The Grimm Brothers strategies: Abusing MEV searchers (and builders) for fun and profit

Bruno Mazorra
Universitat Pompeu Fabra
brunomazorra@gmail.com

Michael Reynolds
University College London
mireynolds@pm.me

September 11, 2022

Abstract

Maximal Extractable value (MEV) usually refers to the extra value that *privileged* players can extract through strategically order, censor and place transactions in a blockchain. These players, also known as searchers, explore different strategies and software designs to have competitive bots to extract MEV opportunities. To do so, searchers often search for new DeFi protocols, niche tokens and ways of generalizing their bots to extract more sophisticated MEV opportunities. However, this comes to a cost of falling to different honeypots or malicious protocols. In this paper, we will describe the *Grimm Brothers* strategies, different strategies to extract value from MEV bots or manipulate their behavior. These strategies can lead to extract funds from bots, but also to DDoS different networks or generate volume on poisonous tokens. However, not just searchers are victims to these strategies. In the paper, we will also give strategies that break fair competition on MEV-boost by exploiting inefficient builders. We openly explain this in order to make builders more aware of the difficulties of builders mechanisms designs.

Keywords: MEV, Flashbots, CFMM, DDoS, MEV-Boost, zero-sum games.

1 Introduction

Maximal extractable value is often studied from an optimal execution perception. However, in general, bots do not have perfect information about the MEV opportunities and its competitors. In some domains, bots responsible for extracting MEV opportunities have flaws or are designed to take some risks. After reverse engineering these bots, sophisticated players can exploit them and the structure of the MEV game for a specific objective. We will call this set of strategies *the Grimm strategies.* When an adversarial player is executing a Grimm strategy, it generates a set of negative externalities that he can exploit to extract or generate more MEV. A good example of this is Salmonella [1], a bot that tricks MEV bots (sandwich traders) to buy a honeypot token. In this paper, we will introduce different strategies that exploit the ordering mechanism and/or the vulnerabilities of other bots.

1.1 Our Contribution

In this paper contributes in the following point:

- Description of a strategy that uses other bots and the bad ordering design of Polygon network to DDoS it with nearly no cost.
- A strategy to drain sandwich bots using the public mempool (PGA).
- A strategy to increase profits using Flashbots v0.6 (Proof of work, before the merge).

We formally model and analyze these strategies and provide empirical results to validate the theoretical results obtained.

2 Preliminars

2.1 Local MEV

Definition 2.1. Let $\mathcal{D}$ be a domain with state $\mathtt{st}$, a player P with local mempool view $\mathcal{T}_P^M$ and a set of transactions $\mathcal{T}_P$ that the player P can construct. We denote by $\mathcal{C}_P = \mathcal{T}_P^M \cup \mathcal{T}_P$ to be the set of reachable transactions. We define the *local* MEV *of P with state $\mathtt{st}$* $\text{MEV}_P(\mathtt{st}, \mathcal{C}_P)$ as the solution to the following

[1] https://github.com/Defi-Cartel/salmonella, accessed on 20 August 2022

optimization problem

$$\begin{aligned} \max_{B} \quad & \Delta b(P; \mathtt{st} \circ B, \mathtt{st}) \\ \text{s.t.} \quad & B \subseteq \mathcal{C}_P, \\ & \mathtt{st} \to \mathtt{st} \circ B \text{ is a valid state transition in } \mathcal{D} \end{aligned}$$

Let $\text{argmev}_P(\mathtt{st})$ be the set of bundles that are a solution to the optimization problem. The constraints of reachable bundles is subject to a player's information, gas efficiency, budget, ability to propose blocks, etc.

Observe that if all players have access to the same MEV opportunities and have access to all bundles, then this definition is equivalent to the one provided in [1]. If the mempool view is the empty set, we will refer to the MEV as *on top of block* MEV, and we will denote it by TMEV_P.

2.2 Uniswap

Uniswap, the most relevant decentralized exchange and the first CFMM, was launched in November 2018 and, to date, more than 40,000 ERC-20 tokens are locked and tradable in the Uniswap protocol, adding a total value of 7 billion USD. In this section, we will provide an overview of the Uniswap V2 protocol.

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 numéraire. Then, a user's trade of size Δ (by providing Δ units of token A for token B, or analogously, the CFMM or liquidity provider is swapping A for B) is valid if:

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

where $1-\gamma$ represents the percentage fee parameter that controls how much the liquidity provider charges for facilitating the trade. Equivalently, the amount of output token that the user receives can then be computed as:

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

Note that when $\gamma = 1$, we have that $k = R_A R_B$ is always constant.

A *slippage limit* $\eta \in [0,1]$ represents how much of a price impact a user is willing to tolerate executing their trade, as measured by the minimum amount of the output token they are willing to receive.

Sandwich attack: Sandwich attack is a popular form of MEV strategy that exploits the slippage limits to take profits. A buy order $\mathtt{tx}$ of size Δ and slippage η will increase the price of an asset, while a sell order decreases the asset price. Therefore, sandwich searcher can utilize such price changes to take a profit from a victim transaction. Searchers can continuously monitor the network to find transactions $\mathtt{tx}$ which entails price differences. When a searcher observes $\mathtt{tx}$, it can buy the asset for a low price before the victim transaction is executed $\mathtt{tx}_b$, and sell the asset after the victim transaction increases the price $\mathtt{tx}_s$. However, in order to execute to make the execution profitable, the searcher must ensure that the transaction of the victim executes. That is, the victim must obtain at least $(1-\eta)G(\Delta)$. In this setting, the optimal size of the searcher buys is Δ_{sand} given by

$$G(\Delta + \Delta_{sand}) - G(\Delta_{sand}) = (1-\eta)G(\Delta) \tag{3}$$

and the optimal profits are given by

$$\text{PNL} = \Delta - G^{-1}((1-\eta)G(\Delta)).$$

TODO: Explain salmonella and posion tokens. And more details in sandwiching.

2.3 Salmonella

2.4 Proof of Work Flashbots auction v0.6

Flashbots Auction (the v0.6 before the Ethereum transition to proof of stake)[2] is a combinatorial auction that all allows players to bid for an ordered set of transactions, known as bundles. The allocation and payment rule used by the Flashbots auction has a similar mechanism to a first-price sealed auction. That is, the bundles are ordered by average gas price and afterwards prune the conflicting bundles. In case of symmetric gas efficiencies, the MEV opportunity is sealed to the higher bidder and pays what they bid. That is, if a set of bundles $B_1, ..., B_n$ compete and have the same gas costs, then the bundle appended in the chain will be the one with higher effective gas bid (see below).

In other words, Flashbots relayer try to build the block with the highest profits among all the block that can be constructed using the transactions in the public mempool and the Flashbots mempool of bundles. But the bundles have a number of allocation constraints which the Flashbots relayer must be accounted for [6]. In order to build the block with the

[2] After the merge, the Flashbots team will make a change to the Flashbots auction, not necessarily preserving the current structure of the auction. In the future, we will investigate this changes.

highest profit, Flashbots (to our knowledge[3]) uses a greedy approximation algorithm, ordering the bundles by average gas price or as the Flashbots team described in their docs, the score of the bundle. The score of a bundle B is defined as

$$sc(\mathtt{st} \circ B) \coloneqq \frac{\Delta_{coinbase}(\mathtt{st}) + \sum_{tx \in B \setminus \mathcal{TX}} g(tx)m(tx)}{\sum_{tx \in B} g(tx)}$$

where $\Delta_{coinbase}$ denotes the direct payment to the miner, $\mathcal{TX}$ is the set of mempool transactions, $g(tx)$ is the gas used by tx and $m(tx)$ is the gas price of tx. By definition, the score of the bundle depends on the current state $\mathtt{st}$. For an state $\mathtt{st}$ and a set of bundles $\mathcal{T}$, we denote by $\mathcal{T}^+(\mathtt{st})$ as the set of bundles that do not revert. For a bundle B, we denote by $\mathcal{R}(B;\mathtt{st})$ the bundles that conflict with B. Currently, MEV auctions are not open sourced or public available. However, we will make the following abstraction. Let $B_1, ..., B_n$ be all the bundles received by the Flashbots relayer, then the Flashbots combinatorial, can be simplified in the following way:

Algorithm 1 Abstraction of Flashbots auction algorithm v0.6

```
st ← Current state
block = []
while gasCost(st ∘ block) ≤ gasLimit
and T ≠ ∅ do
    B ← argmax_{B∈T⁺(st^(i))}{s(st ∘ B)}
    T ← T ∖ R(B, st)
    st ← st ∘ B
    block.append(B)
end while
```

Assuming that the number of operations per bundle is bounded by some constant M, we have that the computational complexity of this algorithm is $O(n^2)$. Since in general bundles do not change their scoring, another potential abstraction could be the following:

Algorithm 2 More efficient abstraction of Flashbots auction algorithm v0.6

```
T⁺(st) ← SortByScore (T⁺(st))
while gasCost(st∘B) ≤ gasLimit and T^(i) ≠ ∅ do
    B ← T⁺(st)[0]
    st ← st ∘ B
end while
```

Moreover, in some cases, ordering the bundles by effective gas price does not give an optimal revenue. Let $\mathrm{FBR}(B_1, ..., B_n)$ be the revenue using the Flashbots greedy approximation algorithm. Let $\mathrm{OTP}(B_1, ..., B_n)$ be the maximal revenue of the Flashbots combinatorial problem.

Proposition 2.2. The Flashbots combinatorial auction is not optimal. More specifically,

$$\inf\{\frac{\mathrm{FBR}(B_1, ..., B_k)}{\mathrm{OPT}(B_1, ..., B_k)} : \text{ for } B_1, ..., B_k \text{ bundles}\} \leq \frac{1}{\lfloor \frac{L}{g_{min}} \rfloor - 1},$$

where g_{min} is the minimal gas consumed by competing bundles and L is the gas limit of a block.

Proof: Let $B_1, ..., B_k$ all the bundles such that, B_1 compete with B_i for all $i \neq 1$ and B_i, B_j are pairwise non-competing bundles. Moreover, assume that B_1 has gas costs L/k and effective gas bid $m + \varepsilon$ for $\varepsilon > 0$ and all the other bundles have gas bid m. Then, the Flashbots algorithm outputs $B = \{B_1\}$, leaving to a sequencers' revenue of $m + \varepsilon$. On the other hand, the optimal valid block is $B = \{B_2, ..., B_k\}$ with $m(k-1)$ revenue. The result follows using bundles with gas cost g_{min}.

Searchers reputation: In order to maintain reliable Flashbots relay performance, Flashbots introduced searcher reputation to provide consistent access to the relay for searchers with a good performance track record during periods of heavy load. Reputation mechanism is the current solution used by the Flashbots builder to make the relay robust against sophisticated Layer 7 attacks. The current reputation system is designed to classify searchers into a high reputation and low reputation queue. The high reputation queue is designed to filter out searchers who use an excessive amount of computation resources on the relay. Otherwise, both queues are identical. According to Flashbots, the reputation of a searcher S is defined as

$$r(S) = \frac{\sum_{\mathtt{tx} \in \mathcal{T}^e(S)} \Delta_{coinbase}(\mathtt{tx}) + g(\mathtt{tx})m(\mathtt{tx})}{\sum_{\mathtt{tx} \in \mathcal{T}(S)} g(\mathtt{tx})} \quad (4)$$

where $\mathcal{T}(S)$ is the set of bundles send by S and $\mathcal{T}^e(S)$ is the subset of bundles sends by S and executed on chain. Nevertheless, probably Flashbots team has added a decay factor to the reputation to add more fairness to new users and current inefficient users in the high reputation queue. That is, for some time windows $[t_i, t_{i+1}]$, the reputation $r(S, i)$ is computed. Let t_k be the current time. Then, the total reputation is computed by

$$R(S) \coloneqq \sum_{i=0}^{k-1} f(t_k - t_{i+1}) r(S, i) \quad (5)$$

where $f : \mathbb{R}^+ \to [0, 1]$ is a monotone decreasing function.

[3] https://docs.flashbots.net/

2.5 MEV game theory

In this section, we will use the notion of MEV game explained in [3]. In the paper, the authors model an MEV game as a follows.

Definition 2.3. Given a set of players $\mathcal{P}$, a random positive variable $\mathcal{B}$, a gossip network graph G and an auction mechanism, $\mathbb{A} = (\mathbf{x}, \mathbf{pr})$, an MEV game is $\mathbb{G} = (\mathcal{P}, G, \mathcal{B}, \mathbb{A}, (c_i)_{i=1,..,n})$. We say the game is symmetric if all players share the same features.

The notion of random positive variable $\mathcal{B}$ models the block duration and the gossip network the latency between nodes. In this paper, we will assume that players do not see each other bids before the game ends. Moreover, we will assume that players will not be able to improve their features, so we will not take into account the function of the tuple of functions (c_i). The auction mechanism and the value that each player will be specified in each result.

However, the notion of Nash equilibrium is not strong in permissionless environments. For example, the equilibrium of firms competing in the Cournot price model [5] is weak against Sybil attacks. Another example is the equilibrium constructed in the theorem of Flashboys 2.0 [2] that proves that exponential raise bidding strategies with grim-trigger area a Nash equilibrium in the game with two players. Nevertheless, one can prove that agents have incentives to use Sybil attacks to maximize their payoff. For this reason, we introduce a stronger notion of a solution concept for MEV games that are resistant to Sybil attacks:

Definition 2.4. Let $\mathbb{G}$ be a symmetric[4] MEV game with n players, and let $\phi : \mathbb{N} \to \bigoplus_{i=1}^{\infty} \mathbb{S}_i$ be a strategy mapping. A Sybil resistant Nash equilibrium is ϕ such that for all $n \in \mathbb{N}$, $\phi(n) \in \bigoplus_{i=1}^{n} \mathbb{S}_i$, and $\phi(n)$ is a Nash equilibrium, where for each $i \in \{1, \dots, n\}$ and $j \in \{1, \dots, n+1\} - \{i\}$,

$$u_i(\phi(n)_i, \phi(n)_{-i}) \geq u_i(\phi(n+1)_i, \phi(n+1)_{-i}) + u_j(\phi(n+1)_j, \phi(n+1)_{-j}).$$

The notion of Sybil resistance is important in permissionless pseudo-anonymous environments such as blockchains. Players have the ability to generate additional addresses to take more profits from cooperative strategies. In future work, we will prove that Sybil resistant cooperative Nash equilibrium exist in the priority gas auction in the non-repeated games, and the existence of Sybil resistant equilibrium in games with private mempools.

[4] This definition can be extended to non-symmetric games, but we will leave it for future work.

3 Hamelin strategy

More examples can arise in different MEV games. In the following, we will explain a clear example of how to execute a Hamelin strategy in the Polygon network that could potentially break liveness in the Polygon network.

The strategy creates visible arbitrage opportunities in the mempool to incentivise MEV bots to spam arbitrage transactions. Manipulation of account nonces and the properties of bor client transaction propagation is used to prevent MEV bots from actually extracting the arbitrage opportunity, rather the trade reverses at no cost to the strategy user except for gas fees. This allows an adversary to create a large number of on-chain spam transactions for minimal cost. In other words, **the strategy bribes MEV bots to fill Polygon PoS blocks with spam transactions, without actually paying the bribe**. The following is an example of such a strategy that can easily be deployed on Polygon PoS using a public RPC with minimal capital requirements:

1. Choose two popular tokens, for example Matic and DAI, with an Uniswap V2 pool with enough liquidity.
2. Deploy a token $\mathbb{T}$ and deploy two Uniswap V2 pools $\texttt{pool}_1$ and $\texttt{pool}_2$ respectively with Matic and Dai.
3. Fix set of addresses $\mathcal{A}$.
4. For each address, construct a buy transaction $\texttt{tx}_1$ in $\texttt{pool}_1$ that leaves a triangular arbitrage opportunity of value $\delta >> 0$.
5. Increment the address nonce and construct a sell transaction $\texttt{tx}_2$ with the quantity of tokens $\mathbb{T}$ that the transaction $\texttt{tx}_1$ will generate, and place a slightly higher gas bid.
6. For each address, broadcast $\texttt{tx}_2$. Wait a few seconds and broadcast $\texttt{tx}_1$.
7. Go back to step 4).

We ran short tests of the example strategy on Polygon PoS mainnet to obtain data to establish a proof of concept; it was impossible to use a private testnet as we were unable to replicate the obfuscated behaviour of MEV bots on mainnet. We generated transactions exclusively **off-chain** with code that is **private** to us; we did **not** attempt the example strategy with such volume and order size as to cause a DDoS event or to cause user transactions to be censored; and we did **not** test the example strategy with such volume,

order size, and for a sustained period as to impact liveness of the chain.

Figure 1 displays the Polygon spam analysis tool results during testing of the example strategy using 5 addresses, an order size of 15 matic, for five iterations separated by 10 seconds; total estimated cost of execution was 0.5 Matic, where each 'burst' cost 0.1 Matic (gas price approximately 75 Gwei). This

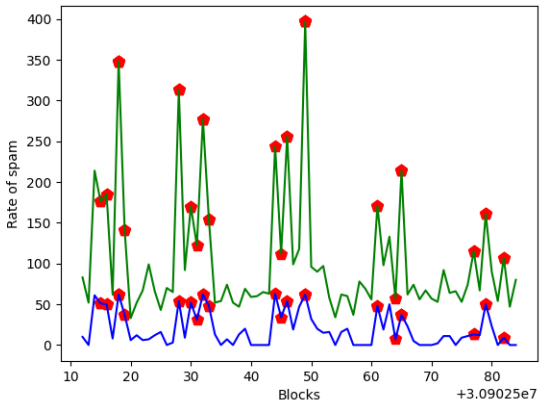

Figure 1: Green line = Number of transactions, Blue line = % Arbitrage transactions (lower bound), Red dots = Fake arbitrage opportunities

demonstrates that it was possible to increase the number of spam transactions in a block from 50 to around 150 (with spikes up to 400) for five blocks for a cost of 0.1 Matic with gas prices at 75 Gwei. We therefore anticipate from these results that by increasing the order size, number of addresses, and volume of trades, it would be possible for an attacker to fill every block with spam transactions initially at a price of 0.02 Matic per block given an initial base fee of 75 Gwei, with later block costs rising with the base fee according to the Polygon EIP-1559 implementation.

Put more simply, we anticipate that it is possible to fill an entire block with spam at 0.75% the cost of using the entire gas limit of a block.

Origin of the issue: It is well known that the transaction propagation mechanism of the bor client incentivizes spamming to extract MEV opportunities. A fair assumption is that the mechanism results in transactions landing in a random block according to a uniform distribution; with transactions ordered according to gas price within the block they end up in. Furthermore, it is reasonable to assume that transactions with the same gas price will be ordered according to arrival time at the elected validator provided they are valid. The above strategy works by ensuring the sell transaction $\texttt{tx}_2$ is propagated fully through the mempool, but is temporarily invalid since the nonce is too high. Then, after broadcasting the buy transaction $\texttt{tx}_1$, $\texttt{tx}_2$ only becomes valid once a validator receives $\texttt{tx}_1$, and since $\texttt{tx}_2$ has a higher gas price than $\texttt{tx}_2$, it is place directly after $\texttt{tx}_1$ in most cases. Now, MEV bots see $\texttt{tx}_1$ as a back-running MEV opportunity, and spam the network with transactions at the same gas price as $\texttt{tx}_1$ in an attempt to be positioned directly after it in the block to extract the MEV. $\texttt{tx}_2$ has a slightly higher gas price than $\texttt{tx}_1$, so $\texttt{tx}_2$ also appears as another back-running MEV opportunity, which causes MEV bots to spam the network to extract the MEV opportunity there as well. However, $\texttt{tx}_2$ doesn't represent a back-running MEV opportunity at all, since $\texttt{tx}_1$ will be placed directly before $\texttt{tx}_2$. On the other hand, bots do not have incentives to add more layers to their bots to prevent this kind of attack. Because, that would add less competitiveness against other bots, leaving extra profits if no one is executing this attack. For these reason, we do think that this is an error or bug of searchers' bots, but rather a fail on the ordering mechanism of Polygon chain.

Theoretical approximation: We will model the arbitrage game as a back running game that have structure of a random ordering MEV game where players do not see each other's bid before the next block. We will assume that the game is an MEV game $\mathbb{G}$, where all players observe the same value v. Players want to back run an MEV opportunity with gas price m. Also, players do not observe each other bids and the block production variable $\mathcal{B}$ is constant. Moreover, we will assume that all players same the same gas costs g for executing the MEV transaction and for the failed transactions γ. If there is a total of m transactions trying to extract the MEV opportunity and $k \leq m$ are sent by the player P_i, then:

$$\mathbb{E}[\Delta b_i] = \frac{k}{m+1}(\texttt{ev} - gm - (k-1)\gamma m_{\min}) - (1 - \frac{k}{m+1})k\gamma m$$

Theorem 3.1. Let $\mathcal{D}$ be a domain with private mempool with random ordering allocation rule. Assume that all players share the same gas cost g for extracting the opportunity and same gas costs for reverted transactions γ. Also, assume that the gas price is fixed for all players. Then,

the expected gas used to capture the fake MEV opportunities are given in the figure 2.

We recall that the block gas limit in polygon is

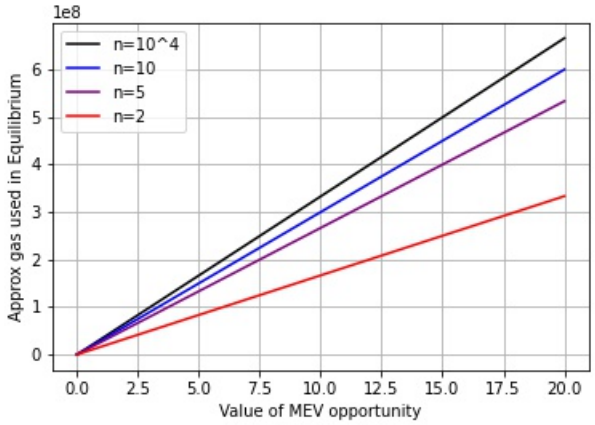

Figure 2: Gas misused in equilibrium in MEV backrunning game in Polygon.

30,000,000, therefore, with two and three competitive players would be enough to create a fake MEV opportunity of 20 and 10 respectively to fill the block with spam arbitrage transactions. Clearly, players that want to send a transaction to polygon will raise their bid enough. With a proper monitor of the mempool, the attacker could create fake MEV opportunities with a slighter gas bid. So, this transaction will be omitted in the following block. This method can be repeated for some time to stop break liveness for enough blocks. This strategy could be implemented to manipulate the price oracles or to artificially create cross chain MEV opportunities [4].

4 Hansel and Gretel strategy

As mentioned in the preliminary, Uniswap transactions with enough slippage can get sandwiched by searchers to extract net profits. The *Hansel and Gretel strategy*, similar to salmonella, tricks searchers to buy a token created by the executioner of the strategy, H for short. To this strategy to work, we have to assume that there are searchers in mainnet that try to capture the sandwich opportunities through PGA.

Let C be the total capital of H. The strategy works as follows:

- Deploy a token **T** and deploy an Uniswap V2 pools `pool` with $R_E \leq C$ and $R_{\mathbf{T}}$.
- Choose $\Delta + R_E \leq C$ and create a swap transaction `tx` with slippage η.
- Wait for a searcher S to send a buy transaction $\texttt{tx}_b$ with optimal sandwich Δ_{sand} in the mempool.
- Create a transaction that removes all the liquidity of the pool $\texttt{tx}_r$ and a bundle $B = \{\texttt{tx}_b, \texttt{tx}, \texttt{tx}_r\}$ and send it to Flashbots RPC with bid $\text{MEV}_S + \varepsilon$.

However, in general, searchers will not sandwich MEV opportunities with not enough net profits. Therefore, we must assume that searchers will try to sandwich the transaction trough PGA, if PNL $\geq$ L for some $L \in \mathbb{R}_{\geq 0}$. Therefore, MEV_H is the solution of the following optimization problem:

$$\begin{aligned} \max \quad & \Delta_{sand}(\Delta, R, \eta) \\ \text{s.t.} \quad & \Delta + R = C, \\ & \Delta - G^{-1}((1-\eta)G(\Delta)) \geq L, \\ & 0 \leq \Delta, 0 \leq R, 0 \leq \eta \leq \eta_{max}, \end{aligned}$$

Lemma 4.1. Let P be a pool with reserves R and R' of token A and B respectively. Let tx be a trade with amount Δ and slippage η. Then, the solution to 3 is given by:

$$\Delta_{sand} = \frac{-(\gamma\Delta + 2R) + \sqrt{(\gamma\Delta + 2R)^2 + 4(R^2 + R\gamma\Delta)\frac{\eta}{1-\eta}}}{2} \tag{6}$$

Proof. Similar to [6]. □

Observe that the function is increasing on η, therefore, to solve MEV_H we can assume that $\eta = \eta_{max}$.

Lemma 4.2. The function $\Delta_{sand}(\Delta, C - \Delta, \eta_{max})$ is monotone decreasing in $\Delta \in [0, C]$.

Proof. It is enough to prove it for $C = 1$ doing a change of variables. Then it is easy to check that $\frac{\partial \Delta_{sand}(\Delta, 1-\Delta, \eta_{max})}{\partial \Delta} < 0$. □

Corollary 4.3. The solution of MEV_H is given by $(\Delta_{min}, C - \Delta_{min}, \eta_{max})$, where Δ_{min} is the unique solution of

$$\Delta - G^{-1}((1-\eta)G(\Delta)) = L.$$

Proof. Since $\Delta_{sand}(\Delta, C - \Delta, \eta_{max})$ is decreasing in Δ, $\Delta - G^{-1}((1-\eta)G(\Delta)) \geq L$ and $\Delta - G^{-1}((1-\eta)G(\Delta))$ is increasing in Δ, we have that the optimal solution of MEV_H is given by Δ_{min}. □

Proposition 4.4. Let S be searcher and H be the one executing the Hansel strategy. Assume that $\mathtt{st}$ is a null MEV state, $\mathcal{T}_S^M = \{\mathtt{tx}\}$ and $\mathcal{T}_H^M = \{\mathtt{tx}, \mathtt{tx}_b\}$. Then,

$$\text{PNL}(\Delta, \eta) = \text{MEV}_S(\mathtt{st}) < \text{MEV}_H(\mathtt{st} \circ \mathtt{tx}_b) = \Delta_{sand} \tag{7}$$

where $\mathtt{st}$ is the current state and $\mathtt{tx}_b$ is the optimal buy transaction of S.

In general, Uniswap V2 sandwich attacks are executed by searchers using the Flashbots auction, paying more than 99%. To outbid conflicting bundles using the Flashbots auction, we have to bid $\text{MEV}_S(\mathtt{st})$. By proposition 4.4, this leads to positive profits, making this strategy dominant under PGA bots.

Theorem 4.5. If exist a PGA bot, the Hansel and Gretel strategy is profitable bidding L. Moreover,

$$\text{MEV}_H = C\left(\sqrt{\frac{1}{1-\eta_{max}}} - 1\right) + O(\gamma) \tag{8}$$

5 Red strategy: A strictly dominant strategy in PoW Flashbots auction v0.6

Flashbots relayer can simulate bundles in order to construct the most profitable bundles. However, there is a potential (and exploitable) difference between the simulation and the execution payoff. Since Flashbots can not predict the *special variables* of the block beforehand, it can not ensure the exactly payoff of each bundle, and so, the payoff of the block. Using this, one can probabilistically boost its bundle score $sc(B)$. More concretely, the score function can change through the execution of the block. Theoretically, this strategy would work in the new Flashbots auction. However, the changes in the Flashbots auction are deeper (and more robust) than the ones specified in the documents (see more details in empirical analysis section). So, we emphasize that this strategy would not have an impact in the new Flashbots auction. However, after MEV-Boost, this strategy could be used in other suboptimal builders.

5.1 Motivational example

Let B_1 and B_2 be two conflicting bundles that extract an MEV opportunity of value 4 ETH. The $\Delta_{coinbase}$ of the first bundle is 2 ETH. However, the $\Delta_{coinbase}$ of the second bundle is probabilistic. If $\mathtt{block.timestamp} \equiv 1 \pmod 2$ it pays 0.1 ETH. Otherwise, it pays $2 + \varepsilon$ ETH. If the Flashbots relayer makes a unique total block simulation, then he will choose B_1 with probability 1/2, and B_2 with probability 1/2. The expected payoff of the first player is 1 since he has 1/2 probability of winning and the total profit is 2. On the other hand, the second player has an expected profit of ≈ 2.

More generally, *fake bidding* is the strategy used by adversarial searchers to fake bids in order to outbid competitors in the FB block simulation. This, would in expectancy, increase their revenue. This strategy can be used after boosting their address reputation to increase its impact. Moreover, fake bidding strategy can be generalized with multiple accounts, to ensure winning the bundle and minimizing the payment with high probability.

5.2 Theoretical analysis

Assuming that, the Flashbots relayer does a unique simulation per block/bundle (this can be generalized with more simulations). The adversary have n accounts with high reputation queue. Let $p \in [0, 1]$. Using the special variables, we can construct a random variable such that the bid b is b_{max} with probability p and b_{min} with probability $(1-p)$. Assume that b_{max} is the exact value of the MEV opportunity. Assume that a set of searchers $\mathcal{P} = \{1, ..., n\}$ are trying to capture the same common knowledge MEV opportunity of gross value v through Flashbots. Let B be the bundle of the player using this strategy and $B_1, ..., B_n$ the bundles of $1, ..., n$ respectively.

Let b_{max} be bid minimum bid such that $s(B) > \max_i s(B_i)$. In this setting, we have that the expected payoff of playing this strategy is

$$\begin{aligned}\mathbb{E}(u \mid p, n, b_{min}, b_{max}) = (1-(1-p)^n)(p(v - b_{max}) \\ +(1-p)(v - b_{min}))\end{aligned}$$

From this equation, it follows immediately the following result.

Lemma 5.1. Assume that all players and in high reputation queue, then:

$$\lim_{p \to 0} \lim_{n \to +\infty} \mathbb{E}(u \mid p, n, b_{min}, b_{max}) = (v - b_{min}) \tag{9}$$

In reality, we see that players that try to capture common knowledge expend similar gas cost for extracting the opportunity. This, makes the estimation of b_{max} feasible. Moreover, in reality, we see that the common knowledge MEV opportunities are nearly completely paid to the miner. For example, we see that Arbitrages using Uniswap V2 protocol or sandwich attacks pay around 99% of the gross profit to the

miner. A theoretical explanation of this is given the following proposition. We prove that, if all p ers bid through gas price without using fake bidc strategy, and have identical gas efficiencies, then all Nash equilibrium lead to maximum bid.

Proposition 5.2. Define the strategy of a searc gas price bidding m be $s(m)$. Assume that gas c induced by winning the MEV opportunity are g ... = g_k = g. Then, let $\mathbb{G}$ be the subgame of Flashbots auction game with MEV opportunity v value v being public information where players use gas price. Then, the set of Nash equilibrium

$$\{(s_1, ..., s_n) \in S_1 \times ... \times S_n : \exists i, j \text{ s.t. } s_i = s_j = s(v/$$

where S_i is the set of all strategies of player i. If players follow the best response strategy, they will converge to the equilibrium

$$(s_1(v/g), ..., s_n(v/g)) .$$

The set of Nash equilibrium is invariant under risk-lover, risk-neutral and risk-averse agents. In particular, in equilibrium, the all the value v is paid to the validator.

However, if a player can use the red strategy, we obtain a completely different results.

Theorem 5.3. Assume that PoW Flashbots builder follows the algorithm 1. Then, the red strategy strictly dominates honest bidding. Moreover, the Flashbots builder is not Sybil resistant and does not maximize validators' profits.

Proof. Assume wlog that players $i = 2, ..., n$ follow an honest strategy. Therefore, their payment $\mathbf{pr}_i \leq v$, since bidding more than v is a strictly dominated strategy, since lead to negative revenue. Let $v_{max} = \max_{j\neq 1}\{v_j\}$. Let's define a fake bidding strategy with $b_{min} = 0$ and $b_{max} = v_{max} + \varepsilon$ with probability p. This strategy leads to $p^2(v - v_{max} - \varepsilon) + p(1-p)v$ profits. For $v_{max} > v/2$ the strategy strictly dominates any other honest bidding strategy. Otherwise, set the fake bidding strategy with $b_{max} = b_{min}$. □

5.3 Game theoretical analysis of the strategy

In previous sections, we have seen that the strategy dominates honest bidding. Moreover, with enough accounts in the high priority queue, the profits of the searchers converge to all the MEV. In this section, we will analyze the game where all players are aware of the Fake bidding strategy.

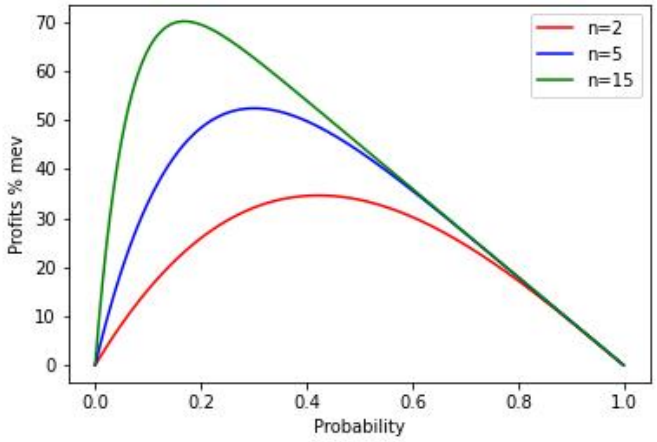

Figure 3: Profits of the Strategy

PoW Flashbots auction game: We will model this game as a MEV game []. We have a set of players $\mathcal{P}$, we assume that players do not see each other's bids and the auction mechanism is the Flashbots mechanism. In the Flashbots subgame without fake bids, players must choose a bid b. However, in the Flashbots game players can choose a positive random variable X, called bidding variables, that determinate its priority in the block construction and its payment to the miner. That is, first there is an instance of the bidding variables to chooses which bundle is executed. Then, there is a second instance that chooses the payment of the winner of the simulation with her respective bidding variable. More precisely, if a set of players $\mathcal{P}$ with same gas efficiency bid for an MEV of net value v with bidding variables $X_1, ...X_n$. Then, the utility of the first players is

$$u_i(X_i, X_{-i}) = \Pr[X_i > \max_{j\neq i}\{X_j\}]\,(v - \mathbb{E}(X_i))\,. \quad (10)$$

Theorem 5.4. Let $\mathbb{G}$ be the Flashbots game and $\mathcal{P}$ a set of n players extracting an MEV opportunity of net value v. Assume that all players share the same gas costs for extracting the MEV opportunity. Then, there exist a Nash equilibrium S such that $u_i(S) > 0$ for all i. In other words, Flashbots auction does not maximize revenue.

Proof. Assume that $S = (X_1, ..., X_n)$ is a Nash equilibrium. Moreover, let's assume that $u_1(S) = 0$. Then, we have that $u_1(b, X_{-i}) = \Pr[b > \max_{j\neq i}\{X_j\}](v - b)$ for all $b \in \mathbb{R}_{\geq 0}$. Since, S is a Nash equilibrium, we have that, for all $b \geq v$, we have that $u_1(b, X_{-i}) = 0$. Therefore, we have that $\Pr[v > \max_{j\neq 1}\{X_j\}] = 0$. On the other hand,

since S is a Nash equilibrium, we have that $\Pr[v < \max_{j\neq 1}\{X_j\}] = 0$, otherwise, there exist a player i with negative utility. So, we have that $\Pr[v < \max_{j\neq 1}\{X_j\}] = 1$. Now, consider the strategy given by the binomial variable $X = (v+\varepsilon)\mathrm{B}(2,p)$ with probability p. Clearly, $u_1(X, X_{-i}) > 0$ leading to a contradiction since S is a Nash equilibrium. □

5.4 Empirical anaylsis

To prove the assumption that bundles are simulated once, we made the following. We deployed a smart contract in Goerli network with two functions, `RandomBid` and `NotRandomBid`. To execute the functions, both require that a variable `bool` is True, otherwise the transaction reverts. Once executed, the functions change `bool` to False. Therefore, both transactions conflict. The function `NotRandomBid` pay the 0.0015 of Goerli ETH. While the `RandomBid` function bids 0.003 or 0.[5] We repeated this experiment 128, landing a total of 55 `RandomBid` and 73 `NonRandomBid`[6]. The 55 `RandomBid`, 32 were bidding the minimum. Therefore, we obtain that the 25% the random bid strategy outbid the normal one but with less ex-post effective gas price.

6 Conclusions

In this paper, we showed that there are different ways to exploit vulnerabilities of different ordering mechanisms, bots design and builder mechanisms. In general, we think that attacks to other players (chains, builders and searchers) is something that will happen to decrease the competitors and efficiency of competitors. Since a lot of the problems that searchers and builders try to solve are NP-problems, we think that, in general, similar strategies will be used to make competitors more inefficient.

References

[1] Kushal Babel et al. "Clockwork finance: Automated analysis of economic security in smart contracts". In: *arXiv preprint arXiv:2109.04347* (2021).

[2] Philip Daian et al. "Flash boys 2.0: Frontrunning in decentralized exchanges, miner extractable value, and consensus instability". In: *2020 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2020, pp. 910–927.

[3] Bruno Mazorra, Michael Reynolds, and Vanesa Daza. "Price of MEV: Towards a Game Theoretical Approach to MEV". In: *arXiv preprint arXiv:2208.13464* (2022).

[4] Alexandre Obadia et al. "Unity is Strength: A Formalization of Cross-Domain Maximal Extractable Value". In: *arXiv preprint arXiv:2112.01472* (2021).

[5] Roy J Ruffin. "Cournot oligopoly and competitive behaviour". In: *The Review of Economic Studies* 38.4 (1971), pp. 493–502.

[6] Alejo Salles. "On the Formalization of MEV". In: *https://writings.flashbots.net/research/formalization-mev/* (2021).

[5] To be sure that both transaction were not rejected by the Flashbots relayer, we made more operations to at least expend 42000 gweis.

[6] More details in GrimmBrothers GitHub.