DeFiesta DEX

8 August 2023

Abstract

Impermanent Loss (IL) is a significant challenge in the Decentralized Finance (DeFi) space that arises when users provide tokens to a liquidity pool, and the price of those tokens changes, either upward or downward. Addressing this issue has been a major obstacle for DeFi, and all previous attempts to solve it have been unsuccessful. This paper introduces a new technology based on formulas that manage liquidity differently by using Fictive Reserve (FR). Although approaches involving FR have been attempted in the past, they did not solve the problem of IL, and in some cases, they made it worse. This scientific article details the functioning of a new Decentralised Exchange (DEX) protocol implemented on Defiesta that, for the first time, solves the problem of IL. It discusses the technical and security challenges of this protocol and explains how it opens a new door for DeFi, enabling liquidity providers to achieve Impermanent Gain (IG).

1 Introduction

Defiesta DEX is an Automated Market Maker (AMM) that addresses the issue of IL and in some cases transforms it into IG. It is an open-source Smart Contract (SC), which is a decentralized software that runs on compatible Ethereum Virtual Machine (EVM) blockchains (such as Ethereum, Binance Smart Chain (BSC), Avalanche, Polygon, etc.). These blockchains are data exchange protocols that, similar to the Bitcoin blockchain, allow for the storage and transmission of information in a public, immutable, and decentralized manner. By using Defiesta DEX, users can exchange decentralized ERC20 tokens, which are digital assets.

1.1 Decentralized Exchanges and Impermanent Loss

Although there are numerous DEXs operating on various networks, it seems that none of them have satisfactorily addressed the issue of impermanent loss so far. Presently, one of the most common methods for creating a DEX involves the use of liquidity pools with the k constant rule (also known as the constant product rule). A pair structure consisting of two different tokens is established, enabling users to swap one token for the other. To achieve this, liquidity providers must deposit liquidity, thereby allowing the protocol to execute swaps. Let k denote the product of the quantities of tokens deposited by liquidity providers, and let x and y represent the quantities of the two tokens in the deposited pair of tokens.

$$k = xy$$

The rule governing swaps dictates that the constant k remains invariant. Therefore, if a user intends to acquire a quantity of Δx tokens (which involves their removal from the protocol) and sell a quantity of Δy tokens (thus adding them to the protocol), the reserves will evolve according to the following:

$$x_{new} = x - \Delta x$$

$$y_{new} = y - \Delta y$$

k must remain invariant:

$$k = x \cdot y = x_{new} \cdot y_{new}$$

$$x.y = (x - \Delta x) \cdot (y + \Delta y)$$

$$\Delta y = \Delta x \cdot \frac{y}{x - \Delta x}$$

The liquidity providers hold a total percentage of the pool. Let us consider the example of the ETH/USDT liquidity pool. We shall denote the ETH portion of the pool as x and the USDT portion as y . Suppose Alice deposits 1 ETH and 1,000 USDT into a pool on a DEX. Since the token pair must have equivalent value, this means that the price of ETH is 1,000 USDT. At the same time, there are a total of 10 ETH and 10,000 USDT in the pool, with the remainder being provided by other liquidity providers such as Alice. This implies that Alice holds a stake equivalent to 10% of the pool. The total liquidity k in this case is 100,000.

If the price of x relative to y changes, the protocol will be arbitrated [8] by external users so that the ratio between x and y corresponds to the market price. In our example, let us suppose that the price of ETH increases to 4,000 USDT, the total liquidity of the pool must remain constant. If ETH is now worth 4,000 USDT, this means that the ratio between the quantity of ETH and the quantity of USDT in the pool has changed due to adjustments made by the arbitragers. There are now 5 ETH and 20,000 USDT in the pool (we can verify that k remains unchanged, still at 100,000).

Alice therefore decides to withdraw her funds and obtain her 10% share of the total pool, which amounts to 0.5 ETH and 2,000 USDT, or a total of 4,000 USDT.

It appears that she has made a nice profit. But what could have happened if she had not deposited her funds into the pool? She would have had 1 ETH and 1,000 USDT, for a total of 5,000 USDT. In fact, Alice would have been better off keeping her funds in her wallet rather than providing liquidity on a DEX because she has incurred an IL of 20%. Currently, DEXs attempt to address this loss by incentivizing liquidity providers through the collection of fees for each swap, but this is not always sufficient.

1.2 Why is IL a major issue in DeFi?

When a liquidity provider provides liquidity on a DEX, they expect to earn a return through fees. However, on most pairs, these fees represent a low annual return percentage. It is most often the case that the IL is greater than the fees collected over any period measured in the past, and thus liquidity providers may incur a loss. This is a significant problem, as liquidity providers are the foundation of DEXs. Without them, protocols cannot be operational, and if there are not enough incentives to encourage providing liquidity, the DeFi ecosystem is destined to fail.

Some farming protocols, which allow users to earn additional tokens by staking LP-tokens in these protocols, have emerged as a way to better incentivize liquidity providers. However, farming protocols typically mint new tokens infinitely, which means they are unsustainable in the long term. Excessive dilution and constant sell pressure cannot be absorbed by the market, which will negatively affect the price of the tokens.

While farming protocols are an additional incentive, they are not sufficient. Despite these challenges, liquidity providers remain critical to the success of DEXs, and it is essential to continue exploring ways to better incentivize their participation to ensure the continued growth of the DeFi ecosystem.

Therefore, it can be concluded that it is not beneficial for liquidity providers to invest in DEXs that operate in this manner, and the future of these DEXs is uncertain as liquidity will gradually disappear.

2 Defiesta DEX, the solution to IL ?

The Defiesta DEX protocol has been designed based on the traditional DEX model, only the k constant rule has been modified. It is, in fact, a "Algorithmic" DEX that will intelligently manage the x and y liquidity in a way that maintains equilibrium in the long term, while reducing IL and potentially generating IG. Defiesta DEX will manage liquidity to buy low and sell high.

2.1 The Fictive Reserves of Defiesta DEX

Traditional DEXs use their liquidity reserves x and y to calculate the price of a token relative to another. Defiesta DEX introduces FR x_f and y_f. The reserves x and y correspond to the real quantities present on the SC, and the FR x_f and y_f are always equal to a percentage of the reserves. The values of x_f and y_f are initialized to half of the reserves x and y the first time.

It is now the ratio of x_f and y_f that determines the price of one token relative to the other.

$$price_{xiny} = \frac{y_f}{x_f}$$

Let's consider the pool in its initial state just after its creation and the deposit of a small amount of liquidity (X_{init} and Y_{init}). We have:

$$\begin{aligned} x &= X_{init} \\ y &= Y_{init} \\ x_f &= 0.5 \cdot X_{init} \\ y_f &= 0.5 \cdot Y_{init} \\ k_f &= 0.25 \cdot X_{init} \cdot Y_{init} \end{aligned} \tag{1}$$

If a user buys $0.25 \cdot X_{init}$ tokens, by applying the k constant rule on the FR, we found that he will provide $0.5 \cdot Y_{init}$ tokens to the protocol. We then have:

$$\begin{aligned} x &= 0.75 \cdot X_{init} \\ y &= 1.5 \cdot Y_{init} \\ x_f &= 0.25 \cdot X_{init} \\ y_f &= Y_{init} \\ k_f &= 0.25 \cdot X_{init} \cdot Y_{init} \end{aligned} \tag{2}$$

The protocol ends up with more tokens y and fewer tokens x. However, since the price is equal to $\frac{yf}{xf}$, the value of the x reserve in y is:

$$x_{valueiny} = x \cdot \frac{y_f}{x_f} = 0.75 \cdot X_{init} \cdot \frac{Y_{init}}{0.25 \cdot X_{init}} = 3 \cdot Y_{init}$$

We see that $x_{valueiny} > y$ because $3 \cdot Y_{init} > 1.5 \cdot Y_{init}$. And so the protocol has a larger x value and will estimate that it needs to sell some. Thus, it will unbalance the FR to offer trades with more liquidity to buyers of x (and sellers of y) and less liquidity to buyers of y (and sellers of x). Buyers of x will cause lower price impacts and therefore have more advantageous trades.

2.2 Calculation for updating Fictive Reserves

2.2.1 Swap :

During a swap, the algorithm first determines the exposure of reserves to the price. We set:

$$\gamma = \frac{ReserveRatio}{price} = \frac{\frac{y}{x}}{\frac{y_f}{x_f}} = \frac{y \cdot x_f}{x \cdot y_f}$$

We then determine the exposure of the reserves to the swap according to its direction. We set:

$$\phi = \left\{ \begin{array}{ll} \frac{1}{2 \cdot \gamma} & \text{if the user buys x, and } \gamma > 1 \\ \frac{\gamma}{2} & \text{if the user buys y, and } \gamma < 1 \\ \frac{1}{2} & \text{else} \end{array} \right\}$$

• γ denote the direction of imbalance of the FR (depending on whether it is greater or smaller than 1).

• ϕ determines whether the swap is in the direction of imbalance of the FR or not. The FR are then recalculated as follows:

$$x_{fnew} = (x + \frac{y}{price}) \cdot \frac{\phi}{2} = (x + \frac{y \cdot x_f}{y_f}) \cdot \frac{\phi}{2}$$
$$y_{fnew} = (y + x \cdot price) \cdot \frac{\phi}{2} = (y + \frac{x \cdot y_f}{x_f}) \cdot \frac{\phi}{2}$$

Finally, the k constant rule is applied with the FR x_f and y_f . These are further added or subtracted by Δx and Δy , which represent the amounts bought and sold in the swap.

2.2.2 Mint, Burn :

For each Mint or Burn (addition or removal of liquidity), both the FR and the reserves are updated proportionally to the added liquidity. For a Mint ($\Delta lp > 0$) or a Burn ($\Delta lp < 0$):

$$x_{fnew} = x_f \cdot (1 + \frac{\Delta lp}{totalSupplyLp})$$
$$y_{fnew} = y_f \cdot (1 + \frac{\Delta lp}{totalSupplyLp})$$

$$x_{new} = x \cdot (1 + \frac{\Delta lp}{totalSupplyLp})$$
$$y_{new} = y \cdot (1 + \frac{\Delta lp}{totalSupplyLp})$$

Through this mechanism, Defiesta DEX will always manage to adjust its liquidity in such a way as to maintain balance in the x and y reserves according to the price. It will concentrate liquidity on selling when the price rises and on buying when the price drops. This will reduce the IL over the long term and sometimes even generate IG.

3 Defiesta DEX, the algorithm implementation

3.1 Potential vulnerability on the DEX swap

The imbalance of the FR Balance can cause problems.

3.1.1 Explanation :

Indeed, if the reserves in the SC are imbalanced and the SC seeks to sell one of the tokens, it leaves an opportunity for a user to make a swap in one direction and then in the opposite direction. The user would sell at a higher price than they bought, resulting in them receiving more tokens than they spent. This is comparable to a sandwich attack [9] and can lead to financial losses for the liquidity providers.

3.1.2 Example :

Let's consider a pair with Wrapped Ether (WETH) and USDT initialized with 10 WETH and 10,000 USDT (the price is 1:1000). We have:

$$x = 10 \text{ WETH}$$
$$y = 10000 \text{ USDT}$$
$$x_f = 5 \text{ WETH}$$
$$y_f = 5000 \text{ USDT}$$
$$price = \frac{y_f}{x_f} = 1000 \text{ USDT / WETH}$$

As a result of market movements, such as arbitrage, it is possible for an ordinary user to purchase 4 WETH from the protocol. After this transaction, the reserves will be as follows (fees are ignored in the example):

$$\begin{aligned} x &= 6 \text{ WETH} \\ y &= 30000 \text{ USDT} \\ x_f &= 1 \text{ WETH} \\ y_f &= 25000 \text{ USDT} \\ price &= 25000 \text{ USDT / WETH} \end{aligned}$$

The price is highly imbalanced, the protocol now holds 6 WETH worth $x \cdot \frac{y_f}{x_f} = 6 \cdot 25000 =$ 15000 USDT . Since $150000 > 30000$. The protocol will thus seek to massively sell WETH at high prices to balance the books.

A malicious actor enters the system and intends to exploit the protocol for their own gain. Although it would be in their best interest to purchase WETH in order to decrease the price impact of the swap, the user instead chooses to sell, causing a significant drop in price. Specifically, the user sells 2 units of WETH for approximately 7627 USDT, with an average selling price of approximately 3813.5 USDT per WETH. We have:

$$\begin{aligned} x &= 8 \text{ WETH} \\ y &= 22373 \text{ USDT} \\ \text{Fictive reserves re-computed: } x_f &= 0.36 \text{ WETH} \\ \text{Added WETH: } x_f &= 2.36 \text{ WETH} \\ \text{Fictive reserves re-computed: } y_f &= 9000 \text{ USDT} \\ \text{Removed USDT: } y_f &= 1373 \text{ USDT} \\ price &= 581.8 \text{ USDT / WETH} \end{aligned}$$

Subsequently, the malicious actor chooses to repurchase WETH, thereby aligning with the protocol's aim of reducing the price impact and promoting a more gradual price recovery. As a consequence, the average selling price of the asset surpasses the average purchase price. The user acquires 2 WETH for an amount of approximately 6757 USDT, at an average purchase price of approximately 3378.5 USDT/WETH.

$$\begin{aligned} x &= 6 \text{ WETH} \\ y &= 29130 \text{ USDT} \\ \text{Fictive reserves re-computed: } x_f &= 2.42 \text{ WETH} \\ \text{Removed WETH: } x_f &= 0.42 \text{ WETH} \\ \text{Fictive reserves re-computed: } y_f &= 1405 \text{ USDT} \\ \text{Added USDT: } y_f &= 8162 \text{ USDT} \\ price &= 19619.4 \text{ USDT / WETH} \end{aligned}$$

From the perspective of the malicious actor, the transaction involves selling and repurchasing 2 units of WETH, resulting in a net zero position. However, the user has received approximately 7627 USDT and paid approximately 6757 USDT, resulting in a profit of approximately 870 USDT. This profit comes at the expense of a reduction in reserves in y , which decreased from the initial values of $x = 6$, $y = 30000$ to $x = 6$, $y = 29130$. Indeed, this loss is directly imposed on the liquidity providers, who bear the brunt of the negative outcome, even if the market remains unchanged. As a result, the liquidity providers may experience a direct financial loss, which can compromise their ability to continue providing liquidity to the system. This underscores the importance of carefully managing the risks associated with liquidity provision.

3.2 Solution :

It is worth noting that following the first trade by the malicious actor, the price of WETH experiences a significant shift in relation to that of USDT, dropping from 25,000 to approximately 581.8 USDT/WETH in this case. Moreover, the larger the price change, the greater the profit for the user, as the protocol further imbalances the FR balances. This creates an arbitrage opportunity, as an arbitrager who realigns the prices prior to the user's second trade could cause the user to lose their initial investment. However, it is possible to execute both operations in a single transaction via a SC acting on behalf of the user. For this reason, a price variation mechanism utilizing a single moving average per block has been implemented, with a recalculation of the fictive reserve balances triggered only when this moving average is crossed, rather than with each individual swap.

3.2.1 Price Average :

We introduce dt to denote the number of seconds elapsed since the last update of the priceAverage, which is capped at 300 seconds.

$$dt \in [0; 300]$$

The priceAverage is a price that varies based on a 300-second moving average. It is updated only once per block, at the time of the very first transaction of the block, according to the following formula:

$$priceAverage_{new} = \frac{priceAverage \cdot (300 - dt) + price \cdot dt}{300}$$

The $priceAverage$ is initialized with the first price at the time of the first update.