Generalised MEV detection

Timur Badretdinov

Abstract

Detection and classification of historical MEV is an open research problem. The current dominating approach is to use specific heuristics for each type of MEV and data classifiers for each protocol. While this method definitely works for the most common types of MEV (e.g. DEX arbitrage), it certainly lacks utility when it comes to the long-tail MEV. Here, we offer a detection model that relies on general-purpose heuristics that are able to detect MEV beyond arbitrages and liquidations. We demonstrate that this model can detect DEX arbitrages across various trading protocols, as well as more esoteric types of MEV.

Introduction

Detection and classification of historical MEV is an open research problem. The current approach usually revolves around defining a set of classifiers to parse the protocol data and detect certain types of MEV (see mev-inspect-py, mev-inspect-js). Those solutions are effective, but limited in scope. DeFi is always evolving, new protocols appear on a monthly basis, and many types of MEV are hidden from the public.

Instead, we can experiment with a generalised approach. Here, instead of defining protocol- or type-specific classifiers, we aim to define a set of general-purpose heuristics that work for any kind of MEV.

To start, it's helpful to define what MEV is. In general, we can describe it as *a transaction or a series of transactions that can earn value risk-free or near risk-free in a short period of time*. This includes arbitrage, liquidations, sandwiches, and excludes trading. Here, we also consider protocol exploits as MEV.

We will limit the scope of this research to single-transaction single-domain MEV. Other, more broad MEV detection is a potential idea for further research.

Model

To start, we will split all MEV into two categories: pure arbitrage and value arbitrage.

Pure arbitrage is a set of actions that result in a net positive balance of each asset for a given account. An example of pure arbitrage is DEX arbitrage.

Value arbitrage is a set of actions that result in a net positive USD value for a given account. Consequently, value arbitrage is a superset of pure arbitrage. An example of value arbitrage is (profitable) collateral liquidation.

For the purpose of this research, we will focus on pure arbitrage. Value arbitrage is a potential idea for further research.

To detect pure arbitrage, we split the process into three steps. First, we extract asset transfers from the blockchain. Second, we calculate balance deltas for each account to get a shortlist of potential MEV transactions. Finally, we apply scoring rules to each item in the shortlist. Any transaction-account pair from the second step that gets the score above the threshold is considered MEV.

To fetch the asset transfers, we need to get both transaction traces and receipts from the blockchain. We use traces to parse ether transfers and receipt logs for ERC20, ERC721, and ERC1155 transfers.

Once we got the transfers, we can calculate the balance deltas. Since we're working at the transaction level, we process each transaction separately. For each one of them, we iterate over transfers of that transaction. For each transfer, we decrease the balance of *transfer.from* account and increase the balance of *transfer.to* account. After that, we iterate over accounts: if any of the accounts have net positive balances for each asset, add the transaction-account pair to the shortlist.

Finally, we apply the scoring rules. Each rule checks the transaction and/or account, looking for a specific pattern. If the pattern is to be found, the score is increased. Currently, all rules are *multiplicative*.

For example, one heuristic looks for outgoing transfers. The idea here is to weed out "rent seeker" wallets and contracts, like NFT royalty collectors, tokens with a transfer fee, DEX aggregators charging the swap fees, and so forth. All these entities are net balance positive after a transaction, but can't be considered as MEV. Incidentally, they usually only receive assets and don't send anything back, unlike MEV searchers and their pet contracts.

A full list of scoring rules can be found in the source code (rules and weights valid at the time of writing).

Results

To measure quantitative results, I run the algorithm until 200 MEV transactions were found. I then manually inspected and labeled them accordingly. Finally, I compared it with the results found with specific MEV detectors. From there, I was able to calculate the approximate false negative (i.e. transactions that include arbitrages but weren't detected) and false positive (i.e. transactions falsely considered as MEV) rates. I was also able to compare the general detector with the specific ones in the situations where the former successfully detected an arbitrage while the latter missed it.

Out of those 200 transactions, 14 were falsely marked as MEV. Four transactions can be considered long-tail MEV. Other 182 transactions are DEX arbitrages; 46 of them were detected exclusively by the generalised method.

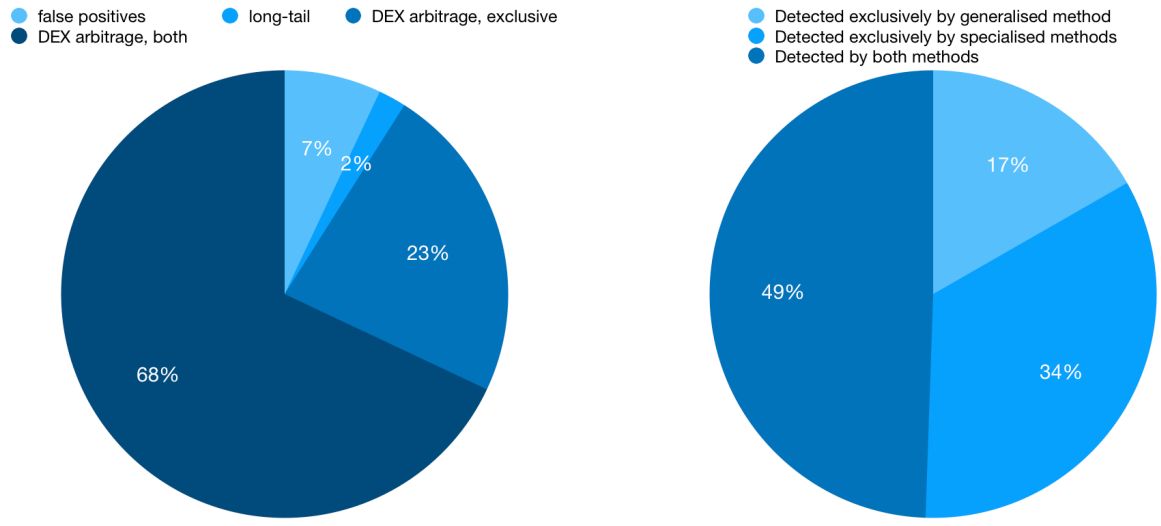

Figure 1. On the left — breakdown of MEV found by generalised method (N = 200). On the right — breakdown of all DEX arbitrage by either of 3 tools (generalised detector, mev-inspect-js, mev-inspect-py) (N = 293).

For comparison, mev-inspect-py detected 111 arbitrages, and mev-inspect-js detected 235 arbitrages in the same timeframe. Note that both work at the swap level, and often find multiple arbitrage events in a single transaction. If we count unique transaction hashes, mev-inspect-py found 101 transactions, while mev-inspect-js found 215 transactions.

But what's maybe more interesting is the MEV that was detected outside of usual DEX arbitrage, the so-called *long-tail MEV*.

First, there are transactions that print money. One case is an MEV searcher taking tokens out of imbalanced Uniswap pools without sending anything in return (example).

The second example of such MEV is NFT arbitrage. While technically it can be detected via specialised inspectors, they are other neglected (example).

Finally, there are complex arbitrages found that are unlikely to be classified correctly via specialised detectors. Such detectors usually look for matching input and output amounts (e.g. 1 WETH → 2000 USDC → 40 XYZ → 1.01 WETH), and complex arbitrages with split amounts (e.g. 1 WETH → 2000 USDC, 1000 USDC → 0.51 WETH, 1000 USDC → 20 XYZ → 0.51 WETH) may be omitted (example).

Future work

First of all, it would be great to reduce both false positive and false negative rates. I believe that by adding a few scoring rules, it's possible to bring down both to about 1%. Additionally, it makes sense to invest more time tweaking scoring weights and the default threshold value.

Second, it would be curious to implement the support for single-block or maybe even multiple-block MEV detection.

Finally, we can broaden our research by including value arbitrage detection.