\Delta_i + \frac{\mu}{\kappa}\left(\frac{\kappa}{\beta} - g(0)\right) - \left(1 + \frac{\mu\nu}{\kappa}\right)\tilde{u}_{i-1}$$

Proof.

$$\begin{aligned}{\Delta^{\text{sand}'}}_i = {\Delta^{\text{sand}}}_i + \Delta_i - G^{-1}\left(G\left({\Delta^{\text{sand}}}_i + \Delta_i + \sum_{j=1}^{i-1} {\Delta^{\text{sand}}}_j + \Delta_j - {\Delta^{\text{sand}'}}_j\right)\right. \\ \left. - G\left({\Delta^{\text{sand}}}_i + \sum_{j=1}^{i-1} {\Delta^{\text{sand}}}_j + \Delta_j - {\Delta^{\text{sand}'}}_j\right)\right.\end{aligned}$$

We have:

$$\begin{aligned}(1+\eta)\kappa\Delta_i \geq \frac{\beta({\Delta^{\text{sand}}}_i + \Delta_i + \tilde{u}_{i-1})^2}{2} \\ + g(0)\left({\Delta^{\text{sand}}}_i + \Delta_i + \tilde{u}_{i-1}\right) \\ - \mu\left({\Delta^{\text{sand}}}_i + \tilde{u}_{i-1}\right)\end{aligned}$$

We now solve the quadratic equation:

$$0 = \underbrace{\frac{\beta}{2}}_{a} {\Delta^{\text{sand}}}_i^2 + \underbrace{\left(\beta\left(\frac{\Delta_i + \tilde{u}_{i-1} + g(0)}{2}\right)\right)}_{b} {\Delta^{\text{sand}}}_i + \underbrace{\frac{\beta}{2}\left(\Delta_i + \tilde{u}_{i-1}\right)^2 + (g(0) - (1+\eta)\kappa)\Delta_i + (g(0) - \kappa)\tilde{u}_{i-1}}_{c}$$

Which gives us the roots:

$$r_\pm = \left(\frac{\kappa}{\beta} - \Delta_i - \tilde{u}_{i-1} - g(0)\right) \pm \sqrt{(\Delta_i + \tilde{u}_{i-1} + g(0) - \kappa)^2 - 2\beta c}$$

Therefore, we can take the negative root and upper bound:

$${\Delta^{\text{sand}}}_i \leq \left(\frac{\kappa}{\beta} - \Delta_i - \tilde{u}_{i-1} - g(0)\right)\nu$$

for some $\nu < 0$. We can now use the definition of G^{-1} to construct an upper bound for ${\Delta^{\text{sand}}}'$.

$$\begin{aligned}
{\Delta^{\text{sand}}}'_i &\leq {\Delta^{\text{sand}}}_i + \Delta_i - \frac{1}{\mu}\left(\kappa({\Delta^{\text{sand}}}_i + \Delta_i + \tilde{u}_{i-1}) - \mu({\Delta^{\text{sand}}}_i + \tilde{u}_{i-1})\right) \\
&\leq {\Delta^{\text{sand}}}_i + \Delta_i - \frac{1}{\mu}\left(\kappa\left(\frac{\kappa}{\beta} - \Delta_i - \tilde{u}_{i-1} - g(0)\right)\nu - \mu\left(\tilde{u}_{i-1} + \left(\frac{\eta\mu}{\mu - \kappa} - 1\right)\Delta_i\right) + \mu\Delta_i\right) \\
&= {\Delta^{\text{sand}}}_i + \Delta_i - \frac{1}{\mu}\left(-\kappa\left(\frac{\eta\kappa}{\mu - \kappa} - 1\right)\Delta_i - \kappa(\nu+1)\Delta_i + \mu\left(\frac{\kappa}{\beta} - g(0)\right) - (\kappa + \nu\mu)\tilde{u}_{i-1}\right) \\
&= -\left(\frac{\eta\kappa}{\mu - \kappa} + \frac{\mu}{\kappa}(\nu+1) - 1\right)\Delta_i + \frac{\mu}{\kappa}\left(\frac{\kappa}{\beta} - g(0)\right) - \left(1 + \frac{\mu\nu}{\kappa}\right)\tilde{u}_{i-1}
\end{aligned}$$

□

Lemma 6. *We can upper bound* ${\Delta^{\text{sand}}}'_i - \Delta_i - {\Delta^{\text{sand}}}_i$ *as:*

$${\Delta^{\text{sand}}}'_i - \Delta_i - {\Delta^{\text{sand}}}_i \leq \left(-1 - \frac{\mu}{\kappa}(\nu+1)\right)\Delta_i + \frac{\mu}{\kappa}\left(\frac{\kappa}{\beta} - g(0)\right) \tag{36}$$

$$+ \left(2 + \frac{\mu\nu}{\kappa}\right)\left(\sum_{j=1}^{i-1} {\Delta^{\text{sand}}}'_i - \Delta_i - {\Delta^{\text{sand}}}_i\right) \tag{37}$$

Proof. We use the lower bound on Δ^{sand} and upper bound on ${\Delta^{\text{sand}'}}_i$ derived in Lemmas 2 and 4 respectively to bound ${\Delta^{\text{sand}'}}_i - {\Delta^{\text{sand}}}_i$. We have:

$$\begin{aligned}
{\Delta^{\text{sand}'}}_i - {\Delta^{\text{sand}}}_i &\leq -\left(\frac{\eta\kappa}{\mu-\kappa} + \frac{\mu}{\kappa}(\nu+1) - 1\right)\Delta_i + \frac{\mu}{\kappa}\left(\frac{\kappa}{\beta} - g(0)\right) - \left(1 + \frac{\mu\nu}{\kappa}\right)\tilde{u}_{i-1} \\
&\quad - \tilde{u}_{i-1} + \left(\frac{\eta\kappa}{\mu-\kappa} - 1\right)\Delta_i \\
&= \frac{\mu}{\kappa}(\nu+1)\Delta_i + \frac{\mu}{\kappa}\left(\frac{\kappa}{\beta} - g(0)\right) - \left(2 + \frac{\mu\nu}{\kappa}\right)\tilde{u}_{i-1}
\end{aligned}$$

and adding $-\Delta_i$ to both sides gives us:

$$\begin{aligned}
{\Delta^{\text{sand}'}}_i - \Delta_i - {\Delta^{\text{sand}}}_i &\leq \left(-1 - \frac{\mu}{\kappa}(\nu+1)\right)\Delta_i + \frac{\mu}{\kappa}\left(\frac{\kappa}{\beta} - g(0)\right) - \left(2 + \frac{\mu\nu}{\kappa}\right)\tilde{u}_{i-1} \\
&= \left(-1 - \frac{\mu}{\kappa}(\nu+1)\right)\Delta_i + \frac{\mu}{\kappa}\left(\frac{\kappa}{\beta} - g(0)\right) \\
&\quad + \left(2 + \frac{\mu\nu}{\kappa}\right)\left(\sum_{j=1}^{i-1} {\Delta^{\text{sand}'}}_j - \Delta_j - {\Delta^{\text{sand}}}_j\right)
\end{aligned}$$

□

Lemma 7. *For* $p_i = \left(-1 - \frac{\mu}{\kappa}(\nu+1)\right)\Delta_i + \frac{\mu}{\kappa}\left(\frac{\kappa}{\beta} - g(0)\right)$*, the sandwich profit* $\mathsf{PNL}_i = {\Delta^{\text{sand}'}}_i - {\Delta^{\text{sand}}}_i$ *can be upper bounded:*

$${\Delta^{\text{sand}'}}_i - {\Delta^{\text{sand}}}_i \leq \left(-\frac{\mu}{\kappa}(\nu+1)\right)\Delta_i + \frac{\mu}{\kappa}\left(\frac{\kappa}{\beta} - g(0)\right) + \left(2 + \frac{\mu\nu}{\kappa}\right)\sum_{\ell=1}^{i-1} p_\ell \left(3 + \frac{\mu\nu}{\kappa}\right)^{i-\ell-1} \tag{38}$$

Proof. We use the discrete Grönwall inequality from Proposition 7 on the inequality derived in Lemma 5. We have:

$${\Delta^{\text{sand}'}}_i - \Delta_i - {\Delta^{\text{sand}}}_i \leq \left(-1 - \frac{\mu}{\kappa}(\nu+1)\right)\Delta_i + \frac{\mu}{\kappa}\left(\frac{\kappa}{\beta} - g(0)\right) \tag{39}$$

$$+ \left(2 + \frac{\mu\nu}{\kappa}\right)\left(\sum_{j=1}^{i-1} {\Delta^{\text{sand}'}}_j - \Delta_j - {\Delta^{\text{sand}}}_j\right) \tag{40}$$

Defining $p_i = \left(-1 - \frac{\mu}{\kappa}(\nu+1)\right)\Delta_i + \frac{\mu}{\kappa}\left(\frac{\kappa}{\beta} - g(0)\right)$, $q_i = \left(2 + \frac{\mu\nu}{\kappa}\right)$, and $f_\ell = 1$ for all ℓ in the

Grönwall inequality, we have:

$$\begin{aligned}
{\Delta^{\text{sand}'}}_i - \Delta_i - {\Delta^{\text{sand}}}_i &\leq p_i + q_i \sum_{\ell=1}^{i-1} p_\ell f_\ell \left(\prod_{k=\ell+1}^{i-1} (1 + q_k f_k) \right) \\
&= p_i + \left(2 + \frac{\mu\nu}{\kappa}\right) \sum_{\ell=1}^{i-1} p_\ell \prod_{k=\ell+1}^{i-1} (1 + q_k) \\
&= p_i + \left(2 + \frac{\mu\nu}{\kappa}\right) \sum_{\ell=1}^{i-1} p_\ell \left(1 + 2 + \frac{\mu\nu}{\kappa}\right)^{i-\ell-1} \\
&= p_i + \left(2 + \frac{\mu\nu}{\kappa}\right) \sum_{\ell=1}^{i-1} p_\ell \left(3 + \frac{\mu\nu}{\kappa}\right)^{i-\ell-1}
\end{aligned}$$

Adding Δ_i to both sides:

$${\Delta^{\text{sand}'}}_i - {\Delta^{\text{sand}}}_i \leq p_i + \Delta_i + \left(2 + \frac{\mu\nu}{\kappa}\right) \sum_{\ell=1}^{i-1} p_\ell \left(3 + \frac{\mu\nu}{\kappa}\right)^{i-\ell-1}$$

and combining terms in p_i with Δ_i we have:

$${\Delta^{\text{sand}'}}_i - {\Delta^{\text{sand}}}_i \leq \left(-\frac{\mu}{\kappa}(\nu + 1)\right) \Delta_i + \frac{\mu}{\kappa} \left(\frac{\kappa}{\beta} - g(0)\right) + \left(2 + \frac{\mu\nu}{\kappa}\right) \sum_{\ell=1}^{i-1} p_\ell \left(3 + \frac{\mu\nu}{\kappa}\right)^{i-\ell-1}$$

□

Lemma 8. *We can lower bound* ${\Delta^{\text{sand}'}}_i - \Delta_i - {\Delta^{\text{sand}}}_i$ *as:*

$$\begin{aligned}
{\Delta^{\text{sand}'}}_i - \Delta_i - {\Delta^{\text{sand}}}_i \geq \left(-1 + \frac{\kappa}{\mu}(\gamma + 1)\right) \Delta_i + \frac{\kappa}{\mu} \left(\frac{\mu}{\beta} - g(0)\right) \qquad (41) \\
+ \left(2 + \frac{\kappa\gamma}{\mu}\right) \left(\sum_{j=1}^{i-1} {\Delta^{\text{sand}'}}_j - \Delta_j - {\Delta^{\text{sand}}}_j \right) \qquad (42)
\end{aligned}$$

Proof. We use the upper bound on ${\Delta^{\text{sand}}}_i$ and lower bound on ${\Delta^{\text{sand}'}}_i$ derived in Lemmas 1 and 3 respectively to bound ${\Delta^{\text{sand}'}}_i - {\Delta^{\text{sand}}}_i$. We have:

$$\begin{aligned}
{\Delta^{\text{sand}'}}_i - {\Delta^{\text{sand}}}_i &\geq -\left(\frac{\eta\mu}{\mu - \kappa} + \frac{\kappa}{\mu}(\gamma + 1) - 1\right) \Delta_i + \frac{\kappa}{\mu} \left(\frac{\mu}{\beta} - g(0)\right) - \left(1 + \frac{\kappa\gamma}{\mu}\right) \tilde{u}_{i-1} \\
&\quad - \tilde{u}_{i-1} - \left(1 - \frac{\eta\mu}{\mu - \kappa}\right) \Delta_i \\
&= \frac{\kappa}{\mu}(\gamma + 1)\Delta_i + \frac{\kappa}{\mu} \left(\frac{\mu}{\beta} - g(0)\right) - \left(2 + \frac{\kappa\gamma}{\mu}\right) \tilde{u}_{i-1}
\end{aligned}$$

and adding $-\Delta_i$ to both sides gives us:

$$\begin{aligned}
{\Delta^{\text{sand}'}}_i - \Delta_i - {\Delta^{\text{sand}}}_i &\geq \left(-1\frac{\kappa}{\mu}(\gamma+1)\right)\Delta_i + \frac{\kappa}{\mu}\left(\frac{\mu}{\beta} - g(0)\right) - \left(2+\frac{\kappa\gamma}{\mu}\right)\tilde{u}_{i-1} \\
&= \left(-1+\frac{\kappa}{\mu}(\gamma+1)\right)\Delta_i + \frac{\kappa}{\mu}\left(\frac{\mu}{\beta} - g(0)\right) \\
&\quad + \left(2+\frac{\kappa\gamma}{\mu}\right)\left(\sum_{j=1}^{i-1} {\Delta^{\text{sand}'}}_j - \Delta_j - {\Delta^{\text{sand}}}_j\right)
\end{aligned}$$

□

Lemma 9. *The sandwich profit* $\mathsf{PNL}_i = {\Delta^{\text{sand}'}}_i - {\Delta^{\text{sand}}}_i$ *can be lower bounded:*

$$ {\Delta^{\text{sand}'}}_i - {\Delta^{\text{sand}}}_i \geq \Delta_i + \left(\frac{\mu}{\mu+\kappa\gamma}\right)^i \tag{43}$$

Proof. We use the discrete Grönwall inequality from Proposition 8 on the inequality derived in Lemma 7. We have from Lemma 7:

$$\begin{aligned}
{\Delta^{\text{sand}'}}_i - \Delta_i - {\Delta^{\text{sand}}}_i &\geq \left(-1+\frac{\kappa}{\mu}(\gamma+1)\right)\Delta_i + \frac{\kappa}{\mu}\left(\frac{\mu}{\beta} - g(0)\right) \\
&\quad + \left(2+\frac{\kappa\gamma}{\mu}\right)\left(\sum_{j=1}^{i-1} {\Delta^{\text{sand}'}}_j - \Delta_j - {\Delta^{\text{sand}}}_j\right)
\end{aligned}$$

Negating this inequality, we have:

$$\begin{aligned}
-{\Delta^{\text{sand}'}}_i + \Delta_i - {\Delta^{\text{sand}}}_i &\leq \left(1-\frac{\kappa}{\mu}(\gamma+1)\right)\Delta_i - \frac{\kappa}{\mu}\left(\frac{\mu}{\beta} - g(0)\right) \\
&\quad + \left(2+\frac{\kappa\gamma}{\mu}\right)\left(\sum_{j=1}^{i-1} -{\Delta^{\text{sand}'}}_j + \Delta_j + {\Delta^{\text{sand}}}_j\right)
\end{aligned}$$

Defining $m_i = \left(1-\frac{\kappa}{\mu}(\gamma+1)\right)\Delta_i - \frac{\kappa}{\mu}\left(\frac{\mu}{\beta} - g(0)\right)$, $q_i = \left(2+\frac{\kappa\gamma}{\mu}\right)$, and $f_\ell = 1$ for all ℓ in the Grönwall inequality, we have:

$$\begin{aligned}
-{\Delta^{\text{sand}'}}_i + \Delta_i - {\Delta^{\text{sand}}}_i &\leq m_i + q_i \sum_{\ell=1}^{i-1} m_\ell f_\ell \left(\prod_{k=\ell+1}^{i-1}(1+q_k f_k)\right) \\
&= m_i + \left(2+\frac{\kappa\gamma}{\mu}\right)\sum_{\ell=1}^{i-1} m_\ell \prod_{k=\ell+1}^{i-1}(1+q_k) \\
&= m_i + \left(2+\frac{\kappa\gamma}{\mu}\right)\sum_{\ell=1}^{i-1} m_\ell \left(1+2+\frac{\kappa\gamma}{\mu}\right)^{i-\ell-1} \\
&= m_i + \left(2+\frac{\kappa\gamma}{\mu}\right)\sum_{\ell=1}^{i-1} m_\ell \left(3+\frac{\kappa\gamma}{\mu}\right)^{i-\ell-1}
\end{aligned}$$

Once again negating, and adding Δ_i to both sides:

$$\Delta^{\text{sand}'}{}_i - \Delta^{\text{sand}}{}_i \geq -m_i + \Delta_i - \left(2 + \frac{\kappa\gamma}{\mu}\right) \sum_{\ell=1}^{i-1} m_\ell \left(3 + \frac{\kappa\gamma}{\mu}\right)^{i-\ell-1}$$

Combining m_i with Δ_i, we have:

$$\Delta^{\text{sand}'}{}_i - \Delta^{\text{sand}}{}_i \geq \frac{\kappa}{\mu}(\gamma+1)\Delta_i + \frac{\kappa}{\mu}\left(\frac{\mu}{\beta} - g(0)\right) - \left(2 + \frac{\kappa\gamma}{\mu}\right) \sum_{\ell=1}^{i-1} m_\ell \left(3 + \frac{\kappa\gamma}{\mu}\right)^{i-\ell-1}$$

We note that $\frac{\kappa}{\mu}\left(\frac{\mu}{\beta} - g(0)\right) \geq 0$ whenever $\mu \geq g(0)\beta$ and $-1 + \frac{\kappa}{\mu}(\gamma+1) \geq 0$ whenever $\gamma \geq \frac{\mu}{\kappa-1}$, and imposing these conditions, we have:

$$\Delta^{\text{sand}'}{}_i - \Delta_i - \Delta^{\text{sand}}{}_i \geq \left(2 + \frac{\kappa\gamma}{\mu}\right) \left(\sum_{j=1}^{i-1} \Delta^{\text{sand}'}{}_j - \Delta_j - \Delta^{\text{sand}}{}_j\right)$$

Applying Proposition 8, and in particular noting that $q_r = -\left(2 + \frac{\kappa\gamma}{\mu}\right)$, $f_\ell = 1$ for all ℓ, we have:

$$\begin{aligned}
\Delta^{\text{sand}'}{}_i - \Delta_i - \Delta^{\text{sand}}{}_i &\geq \prod_{\ell=1}^{i} \left(1 - \left(2 + \frac{\kappa\gamma}{\mu}\right)\right)^{-1} \\
&= \prod_{\ell=1}^{i} \frac{-1}{1 + \frac{\kappa\gamma}{\mu}} \\
&= \left(\frac{-1}{1 + \frac{\kappa\gamma}{\mu}}\right)^i
\end{aligned}$$

Adding Δ_i to both sides, we have:

$$\Delta^{\text{sand}'}{}_i - \Delta^{\text{sand}}{}_i \geq \Delta_i + \left(\frac{-1}{1 + \frac{\kappa\gamma}{\mu}}\right)^i$$

We take the bound for even i to get:

$$\Delta^{\text{sand}'}{}_i - \Delta^{\text{sand}}{}_i \geq \Delta_i + \left(\frac{\mu}{\mu + \kappa\gamma}\right)^i$$

□

F Proof of Proposition 1: Sandwich Pairwise Locality

We now show conditions for sandwich attacks to be *pairwise local*. That is, for any adjacent trades Δ_i, Δ_{i+1}:

$$\mathsf{PNL}(\Delta_i + \Delta_{i+1}) \leq \mathsf{PNL}(\Delta_i) + \mathsf{PNL}(\Delta_{i+1})$$

We use the bounds derived on PNL to show this. Recall that in the case of $\mathsf{PNL}(\Delta_i + \Delta_{i+1})$, the optimal sandwich for the composite trade $\Delta_i + \Delta_{i+1}$, ${\Delta^{\text{sand}}}_{i,i+1}$ satisfies:

$$G\left(\Delta_i + \Delta_{i+1} + {\Delta^{\text{sand}}}_{i,i+1} + \sum_{j=1}^{i-1} \xi_j\right) - G\left({\Delta^{\text{sand}}}_{i,i+1} + \Delta_{i+1} \sum_{j=1}^{i-1} \xi_j\right) = (1-\eta) G\left(\Delta_i + \Delta_{i+1}\right)$$

Using Lemma 7, we have:

$$\mathsf{PNL}(\Delta_i + \Delta_{i+1}) \leq \left(-\frac{\mu}{\kappa}(\nu+1)\right)(\Delta_i + \Delta_{i+1}) + \frac{\mu}{\kappa}\left(\frac{\kappa}{\beta} - g(0)\right) + \left(2 + \frac{\mu\nu}{\kappa}\right) \sum_{\ell=1}^{i-1} p_\ell \left(3 + \frac{\mu\nu}{\kappa}\right)^{i-\ell-1}$$

for $p_i = \left(-1 - \frac{\mu}{\kappa}(\nu+1)\right)(\Delta_i + \Delta_{i+1}) + \frac{\mu}{\kappa}\left(\frac{\kappa}{\beta} - g(0)\right)$.

Similarly, we have lower bounds for $\mathsf{PNL}(\Delta_i)$ and $\mathsf{PNL}(\Delta_{i+1})$ from Lemma 9, which gives:

$$\mathsf{PNL}(\Delta_i) \geq \Delta_i + \left(\frac{\mu}{\mu + \kappa\gamma}\right)^i$$

and

$$\mathsf{PNL}(\Delta_{i+1}) \geq \Delta_{i+1} + \left(\frac{\mu}{\mu + \kappa\gamma}\right)^{i+1}$$

Combining these bounds, we have:

$$\begin{aligned}
&\mathsf{PNL}(\Delta_i + \Delta_{i+1}) - \mathsf{PNL}(\Delta_i) - \mathsf{PNL}(\Delta_{i+1}) \\
&\leq \left(-\frac{\mu}{\kappa}(\nu+1)\right)(\Delta_i + \Delta_{i+1}) + \frac{\mu}{\kappa}\left(\frac{\kappa}{\beta} - g(0)\right) + \left(2 + \frac{\mu\nu}{\kappa}\right) \sum_{\ell=1}^{i-1} p_\ell \left(3 + \frac{\mu\nu}{\kappa}\right)^{i-\ell-1} \\
&- \Delta_i - \left(\frac{\mu}{\mu + \kappa\gamma}\right)^i - \Delta_{i+1} - \left(\frac{\mu}{\mu + \kappa\gamma}\right)^{i+1} \\
&= \left(\frac{\mu}{\kappa}(\nu+1) - 1\right)(\Delta_i + \Delta_{i+1}) + \frac{\mu}{\kappa}\left(\frac{\kappa}{\beta} - g(0)\right) \\
&+ \left(2 + \frac{\mu\nu}{\kappa}\right) \sum_{\ell=1}^{i-1} p_\ell \left(3 + \frac{\mu\nu}{\kappa}\right)^{i-\ell-1} - \left(\frac{\mu}{\mu + \kappa\gamma}\right)^i - \left(\frac{\mu}{\mu + \kappa\gamma}\right)^{i+1}
\end{aligned}$$

This bound gives us a sufficient condition for when $\mathsf{PNL}(\Delta_i+\Delta_{i+1})-\mathsf{PNL}(\Delta_i)-\mathsf{PNL}(\Delta_{i+1}) \leq 0$. In particular, we need:

$$\begin{aligned}&\left(\frac{\mu}{\kappa}(\nu+1)-1\right)(\Delta_i+\Delta_{i+1})+\frac{\mu}{\kappa}\left(\frac{\kappa}{\beta}-g(0)\right)\\&+\left(2+\frac{\mu\nu}{\kappa}\right)\sum_{\ell=1}^{i-1}p_\ell\left(3+\frac{\mu\nu}{\kappa}\right)^{i-\ell-1}-\left(\frac{\mu}{\mu+\kappa\gamma}\right)^i-\left(\frac{\mu}{\mu+\kappa\gamma}\right)^{i+1}\leq 0\end{aligned}\tag{44}$$

for all $i \in [n]$.

G Proof of Proposition 3

Recall by Lemma 7 that:

$$\Delta^{\text{sand}'}{}_i-\Delta^{\text{sand}}{}_i\leq\left(-\frac{\mu}{\kappa}(\nu+1)\right)\Delta_i+\frac{\mu}{\kappa}\left(\frac{\kappa}{\beta}-g(0)\right)+\left(2+\frac{\mu\nu}{\kappa}\right)\sum_{\ell=1}^{i-1}p_\ell\left(3+\frac{\mu\nu}{\kappa}\right)^{i-\ell-1}$$

Suppressing the constants, we write this as:

$$\mathsf{PNL}_i=\Delta^{\text{sand}'}{}_i-\Delta^{\text{sand}}{}_i\leq a\Delta_i+b+c\sum_{\ell=1}^{i-1}d^{i-\ell-1}\Delta_\ell$$

Now, by Lemma 9 we have:

$$\Delta^{\text{sand}'}{}_i-\Delta^{\text{sand}}{}_i\geq\Delta_i+\left(\frac{\mu}{\mu+\kappa\gamma}\right)^i$$

and once again suppressing constants:

$$\mathsf{PNL}_i=\Delta^{\text{sand}'}{}_i-\Delta^{\text{sand}}{}_i\geq\Delta_i+e^i$$

where $\Delta^{\text{sand}'}{}_i-\Delta^{\text{sand}}{}_i=\mathsf{PNL}_i$.

H Proof of Proposition 4

Using the bound from Proposition 3 we have for a permutation π:

$$\mathsf{PNL}_{\pi(i)}-\mathsf{PNL}_i\leq a\Delta_{\pi(i)}+b+c\left(\sum_{\ell=1}^{\pi(i)-1}d^{\pi(i)-\ell-1}\Delta_\ell\right)-\Delta_i-e^i$$

and correspondingly, we have a lower bound:

$$\mathsf{PNL}_{\pi(i)} - \mathsf{PNL}_i \geq \Delta_{\pi(i)} + e^{\pi(i)} - a\Delta_i - b - c\sum_{\ell=1}^{i-1} d^{i-\ell-1}\Delta_\ell$$

Applying max and taking expectations over $\pi \sim S_k$:

$$\mathop{\mathbf{E}}_{\pi\sim S_k}\left[\max_{i\in[k]} |\mathsf{PNL}_{\pi(i)} - \mathsf{PNL}_i|\right] \leq \mathop{\mathbf{E}}_{\pi\sim S_k}\left[\max_{i\in[k]} a\Delta_{\pi(i)} + b + c\left(\sum_{\ell=1}^{\pi(i)-1} d^{\pi(i)-\ell-1}\Delta_\ell\right) - \Delta_i - e^i\right]$$

We now adapt the methodology used by [CAE22] to get our final bound. First, define the partial sums $R_i(T_k,\pi) = a\Delta_{\pi(i)} + b + c\left(\sum_{\ell=1}^{\pi(i)-1} d^{\pi(i)-\ell-1}\Delta_\ell\right) - \Delta_i - e^i$ and consider the binary search tree $\mathsf{BST}(\mathbf{R}(T_k,\pi))$ whose root is $R_1(T_k,\pi)$. The elements $R_j(T_k,\pi)$ are added sequentially to this tree. Then, following [CAE22, §3], we have the following bounds:

$$\max_i |\mathsf{PNL}_{\pi(i)} - \mathsf{PNL}_i| \leq |R_1(T_k,\pi)| + \max_j \left|R_j(T_k,\pi)\right| \mathsf{height}(\mathsf{BST}(\mathbf{R}(T_k,\pi)))$$

Now, recalling from [Ree03] that for equiprobable permutations $\mathbf{E}_{\pi\sim S_k}[\mathsf{height}(\mathsf{BST}(\mathbf{R}(T_k,\pi)))] = \alpha\log k - \beta\log\log k$, we have:

$$\begin{aligned}
\mathop{\mathbf{E}}_{\pi\sim S_k}\left[\max_i |\mathsf{PNL}_{\pi(i)} - \mathsf{PNL}_i|\right] &\leq \mathop{\mathbf{E}}_{\pi\sim S_k}\left[|R_1(T_k,\pi)| + \max_j \left|R_j(T_k,\pi)\right| \mathsf{height}(\mathsf{BST}(\mathbf{R}(T_k,\pi)))\right] \\
&= \mathop{\mathbf{E}}_{\pi\sim S_k}[|R_1(T_k,\pi)|] + \left(\max_j |R_j(T_k,\pi)|\right) \mathop{\mathbf{E}}_{\pi\sim S_k}[\mathsf{height}(\mathsf{BST}(\mathbf{R}(T_k,\pi)))] \\
&\leq \mathop{\mathbf{E}}_{\pi\sim S_k}[|R_1(T_k,\pi)|] \\
&\quad + \max_{i,j}\left|a\Delta_i + b + c\sum_{\ell=1}^{i-1} d^{i-\ell-1}\Delta_\ell - \Delta_j - e^j\right| (\alpha\log k - \beta\log\log k)
\end{aligned}$$

where the last inequality uses the following identity

$$\max_j \left|a\Delta_{\pi(j)} + b + c\sum_{\ell=1}^{\pi(j)-1} d^{\pi(j)-\ell-1}\Delta_\ell - \Delta_j - e^j\right| \leq \max_{i,j}\left|a\Delta_i + b + c\sum_{\ell=1}^{i-1} d^{i-\ell-1}\Delta_\ell - \Delta_j - e^j\right|$$

Now, note that we can bound:

$$\mathop{\mathbf{E}}_{\pi\sim S_k}|R_1(T_k,\pi)| = \frac{1}{k}\sum_{j=1}^{k}\left|a\Delta_j + b + c\sum_{\ell=1}^{j-1} d^{j-\ell-1}\Delta_\ell - \Delta_1 - e^1\right| \leq \max_{i,j}\left|a\Delta_i + b + c\sum_{\ell=1}^{i-1} d^{i-\ell-1}\Delta_\ell - \Delta_j - e^j\right|$$

which gives us the bound:

$$\mathop{\mathbf{E}}_{\pi\sim S_k}\left[\max_i |\mathsf{PNL}_{\pi(i)} - \mathsf{PNL}_i|\right] \leq \max_{i,j}\left|a\Delta_i + b + c\sum_{\ell=1}^{i-1} d^{i-\ell-1}\Delta_\ell - \Delta_j - e^j\right| (\alpha\log k - \beta\log\log k)$$

which allows us to conclude that $\mathbf{E}_{\pi\sim S_k}[\max_i |\mathsf{PNL}_{\pi(i)} - \mathsf{PNL}_i|] = O(\log k)$.

I Proof of Proposition 5

Recall that we have the following lower bound on $\mathsf{PNL}_{\pi(i)} - \mathsf{PNL}_i$ from Section G:

$$\mathsf{PNL}_{\pi(i)} - \mathsf{PNL}_i \geq \Delta_{\pi(i)} + e^{\pi(i)} - a\Delta_i - b - c\sum_{\ell=1}^{i-1} d^{i-\ell-1}\Delta_\ell$$

Now, taking absolute values and averages, we have:

$$\begin{aligned}
\frac{1}{n}\sum_{i=1}^{n}|\mathsf{PNL}_{\pi(i)} - \mathsf{PNL}_i| &\geq \frac{1}{n}\sum_{i=1}^{n}|\Delta_{\pi(i)} + e^{\pi(i)} - a\Delta_i - b - c\sum_{\ell=1}^{i-1} d^{i-\ell-1}\Delta_\ell| \\
&\geq \frac{1}{n} n \min_i \left[\Delta_{\pi(i)} + e^{\pi(i)} - a\Delta_i - b - c\sum_{\ell=1}^{i-1} d^{i-\ell-1}\Delta_\ell\right] \\
&\geq \min_i \left[\Delta_{\pi(i)} + e^{\pi(i)} - a\Delta_i - b - c\sum_{\ell=1}^{i-1} d^{i-\ell-1}\Delta_\ell\right]
\end{aligned}$$

which allows us to conclude that $\frac{1}{n}\sum_{i=1}^{n}|\mathsf{PNL}_{\pi(i)} - \mathsf{PNL}_i| = \Omega(1)$.

J Proof of Theorem 1

We combine Propositions 4 and 5 to get the main result:

$$\begin{aligned}
\mathsf{CoF}(T_n) &= \frac{\mathbf{E}_{\pi\sim S_n}\left[\max_{i\in[n]}|\mathsf{PNL}_{\pi(i)} - \mathsf{PNL}_i|\right]}{\mathbf{E}_{\pi\sim S_n}\left[\frac{1}{n}\sum_{i=1}^{n}|\mathsf{PNL}_{\pi(i)} - \mathsf{PNL}_i|\right]} \\
&\leq \frac{\max_{i,j}\left|a\Delta_i + b + c\sum_{\ell=1}^{i-1} d^{i-\ell-1}\Delta_\ell - \Delta_j - e^j\right|(\alpha\log n - \beta\log\log n)}{\mathbf{E}_{\pi\sim S_n}\left[\frac{1}{n}\sum_{i=1}^{n}|\mathsf{PNL}_{\pi(i)} - \mathsf{PNL}_i|\right]} \\
&\leq \frac{\max_{i,j}\left|a\Delta_i + b + c\sum_{\ell=1}^{i-1} d^{i-\ell-1}\Delta_\ell - \Delta_j - e^j\right|(\alpha\log n - \beta\log\log n)}{\min_i\left[\Delta_{\pi(i)} + e^{\pi(i)} - a\Delta_i - b - c\sum_{\ell=1}^{i-1} d^{i-\ell-1}\Delta_\ell\right]} \\
&= O(\log n)
\end{aligned} \tag{45}$$

K Routing MEV

In this section, we provide proofs of Proposition 6 and Theorem 2. Together, these establish the locality of sandwich attacks on a CFMM network as well as the bound on price of anarchy.

K.1 Proof of Proposition 6

Defining Sandwich Profit on a Graph. In order to prove the bounds of Proposition 6, we first need to define what the sandwich profit will be. This profit function can then be used to implicitly define sandwich sizes and we can use (μ, κ)-smoothness to construct the bounds mentioned.

Definition 6. Given a path $p = (p_1, \ldots, p_k) \in \mathcal{P}$, we define the *cumulative output* of a path $p \in \mathcal{P}$ as

$$G_p(\Delta) = G_{p_k}(G_{p_{k-1}}(\ldots(G_{p_2}(G_{p_1}(\Delta)))))$$

where, *e.g.* edge p_1 is the edge adjacent to vertex $A \in V$

That is, we compose the output functions over a path to recover the amount of output token B that a user will receive over path p if they provide an input of Δ units of token A to the network. Under the presence of sandwich attackers on the network, we define the following analogue:

Definition 7. Let $\Delta^{\text{sand}}{}_e$ be a sandwich attack on edge $e \in E$ and $p = (p_1, \ldots, p_k) \in \mathcal{P}$. Then we define the *cumulative output under sandwiching* of path p to be

$$\hat{G}_p(\Delta) = G_{p_k}(G_{p_{k-1}}(\cdots(G_{p_2}(G_{p_1}(\Delta + \Delta^{\text{sand}}{}_{p_1}) + \Delta^{\text{sand}}{}_{p_2})\cdots + \Delta^{\text{sand}}{}_{p_{k-1}}) + \Delta^{\text{sand}}{}_{p_k})$$

to be the **cumulative output under sandwiching** of path p.

Similarly, we define the implied output quantities $\tilde{G}$ which represent the amount of price impact had we only had the sandwiches:

Definition 8. For a path $p = (p_1, \ldots, p_k)$, we define the *cumulative output of sandwiches only* as

$$\tilde{G}_p(\Delta) = G_{p_k}(G_{p_{k-1}}(\cdots(G_{p_2}(G_{p_1}(\Delta^{\text{sand}}{}_{p_1}) + \Delta^{\text{sand}}{}_{p_2}) + \cdots + \Delta^{\text{sand}}{}_{p_{k-1}}) + \Delta^{\text{sand}}{}_{p_k})$$

where $\Delta^{\text{sand}}{}_{p_k}(\Delta)$ is the implied sandwich size given an input trade of size Δ (note that this can be solved for recursively).

Definition 9. Define the *cumulative output without sandwiches* as

$$\bar{G}_{p_i}(\alpha_p \Delta) = G_{p_i^{k_i}}(G_{p_i^{k_i-1}}(\ldots(G_{p_i^2}(G_{p_i^1}(\alpha_p \Delta)))))$$

The slippage limit defines the allowed sandwich attacks. That is, the splitting α is accepted if:

$$\sum_p \hat{G}_{p_i}(\alpha_p \Delta, \Delta^{\text{sand}}{}_p) - \tilde{G}_{p_i}(\Delta^{\text{sand}}{}_p) = (1-\eta)\sum_p \bar{G}(\alpha_p \Delta) \tag{46}$$

Suppose we have a function ${\Delta^{\text{sand}}}_e$ that takes the net flow entering edge $e \in E$ and outputs the sandwich attack on that edge. Then, the cost on edge e is:

$$c_e(\Delta_e) = {\Delta^{\text{sand}'}}_e(\Delta_e, \eta_e) - {\Delta^{\text{sand}}}_e(\Delta_e, \eta_e)$$

This is the profit captured by the sandwicher on edge $e \in E$.

We can now write a defining equation for ${\Delta^{\text{sand}}}_e$ as a function of the flow entering that edge. We write this equation in words first and then incorporate symbols: At the terminal node, we know we need to receive $(1-\eta)\bar{G}(\Delta) = (1-\eta)G_T(\Delta)$ units of output token out. Now, look at the node immediately preceding it. Call this node e. We can write an equation:

$$G_e(\Delta_e + {\Delta^{\text{sand}}}_e) - G_e({\Delta^{\text{sand}}}_e) = (1-\eta)G_T(\Delta) - G_e(\hat{G}_{e-1}(\Delta))$$

where $\hat{G}_{e-1}(\Delta)$ is the profit up to node $e-1$.

Implied Slippage Limits over a Path. We seek an explicit representation of ${\Delta^{\text{sand}}}_e$ in terms of the flow entering that edge, Δ_e and the *global* slippage limit η. Suppose that we have a path $\mathcal{P} = (e_1, \ldots, e_T)$ where e_T is the terminal edge (*e.g.* returns desired output token when traversed). To do this, we write a Bellman-type equation that writes ${\Delta^{\text{sand}}}_{e_i}$ on every edge as a function of the terminal slippage η and the slippages that occurred before. For the final node, we have the following equivalence.

$$G_{e_{T-1}}(\Delta_{e_{T-1}} + {\Delta^{\text{sand}}}_{e_{T-1}}) - G_{e_{T-1}}({\Delta^{\text{sand}}}_{e_{T-1}}) = (1-\eta)G_T(\Delta_T) - G_{e_{T-2}}({\Delta^{\text{sand}}}_{e_{T-2}})$$

The left hand side of this equation represents the excess price impact that occurs at edge e_i when the path $\mathcal{P}$ is traversed. The right hand side is contribution to the terminal impact (a boundary term) from the e_{i-1}th edge. This effectively says the flow into e_{T-1} needs to be routed such that it exactly compensates for the excess price impact plus the output quantity. Another way of framing this condition is as a divergence-free condition for the flow (*e.g.* input flow and output flows have to be equal in terms of their net price impact). Similarly, we can recursively construct slippage limits for each e_i as

$$G_{e_{i-1}}(\Delta_{e_{i-1}} + {\Delta^{\text{sand}}}_{e_{i-1}}) - G_{e_{i-1}}({\Delta^{\text{sand}}}_{e_{i-1}}) = (1-\eta_i)G_{e_i}(\Delta_{e_i}) - G_{e_{i-2}}({\Delta^{\text{sand}}}_{e_{i-2}}) \tag{47}$$

From this we have a sequence of $T-1$ equations for solving for $T-1$ unknown variables $\eta_{e_1}, \ldots, \eta_{e_{T-1}}$. This can be solved via dynamic programming, as this is an analogue of the Kolmogorov backward equation, albeit for slippage limits. Therefore, there exists a unique way to solve for implied slippage limits along a route η.

This means that the net amount of output token the user receives from the CFMM network under sandwiching must be no more than $1-\eta$ times the amount the user would have received under no sandwiching. The optimal sandwich attacks ${\Delta^{\text{sand}}}_e$ solve the above equation (46). As there is just one equation for the network, but $|E|$ sandwiches to be solved for, we provide a heuristic that can be used to solve for each individual sandwich ${\Delta^{\text{sand}}}_e$ using a fixed point iteration, and use the solution that results to provide price of anarchy bounds for the network.

Proof of (18) **and** (19)**.** We use the equations (47) to construct the bounds on $\Delta^{\text{sand}}{}_e$ described in Proposition 6. We assume that we have uniform upper and lower bounds on all $G_e(\cdot)$. That is, $\kappa\Delta \leq G_e(\Delta) \leq \mu\Delta$ for all $e \in E$. Recall the equations:

$$G_{T-1}(\Delta_{T-1} + \Delta^{\text{sand}}{}_{T-1}) - G_{e_{T-1}}(\Delta^{\text{sand}}{}_{e_{T-1}}) = (1-\eta)G_{T-1}(G_{T-2}(\dots(\Delta)))$$

for the terminal sandwich $\Delta^{\text{sand}}{}_{T-1}$ and:

$$\begin{aligned} G_i(\Delta_i + \Delta^{\text{sand}}{}_i) - G_i(\Delta^{\text{sand}}{}_i) &= (1-\eta)G_{T-1}(G_{T-2}(\dots(\Delta))) \\ -G_i(G_{i-1}(G_{i-2}(\dots(G_1(\Delta) + \Delta^{\text{sand}}{}_1) + \dots\Delta^{\text{sand}}{}_{i-2}) + \Delta^{\text{sand}}{}_{i-1})) & \end{aligned}$$

for the intermediate sandwiches $\Delta^{\text{sand}}{}_i$ for $i = 1, \dots, T-2$. Let $\Delta_{T-1} = G_{T-2}(G_{T-3}(\dots(G_1(\Delta) + \Delta^{\text{sand}}{}_1) + \dots\Delta^{\text{sand}}{}_{T-3}) + \Delta^{\text{sand}}{}_{T-2})$. We now apply μ and κ bounds to the above equation to get:

$$\Delta_{T-1} \leq \mu^{T-2}(\Delta + \Delta^{\text{sand}}{}_1) + \mu^{T-3}\Delta^{\text{sand}}{}_2 + \dots + \mu\Delta^{\text{sand}}{}_{T-2}$$

Which gives us:

$$\begin{aligned} G_{T-1}(\Delta_{T-1} + \Delta^{\text{sand}}{}_{T-1}) - G_{T-1}(\Delta^{\text{sand}}{}_{T-1}) \leq \mu^{T-1}(\Delta + \Delta^{\text{sand}}{}_1) + \mu^{T-2}\Delta^{\text{sand}}{}_2 \\ + \dots + \mu^2\Delta^{\text{sand}}{}_{T-2} + \mu\Delta^{\text{sand}}{}_{T-1} - \kappa\Delta^{\text{sand}}{}_{T-1} \end{aligned}$$

and

$$\begin{aligned} &(1-\eta)G_T(G_{T-1}(G_{T-2}(\dots(\Delta)))) \\ &- G_{T-1}(G_{T-2}(G_{T-3}(\dots(G_1(\Delta) + \Delta^{\text{sand}}{}_1) + \dots\Delta^{\text{sand}}{}_{T-3}) + \Delta^{\text{sand}}{}_{T-2})) \\ &\leq (1-\eta)\mu^{T-1}\Delta - \kappa^{T-1}(\Delta + \Delta^{\text{sand}}{}_1) + \kappa^{T-2}\Delta^{\text{sand}}{}_2 + \dots + \kappa\Delta^{\text{sand}}{}_{T-2} \end{aligned}$$

Forcing the bound on the RHS to be greater than the bound on the LHS, we have:

$$\begin{aligned} \mu^{T-1}(\Delta + \Delta^{\text{sand}}{}_1) + \mu^{T-2}\Delta^{\text{sand}}{}_2 + \dots + \mu^2\Delta^{\text{sand}}{}_{T-2} + \mu\Delta^{\text{sand}}{}_{T-1} - \kappa\Delta^{\text{sand}}{}_{T-1} \\ \geq (1-\eta)\mu^{T-1}\Delta - \kappa^{T-1}(\Delta + \Delta^{\text{sand}}{}_1) + \kappa^{T-2}\Delta^{\text{sand}}{}_2 + \dots + \kappa\Delta^{\text{sand}}{}_{T-2} \end{aligned}$$

Moving all the $\Delta^{\text{sand}}{}_{T-j}$ terms for $j > 1$ to the RHS, we have:

$$\begin{aligned} (\mu-\kappa)\Delta^{\text{sand}}{}_{T-1} \leq (\mu^{T-1} - \kappa^{T-1})\Delta - (\eta+1)\mu^{T-1}\Delta \\ - (\mu^{T-1} + \kappa^{T-1})\Delta^{\text{sand}}{}_1 - \dots - (\mu^2 + \kappa^2)\Delta^{\text{sand}}{}_{T-2} \end{aligned}$$

and dividing by $\mu - \kappa$:

$$\begin{aligned} \Delta^{\text{sand}}{}_{T-1} \leq \frac{1}{\mu-\kappa}\Big((\mu^{T-1} - \kappa^{T-1})\Delta - (\eta+1)\mu^{T-1}\Delta \\ -(\mu^{T-1} + \kappa^{T-1})\Delta^{\text{sand}}{}_1 - \dots - (\mu^2 + \kappa^2)\Delta^{\text{sand}}{}_{T-2}\Big) \end{aligned} \tag{48}$$

Recall the defining recursion for a sandwich:

$$G_{e_{i-1}}(\Delta_{e_{i-1}} + \Delta^{\text{sand}}{}_{e_{i-1}}) - G_{e_{i-1}}(\Delta^{\text{sand}}{}_{e_{i-1}}) = (1-\eta_i)G_{e_i}(\Delta_{e_i}) - G_{e_{i-2}}(\Delta^{\text{sand}}{}_{e_{i-2}}) \tag{49}$$

Using the (μ, κ)-smoothness of $G_{e_{i-1}}$ we can upper bound the left hand side and lower bound the right hand side (which is the defining relation for η_i) as:

$$\mu(\Delta_{e_{i-1}} + \Delta^{\text{sand}}{}_{e_{i-1}}) - \kappa\Delta^{\text{sand}}{}_{e_{i-1}} \leq (1-\eta_i)\kappa\Delta_{e_i} - \mu\Delta^{\text{sand}}{}_{e_{i-2}}$$

Rearranging and collecting terms:

$$\Delta^{\text{sand}}{}_{e_{i-1}} \leq -\frac{\mu}{\mu-\kappa}\Delta^{\text{sand}}{}_{e_{i-2}} + \left(\frac{(1-\eta_i)\kappa - \mu}{\mu-\kappa}\right)\Delta_{e_i} \leq -\Delta_{e_i} - \frac{\mu}{\mu-\kappa}\Delta^{\text{sand}}{}_{e_{i-2}}$$

If combining this equation with (50) gives:

$$\begin{aligned}\Delta^{\text{sand}}{}_{T-1} \leq \frac{1}{\mu-\kappa}\Big((\mu^{T-1} - \kappa^{T-1})\Delta - (\eta+1)\mu^{T-1}\Delta \\ +(\mu^{T-1} + \kappa^{T-1})\Delta^{\text{sand}}{}_{e_1} - \cdots - (\mu^2+\kappa^2)\left(\Delta_{T-2} + \frac{\mu}{\mu-\kappa}\Delta^{\text{sand}}{}_{T-2}\right)\Big)\end{aligned} \tag{50}$$

Solving the recursions using Propositions 7 and 8 for this bound yields Eq. (18). We can compute a similar bound for Δ_{e_i} using the other bound for (49) and arrive at a similar bounded recursion, yielding (19).

K.2 Proof of Theorem 2

Recall that we can bound (using the α-stability and β-liquidity of g_e):

$$\begin{aligned}c_e(\alpha, \Delta) &= g_e(\Delta_e + \Delta^{\text{sand}}{}_e) - g_e(\Delta^{\text{sand}}{}_e) \\ &\leq \alpha(\Delta_e + \Delta^{\text{sand}}{}_e) - \beta\Delta^{\text{sand}}{}_e \\ &= \alpha\Delta_e + (\alpha-\beta)\Delta^{\text{sand}}{}_e\end{aligned}$$

Using the bound $\Delta^{\text{sand}}{}_e \leq f(\kappa,\mu,\eta)^{|\mathcal{P}|}\Delta_e$ for all e from Proposition 6, we have:

$$\begin{aligned}c_e(\alpha, \Delta) &\leq \alpha\Delta_e + (\alpha-\beta)f(\kappa,\mu,\eta)^{|\mathcal{P}|}\Delta_e \\ &= \left(\alpha + (\alpha-\beta)f(\kappa,\mu,\eta)^{|\mathcal{P}|}\right)\Delta_e\end{aligned}$$

We now deploy arguments from [Rou15] and in particular show that $\mathsf{PoA}(\Delta)$ is upper bounded by a constant using a (λ, μ)-smoothness argument. In particular, we seek to find constants λ and μ for $\lambda > 0$ and $0 < \mu < 1$ such that:

$$c_e(\Delta_e)\Delta_e^\star \leq \lambda c_e(\Delta_e)\Delta_e + \mu c_e(\Delta_e^\star)\Delta_e^\star$$

for any $\Delta_e^\star, \Delta_e$ and for all $e \in E$. For if this is the case, [Rou15] shows that $\mathsf{PoA}(\Delta) \leq \frac{\lambda}{1-\mu}$. Indeed, we see that we have for any equilibrium flows $\Delta_e^\star$ and optimal flows Δ_e:

$$\begin{aligned}
\sum_e c_e(\Delta_e^\star)\Delta_e^\star &\leq \sum_e c_e(\Delta_e)\Delta_e^\star \qquad (51)\\
&\leq \sum_e \left(\alpha + (\alpha-\beta) f(\kappa,\mu,\eta)^{|\mathcal{P}|}\right) \Delta_e \Delta_e^\star \\
&\leq \sum_e \left(\alpha + (\alpha-\beta) f(\kappa,\mu,\eta)^{|\mathcal{P}|}\right) \left(\Delta_e^2 + \Delta_e^{\star 2}\right) \\
&\leq \sum_e \left(\alpha + (\alpha-\beta) f(\kappa,\mu,\eta)^{|\mathcal{P}|}\right) \left(g(\kappa,\mu,\eta)^{|\mathcal{P}|} {\Delta^{\text{sand}}}_e \Delta_e + g(\kappa,\mu,\eta)^{|\mathcal{P}|} {\Delta^{\text{sand}^\star}}_e \Delta_e^\star\right)
\end{aligned}$$

where Equation (51) is by the definition of equilibrium, and the last inequality is by $xy \leq x^2 + y^2$ for all $x, y \in \mathbf{R}_+$. Using the condition placed on f and g makes the right-hand side constant.

We note that our conditions on f and g are relatively strong — they imply that the liquidity of the network has to grow rapidly with the diameter of the network graph for the PoA to be bounded. However, if one restricts themselves to a stricter set of CFMMs (e.g. geometric mean market makers or bounded support CFMMs [CAE21]), then these bound can be dramatically improved (as illustrated in the examples of §4.1 and §4.2).