The Potential of Self-Regulation for Front-Running Prevention on DEXes

Lioba Heimbach, Eric Schertenleib, Roger Wattenhofer

WEIS 2023

ETH Zurich
Decentralized exchanges (DEXes)
Decentralized exchanges (DEXes)
Constant product market makers (CPMMs)

liquidity pool for every token pair
Constant product market makers (CPMMs)

liquidity pool for every token pair

liquidity providers deposit reserves in pools
Constant product market makers (CPMMs)
Constant product market makers (CPMMs)

price curve: $x \cdot y = k$

trading along price curve
Constant product market makers (CPMMs)

price curve: $x \cdot y = k$

trading along price curve

$T$: trade $X \rightarrow Y$
Constant product market makers (CPMMs)

price curve: \( x \cdot y = k \)

trading along price curve

\( T \): trade \( X \rightarrow Y \)

liquidity providers earn fees proportional to trade input
Constant product market makers (CPMMs)

trading along price curve

$T$: trade $X \rightarrow Y$

liquidity providers earn fees proportional to trade input

the higher the liquidity the better the price

price curve: $x \cdot y = k$
Unexpected slippage

price curve: $x \cdot y = k$

unexpected slippage: unexpected price increase/decrease
Unexpected slippage

Unexpected slippage: unexpected price increase/decrease

Slippage tolerance specifies maximum price movement

Price curve: $x \cdot y = k$
Unexpected slippage

unexpected slippage: unexpected price increase/decrease

slippage tolerance specifies maximum price movement

trade fails if slippage tolerance exceeded

price curve: $x \cdot y = k$
Sandwich attack mechanism
Sandwich attack mechanism
Sandwich attack mechanism
Sandwich attack mechanism
Sandwich attack mechanism
Sandwich attack mechanism
Sandwich attack mechanism
Sandwich attack mechanism

$Y_{reserves}$

$X_{reserves}$

$T_V$

$T_{A_1}$

$T_{A_2}$

$a_y$

$a_x^{in}$

$a_x^{out}$
Front-running on DEXes

<table>
<thead>
<tr>
<th>Summary</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Tx Count</td>
<td>156866</td>
</tr>
<tr>
<td>Profit</td>
<td>$2,362,169.39</td>
</tr>
<tr>
<td>Cost</td>
<td>$11,782,863.77</td>
</tr>
<tr>
<td>Attackers</td>
<td>198</td>
</tr>
<tr>
<td>Victims</td>
<td>73418</td>
</tr>
</tbody>
</table>

https://eigenphi.io/mev/ethereum/sandwich
Front-running on DEXes

<table>
<thead>
<tr>
<th>Summary</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Tx Count</td>
<td>156,866</td>
</tr>
<tr>
<td>Profit</td>
<td>$2,362,169.39</td>
</tr>
<tr>
<td>Cost</td>
<td>$11,782,863.77</td>
</tr>
<tr>
<td>Attackers</td>
<td>198</td>
</tr>
<tr>
<td>Victims</td>
<td>73,418</td>
</tr>
</tbody>
</table>
Are liquidity providers incentivized to move to DEXes that implement front-running protection?
Pools

liquidity pool with front-running

\( W \)
Pools

liquidity pool with front-running

liquidity pool without front-running
Pools

liquidity pool with front-running

liquidity pool without front-running

$W$

$N$

external market price constant
Pools

liquidity pool with front-running

$W$

liquidity pool without front-running

$N$

same fee $f$ in both pools
Pools

$W$ is a liquidity pool with front-running, and $N$ is a liquidity pool without front-running. Both pools have the same slippage tolerance $s$. 
sandwich attacker
Players

sandwich attacker

arbitrageur
Players

sandwich attacker  arbitrageur  trader
Players

sandwich attacker

arbitrageur

trader

liquidity provider
Sandwich attacker

maximize profit from attack
Sandwich attacker

$\mathcal{P}$
Sandwich attacker

\[
\max a_x^{\text{out}} - a_x^{\text{in}}
\]
Arbitrageur

restore price in pools
Arbitrageur

\[ P = P_N = \frac{y_N}{x_N} \]
Arbitrageur
Arbitrageur

\[ P = P_W = \frac{y_W}{x_W} \]
maximize personal benefit
Trader

maximize personal benefit

associates personal benefit $\alpha$ with token $Y$
Trader

![Diagram of a trader with a graph showing reserves and pool size.]
Trader

\[ \max (1 + \alpha) \delta_{y_N} - \frac{y}{x} \delta_{x_N} \]
Trader

Pool$_W$
Trader

\[
\max (1 + \alpha) \delta_{yw} - \frac{y}{x} \delta_{xw}
\]
Liquidity provider

maximize fee revenue
Liquidity provider

maximize fee revenue

receives fees proportional to trade input amount
Liquidity provider
Liquidity provider

\( F_W \)

\( \text{Pool}_W \)
Liquidity provider

\[ \text{max } F_N + F_W \]
Are liquidity providers incentivized to move to DEXes that implement front-running protection?
Homogenous Traders

same relative benefit $\alpha$ for all traders
Nash Equilibrium in Homogenous Setting

![Graph showing Nash Equilibrium in Homogenous Setting](image)
Nash Equilibrium in Homogenous Setting
Nash Equilibrium in Homogenous Setting
for most parameter configurations \( Pool_N \) is the Nash equilibrium
for most parameter configurations $Pool_N$ is the Nash equilibrium

liquidity providers are currently in markets without front-running protection
Summary

for most parameter configurations $Pool_N$ is the Nash equilibrium

liquidity providers are currently in markets without front-running protection

benefit from adjusting a liquidity distribution is often only small
Summary

for most parameter configurations $Pool_N$ is the Nash equilibrium

liquidity providers are currently in markets without front-running protection

benefit from adjusting a liquidity distribution is often only small

liquidity provider might require additional benefits to move liquidity
The Potential of Self-Regulation for Front-Running Prevention on DEXes

Questions?

@liobaheimbach
hlioba@ethz.ch