The SMARDEX Protocol: A Novel Solution to Impermanent Loss in Decentralized Finance

Eric Rabl
co-CEO & co-founder
REAL ESTATE EXECUTIVE SA
Montreux, Switzerland

Jean Rausis
co-CEO & co-founder
REAL ESTATE EXECUTIVE SA
Montreux, Switzerland

Stéphane Ballmer
CTO
REAL ESTATE EXECUTIVE SA
Montreux, Switzerland

Abstract

Impermanent Loss (IL [1]) 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 [2]) protocol implemented on SMARDEX 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).

I. Introduction

SMARDEX 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) [3], which is a decentralized software that runs on compatible Ethereum Virtual Machine (EVM) blockchains (such as Ethereum [4] [5] , Binance Smart Chain (BSC), Avalanche, Polygon, etc.). These blockchains are data exchange protocols that, similar to the Bitcoin [6] blockchain, allow for the storage and transmission of information in a public, immutable, and decentralized manner. By using SMARDEX, users can exchange decentralized ERC20 [7] tokens, which are digital assets.

A. 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.

$$k = xy \tag{1}$$

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:

$$\begin{aligned} x_{new} &= x - \Delta x \\ y_{new} &= y + \Delta y \end{aligned} \tag{2}$$

k must remain invariant:

$$k = x \cdot y = x_{new} \cdot y_{new} \tag{3}$$

$$x \cdot y = (x - \Delta x) \cdot (y + \Delta y) \tag{4}$$

$$\Delta y = \Delta x \cdot \frac{y}{x - \Delta x} \tag{5}$$

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.

B. 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.

II. SMARDEX, THE SOLUTION TO IL ?

The SMARDEX protocol has been designed based on the traditional DEX model, only the k constant rule has been modified. It is, in fact, a "SMART" 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. SMARDEX will manage liquidity to buy low and sell high.

A. The Fictive Reserves of SMARDEX

Traditional DEXs use their liquidity reserves x and y to calculate the price of a token relative to another. SMARDEX 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.

$$price_{x\_in\_y} = \frac{y_f}{x_f} \tag{6}$$

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:

$$\begin{aligned} x &= X_{init} \\ y &= Y_{init} \\ x_f &= 0.5 \cdot X_{init} \\ y_f &= 0.5 \cdot Y_{init} \\ k_f &= 0.25 \cdot X_{init} \cdot Y_{init} \end{aligned} \tag{7}$$

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:

$$\begin{aligned} x &= 0.75 \cdot X_{init} \\ y &= 1.5 \cdot Y_{init} \\ x_f &= 0.25 \cdot X_{init} \\ y_f &= Y_{init} \\ k_f &= 0.25 \cdot X_{init} \cdot Y_{init} \end{aligned} \tag{8}$$

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

$$x_{value\_in\_y} = x \cdot \frac{y_f}{x_f} = 0.75 \cdot X_{init} \cdot \frac{Y_{init}}{0.25 \cdot X_{init}} = 3 \cdot Y_{init} \tag{9}$$

We see that $x_{value\_in\_y} > y$ 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.

B. Calculation for updating Fictive Reserves.

1) Swap:

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

$$\gamma = \frac{reserve\_ratio}{price} = \frac{\frac{y}{x}}{\frac{y_f}{x_f}} = \frac{y \cdot x_f}{x \cdot y_f} \tag{10}$$

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

$$\phi = \begin{cases} \dfrac{1}{2 \cdot \gamma} & \text{if the user buys } x\text{, and } \gamma > 1 \\ \\ \dfrac{\gamma}{2} & \text{if the user buys } y\text{, and } \gamma < 1 \\ \\ \dfrac{1}{2} & \text{else} \end{cases} \tag{11}$$

- 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:

$$\begin{aligned} x_{fnew} &= \left(x + \frac{y}{price}\right) \cdot \frac{\phi}{2} = \left(x + \frac{y \cdot x_f}{y_f}\right) \cdot \frac{\phi}{2} \\ y_{fnew} &= (y + x \cdot price) \cdot \frac{\phi}{2} = \left(y + \frac{x \cdot y_f}{x_f}\right) \cdot \frac{\phi}{2} \end{aligned} \tag{12}$$

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) 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 ($\Delta lp < 0$):

$$\begin{aligned} x_{fnew} &= x_f \cdot \left(1 + \frac{\Delta lp}{totalSupplyLp}\right) \\ y_{fnew} &= y_f \cdot \left(1 + \frac{\Delta lp}{totalSupplyLp}\right) \end{aligned} \tag{13}$$

$$\begin{aligned} x_{new} &= x \cdot \left(1 + \frac{\Delta lp}{totalSupplyLp}\right) \\ y_{new} &= y \cdot \left(1 + \frac{\Delta lp}{totalSupplyLp}\right) \end{aligned} \tag{14}$$

Through this mechanism, SMARDEX 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. SMARDEX 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.

III. SMARDEX, THE ALGORITHM IMPLEMENTATION

A. *Potential vulnerability on the SMARDEX swap*

The imbalance of the FR balance can cause problems.

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.

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:

$$\begin{aligned} x &= 10 \textit{ WETH} \\ y &= 10,000 \textit{ USDT} \\ x_f &= 5 \textit{ WETH} \\ y_f &= 5,000 \textit{ USDT} \\ price &= \frac{y_f}{x_f} = 1,000 \textit{ USDT / WETH} \end{aligned} \tag{15}$$

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):

$$\begin{aligned} x &= 6 \textit{ WETH} \\ y &= 30,000 \textit{ USDT} \\ x_f &= 1 \textit{ WETH} \\ y_f &= 25,000 \textit{ USDT} \\ price &= 25,000 \textit{ USDT / WETH} \end{aligned} \tag{16}$$

The price is highly imbalanced, the protocol now holds 6 WETH worth $x \cdot \frac{y_f}{x_f} = 6 \cdot 25,000 = 150,000$ USDT. Since $150,000 > 30,000$. 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 7,627 USDT, with an average selling price of approximately 3,813.5 USDT per WETH. We have:

$$\begin{aligned} x &= 8 \textit{ WETH} \\ y &\approx 22,373 \textit{ USDT} \\ \textit{Fictive reserves re-computed:} \quad x_f &= 0.36 \textit{ WETH} \\ \textit{Added WETH:} \quad x_f &= 2.36 \textit{ WETH} \\ \textit{Fictive reserves re-computed:} \quad y_f &= 9,000 \textit{ USDT} \\ \textit{Removed USDT:} \quad y_f &\approx 1,373 \textit{ USDT} \\ price &\approx 581.8 \textit{ USDT / WETH} \end{aligned} \tag{17}$$

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 6,757 USDT, at an average purchase price of approximately 3,378.5 USDT/WETH.

$$\begin{aligned} x &= 6 \textit{ WETH} \\ y &\approx 29130 \textit{ USDT} \\ \textit{Fictive reserves re-computed:} \quad x_f &\approx 2.42 \textit{ WETH} \\ \textit{Removed WETH:} \quad x_f &\approx 0.42 \textit{ WETH} \\ \textit{Fictive reserves re-computed:} \quad y_f &\approx 1,405 \textit{ USDT} \\ \textit{Added USDT:} \quad y_f &\approx 8162 \textit{ USDT} \\ price &\approx 19,619.4 \textit{ USDT / WETH} \end{aligned} \tag{18}$$

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 7,627 USDT and paid approximately 6,757 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.

B. 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.

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.

$$dt \in \ [0\,;\,300] \tag{19}$$

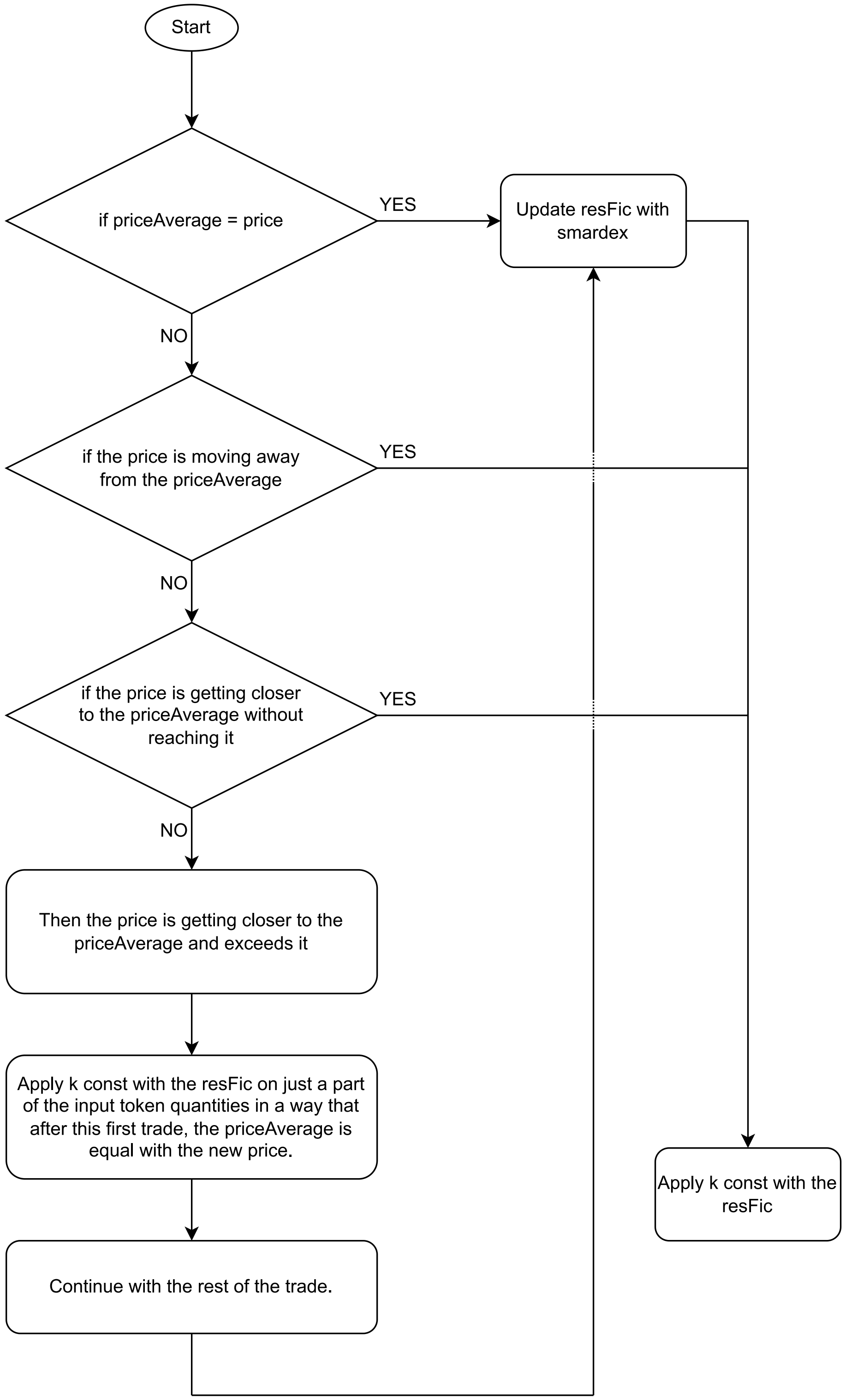
The priceAverage is a price that varies based on a 300-second moving average. It is updated only once per block, at the time of the very first transaction of the block, according to the following formula:

$$priceAverage_{new} = \frac{priceAverage \cdot (300 - dt) + price \cdot dt}{300} \tag{20}$$

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

2) SMARDEX's calculation algorithm:

The following algorithm is applied to recalculate the swap:


Start
if priceAverage = price
YES
Update resFic with smardex
NO
if the price is moving away from the priceAverage
YES
NO
if the price is getting closer to the priceAverage without reaching it
YES
NO
Then the price is getting closer to the priceAverage and exceeds it
Apply k const with the resFic on just a part of the input token quantities in a way that after this first trade, the priceAverage is equal with the new price.
Continue with the rest of the trade.
Apply k const with the resFic

In this way, it is not possible to take advantage of the previously mentioned exploit. Indeed, if the user tries to do so anyway, they will first make a trade that will significantly unbalance the pair, and the FR will be recalculated because at the beginning, the price and the priceAverage are the same.

The user will then make the opposite trade in the same block, but the priceAverage will not have been updated yet (1 update per block), so the k constant rule will be applied to return to the price. At best, the user will raise the price back to what it was before, but will have to pay protocol fees and will make a loss on their transaction.

C. Verification of the solution

If we take the above example again (18), the malicious actor no longer benefits from this manipulation. Indeed, at the time of buying back WETH, the priceAverage will not have moved because it is in the same block. The price will therefore return exactly to what it was before their first transaction, except that the fees were ignored. Taking those into account, the actor will incur a loss in USDT.

If a malicious actor wishes to benefit from this kind of manipulation, they would be smart to wait for a few blocks between the two trades in order to get the priceAverage closer to the price. Doing so would give them the chance to make a profit from the situation. The closer the priceAverage gets to the price, the higher the potential gain the actor can make since the protocol will execute the following trade: Apply the k rule until the price is increased to the priceAverage, then recalculate the FR and finally use the k rule on the new FR.

It is by taking advantage of this new FR that the user can make a profit, so they should try to make the biggest possible trade. However, the longer they wait between blocks, the more they risk being affected by arbitragers, which could lead to them losing everything. Additionally, across several blocks, they might not be able to use a flashloan [10] . A flashloan is a powerful tool that allows people to borrow a large number of tokens without needing to put up collateral. This can be used for arbitrage opportunities and gives users the ability to leverage huge amounts of tokens. The downside is that the tokens must be returned in the same transaction and hence the same block. Therefore, this solution is not feasible in this case.

IV. Results

From 2018 to December 2022, the ETH/USDT pair on Binance [11] was simulated to be arbitrated every 15 minutes with UNISWAP v2 and SMARDEX pools.

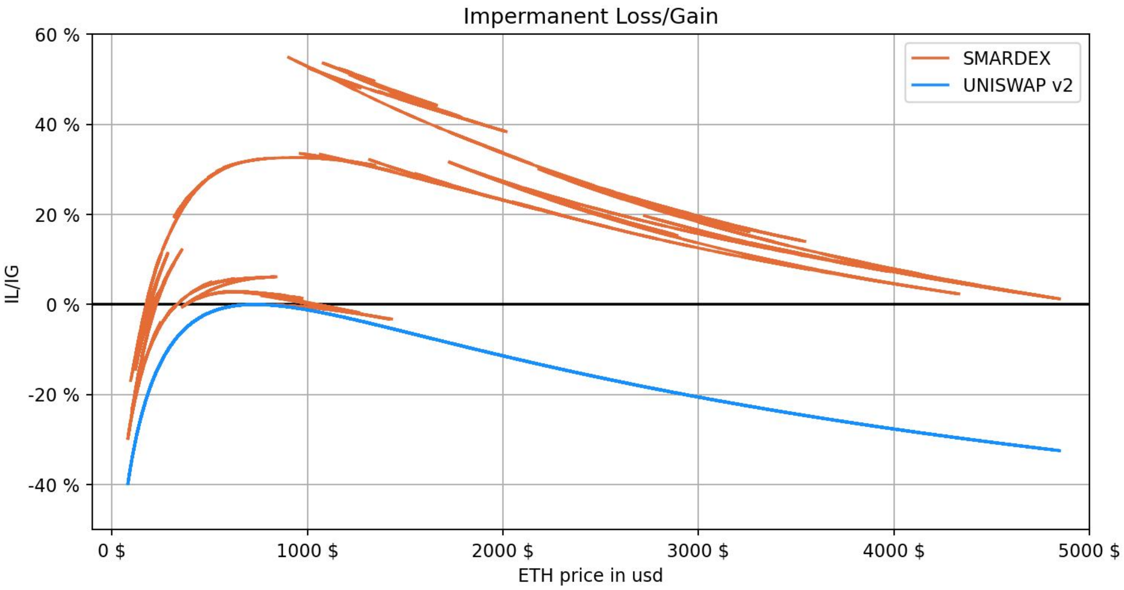


Fig. 1: IL comparison between SMARDEX and UNISWAP v2 [12], over the ETH from 2018-2023

Furthermore, the 50 biggest tokens by market size, excluding stablecoins, were simulated during this period, and all pairs were arbitrated every 15 minutes if they were listed on Binance as of January the 1^{st}, 2018. If not, the simulation began on the date indicated in the first column. The simulation ended on January the 1^{st}, 2023. Columns 3 and 4 display either a negative value (IL) or a positive value (IG). Last column displays the price variation over the period. Tokens that have been identified as highly manipulated were removed from the simulation.

Start date	Binance data	SMARDEX +IG or -IL	Normal DEX +IG or -IL	Token price var.
01.01.2018	BTC	30.00%	-0.49%	22%
01.01.2018	ETH	51.39%	-2.87%	62%
01.01.2018	BNB	-35.59%	-64.23%	2,823%
01.01.2018	LTC	18.99%	-16.31%	-71%
17.04.2018	ADA	80.08%	-0.02%	-4%
11.06.2018	TRX	28.33%	-0.15%	11%
26.04.2019	MATIC	-60.49%	-82.75%	13,148%
05.07.2019	DOGE	-5.39%	-56.29%	1,788%
18.08.2020	DOT	37.84%	-2.05%	50%
10.05.2021	SHIB	15.95%	-17.60%	-72%
11.08.2020	SOL	51.20%	-18.66%	278%
19.09.2020	UNI	32.19%	-4.68%	87%
23.09.2020	AVAX	56.15%	-9.01%	142%
17.01.2019	LINK	-18.28%	-46.11%	1,069%
15.03.2019	XMR	4.52%	-11.82%	178%
29.04.2019	ATOM	34.48%	-7.42%	122%
12.06.2018	ETC	62.53%	0.00%	-1%
31.05.2018	XLM	7.45%	-20.33%	-75%
28.11.2019	BCH	28.19%	-0.16%	-11%
22.06.2019	ALGO	-26.98%	-54.49%	-94%
17.02.2022	APE	0.87%	-9.28%	-59%
25.07.2018	VET	76.44%	-1.59%	-30%
14.10.2020	NEAR	46.03%	-0.02%	-4%
29.09.2019	HBAR	51.40%	0.00%	0%
15.10.2020	FIL	-44.81%	-62.80%	-96%
28.05.2018	EOS	-31.35%	-50.46%	-93%
03.09.2020	EGLD	59.70%	-0.72%	27%
15.10.2020	AAVE	32.57%	-2.17%	-34%
11.04.2019	THETA	16.31%	-28.98%	476%
24.09.2019	XTZ	26.99%	-1.78%	-32%
21.03.2019	ZEC	31.26%	-2.17%	-34%
04.11.2020	AXS	-32.54%	-66.57%	3,277%
06.09.2019	CHZ	11.21%	-44.35%	982%
14.08.2020	SAND	31.38%	-27.46%	442%
27.01.2021	TWT	-0.78%	-23.69%	365%
06.08.2020	MANA	18.74%	-25.59%	403%
11.06.2019	FTM	96.10%	-26.00%	411%
	Average:	21.14%	-21.33%	

TABLE I: IL simulation of multiple crypto-currencies with an arbitrage every 15min

The SMARDEX protocol has an average gain (IG) of +21.14%, as seen in the table above, compared to a typical DEX where the loss (IL) is an average -21.33%. This simulation does not take into account the calculation of fees. An investor would have had, on average, saved 42.47% more value since 2018 if they had used our protocol rather than in a typical DEX.

Disclaimer: The present simulation showcases the efficiency of the SMARDEX protocol. It has been chosen by our team for this very reason. A large number of simulations have been conducted to test the performance of the SMARDEX protocol in comparison to other decentralized exchanges. In every instance (100%), the results have favored SMARDEX over its peers, indicating that it consistently outperforms other DEXs. It is important to note that the simulations were conducted with rigorous and standardized methodologies to ensure accurate and unbiased results. Furthermore, the simulations were performed using real-world data to reflect actual market conditions.

The positive outcomes of the simulations strongly suggest that SMARDEX provides a superior trading experience compared to other DEXs. Interested parties can test the protocol in real-world scenarios [13] to confirm the positive outcomes observed in the simulations. Conducting live tests is a recommended approach to validating the efficacy of a protocol as it allows researchers to observe the protocol in action in real-time, which can help to detect any potential issues or limitations that may have been overlooked during the simulation phase.

Additionally, live tests can help to uncover nuances in the protocol's functionality that may not be apparent in a simulated environment. Overall, verifying the results of the simulations through live testing is an essential step in establishing the reliability and validity of the findings and ensuring that users can trust the protocol's performance in actual use cases.

V. Conclusion

As shown above, SMARDEX has introduced a simple yet efficient technology that optimizes IL and can even generate IG in many cases. Based on the various scenarios presented, it seems clear that the technology works and that old-generation DEXs like UniSwap v2 will quickly become obsolete. The success of the SMARDEX protocol highlights the importance of continuing research and innovation in the field of DeFi. It opens up new possibilities for DeFi, providing a promising avenue for users, developers, and investors. The next step is to encourage the adoption of this new technology and explore its full potential.

Acronyms

AMM	Automated Market Maker.
BSC	Binance Smart Chain.
DeFi	Decentralized Finance.
DEX	Decentralised Exchange.
ERC20	Ethereum Request for Comment - fungible tokens.
ETH	Ether.
EVM	Ethereum Virtual Machine.
FR	Fictive Reserve.
IG	Impermanent Gain.
IL	Impermanent Loss.
LP	Liquidity Provider.
SC	Smart Contract.
USDT	Tether USD, an ERC20 crypto-currency pegged to the US dollar.
WETH	Wrapped Ether.

References

[1] A. A. Aigner and Gurvinder Dhaliwal, “Uniswap: Impermanent loss and risk profile of a liquidity provider,” 2021.

[2] J. Xu, K. Paruch, S. Cousaert, and Y. Feng, “SoK: Decentralized exchanges (DEX) with automated market maker (AMM) protocols,” *ACM Computing Surveys*, 2022.

[3] S. Wang, Y. Yuan, X. Wang, J. Li, R. Qin, and F.-Y. Wang, “An overview of smart contract: Architecture, applications, and future trends,” pp. 108–113, 2018.

[4] G. Wood, “Ethereum yellow paper: A secure decentralised generalised transaction ledger,” 2014.

[5] V. Buterin, “Ethereum white paper: A next generation smart contract & decentralized application platform,” 2013.

[6] S. Nakamoto, “Bitcoin: A peer-to-peer electronic cash system,” 2008.

[7] R. Rahimian and J. Clark, “Tokenhook: Secure erc-20 smart contract,” 2021.

[8] Y. Wang, Y. Chen, H. Wu, L. Zhou, S. Deng, and R. Wattenhofer, “Cyclic arbitrage in decentralized exchanges,” p. 12–19, 2022.

[9] L. Zhou, K. Qin, C. F. Torres, D. V. Le, and A. Gervais, “High-frequency trading on decentralized on-chain exchanges,” *CoRR*, vol. abs/2009.14021, 2020.

[10] D. Wang, S. Wu, Z. Lin, L. Wu, X. Yuan, Y. Zhou, H. Wang, and K. Ren, “Towards a first step to understand flash loan and its applications in defi ecosystem,” p. 23–28, 2021.

[11] https://binance.com

[12] D. R. Hayden Adams, Noah Zinsmeister, “Uniswap v2 core,” 2020.

[13] https://smardex.io, “Smardex, decentralized finance becomes smart,” 2023.