Invariant: First AMM DEX providing concentrated liquidity on Solana

Norbert Bodziony
norbert@akudama.xyz

Wojciech Cichocki
wojciech.cichocki@invariant.app

Mateusz Zając
mateusz.zajac@invariant.app

Maciej Zięba
maciej.zieba@invariant.app

December 6, 2021

Abstract

The Invariant protocol is a peer-to-peer system for exchanging assets on the Solana blockchain. The protocol is implemented as a set of smart contracts that prioritize censorship resistance, security, self-custody, and the ability to operate without the use of trusted intermediaries who can selectively restrict access.

1 Introduction

While Solana [1] has multiple exchange protocols, the most advanced ones appear to focus on other blockchains. That is something we want Invariant to change. We decided to bring Uniswap (see [2], [3]) to Solana because its concept of providing liquidity only within a certain range seemed the most promising.

At first glance, Invariant may appear to be another exchange, complete with liquidity providers, swaps, and trade fees. The distinction is in the ability to provide liquidity only within certain price ranges. This has the effect of keeping liquidity where the price is, and thus where it is most needed.

Because liquidity is more efficient on *Invariant*, it stands to reason that trades will benefit from it. It results in a lower fee on trades by an order of magnitude. By default, 0.04 percent will be set as the default value. Furthermore, users benefit from increased virtual liquidity, which results in less slippage on trades.

Liquidity concentration enables much higher efficiency than traditional AMMs. In other words, producing the same amount of liquidity requires fewer tokens. Keeping liquidity close to the price can amplify this effect even more.

2 Theory

In *Invariant*, liquidity can be provided to almost any price interval, which is referred to as a position. If the price remains in the interval $[p_l, p_u]$, the liquidity assigned to position $[p_l, p_u]$ earns fees. Token currency pairs can be exchanged for each other using liquidity pools that contain both tokens. The reserve curve determines which transactions are permitted.

$$xy = L^2 = \text{constant}.$$

The price of a pair is defined as the negative gradient of the slope at a point (x, y), i.e.

$$p = -\frac{\partial y}{\partial x} = \frac{L^2}{x} = \frac{y}{x}.$$

Each point on a curve $xy = L^2$ can be parameterized using the variable p in the following way:

$$\varphi : (x, y) \longmapsto (\frac{L}{\sqrt{p}}, L\sqrt{p})$$

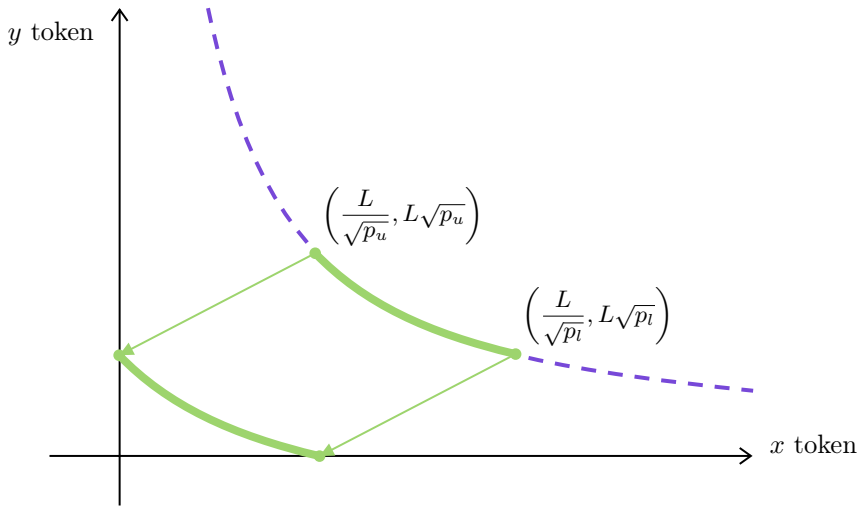

Figure 1: Translation of a curve

We can create an associated curve using translation for a curve $xy = L^2$ and a specified price interval $[p_l, p_u]$.

$$T(x, y) = (x + \frac{L}{\sqrt{p_u}}, y + L\sqrt{p_l})$$

Above translation define a new curve $\widetilde{x}\widetilde{y} = L^2$, where:

- $\widetilde{x} := \left(x + \frac{L}{\sqrt{p_u}}\right)$
- $\widetilde{y} := (y + L\sqrt{p_l})$

When $p \in [p_l, p_u]$, then we have following identity:

$$\widetilde{x}\widetilde{y} = L^2 \implies \begin{cases} p\widetilde{x} = \widetilde{y} \\ \widetilde{y} = \sqrt{p}L \end{cases} \implies \begin{cases} x = L\frac{\sqrt{p_u} - \sqrt{p_c}}{\sqrt{p_c}\sqrt{p_u}}, \\ y = L(\sqrt{p_c} - \sqrt{p_l}) \end{cases}$$

As a result, Invariant is capable of supporting a broad range of liquidity management strategies. In order to earn fees, you must have liquidity L and the price must be within the range $[p_l, p_u]$. Whenever the price falls outside of the range, you will not be paid any commissions until the price returns to the range.

Except for when prices are fluctuating within the price range, all liquidity is provided by a single asset, which is either X or Y, depending on which side of the price range it is currently trading on. We have the following four alternatives:

1. Current price p_c is in the range: $p_c \in [p_l, p_u]$, then values of x or y can be simply calculated using following formulas

$$x = L\frac{\sqrt{p_u} - \sqrt{p_c}}{\sqrt{p_c}\sqrt{p_u}},$$
$$y = L(\sqrt{p_c} - \sqrt{p_l}).$$

2. Current price p_c is in the range: $p_c \in (0, +\infty)$, then using the formulas above, x or y values can be easily obtained by

$$x = \lim_{p_u \to \infty} L\frac{\sqrt{p_u} - \sqrt{p_c}}{\sqrt{p_c}\sqrt{p_u}} = \frac{L}{\sqrt{p_c}},$$
$$y = \lim_{p_l \to 0} L(\sqrt{p_c} - \sqrt{p_l}) = L\sqrt{p_c}.$$

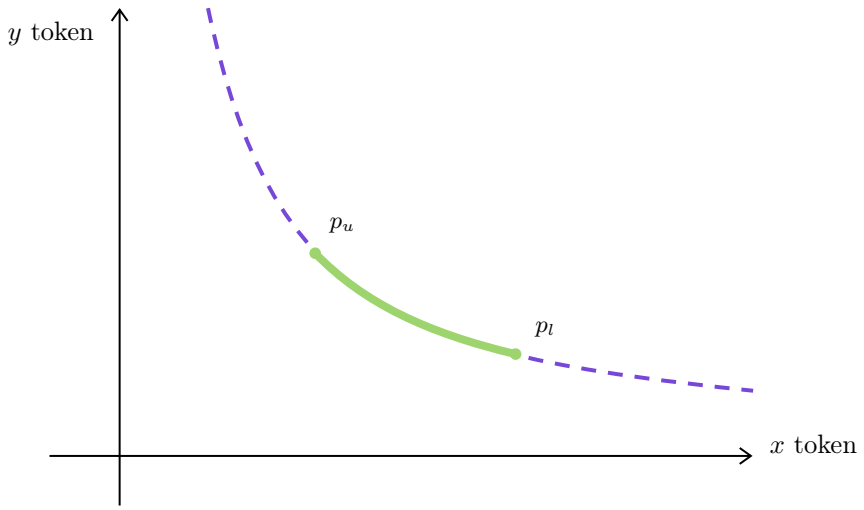

Figure 2: Single position

3. Current price p_c is smaller than lower price: $p_c < p_l$, then

$$x = L\frac{\sqrt{p_u} - \sqrt{p_l}}{\sqrt{p_l} \cdot \sqrt{p_u}},$$
$$y = 0$$

and the liquidity:

$$L = x\frac{\sqrt{p_l} \cdot \sqrt{p_u}}{\sqrt{p_u} - \sqrt{p_l}}.$$

4. Current price p_c is greater than upper price: $p_u < p_c$, then

$$x = 0,$$
$$y = L(\sqrt{p_u} - \sqrt{p_l})$$

and the liquidity:

$$L = \frac{y}{\sqrt{p_u} - \sqrt{p_l}}.$$

2.1 Ticks

To facilitate the selection of the price range, *ticks* are introduced.

Consider a following function

$$\tau : \mathbb{Z} \to \mathbb{R}_+,$$
$$\tau : i \mapsto 1.0001^i.$$

A i-th tick is a value of function τ for argument i, for example:

- $\tau(0) = 1$,
- $\tau(365) = 1.03717$,
- $\tau(-365) = 0.96416$,

Every time a price is calculated by multiplying it by the previous price by 1.0001, then the price change is always the same as 0.01%.

There is also the concept of *tick spacing* for each pool. Tick spacing, $\mathcal{T}_s$ is an additional grid on top of tick spacing that limits the number of ticks in which liquidity can be placed. For given tick spacing you can only consider ticks $\tau(i)$, when $\mathcal{T}_s | i$.

Liquidity providers can provide liquidity in a range between two tick, if their spacing is at least 2.

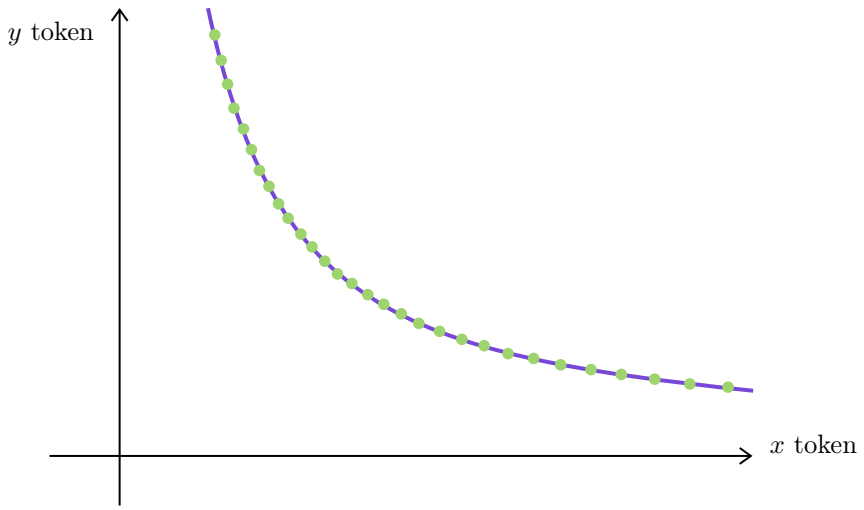

Figure 3: Visualisation of ticks

2.2 Slippage

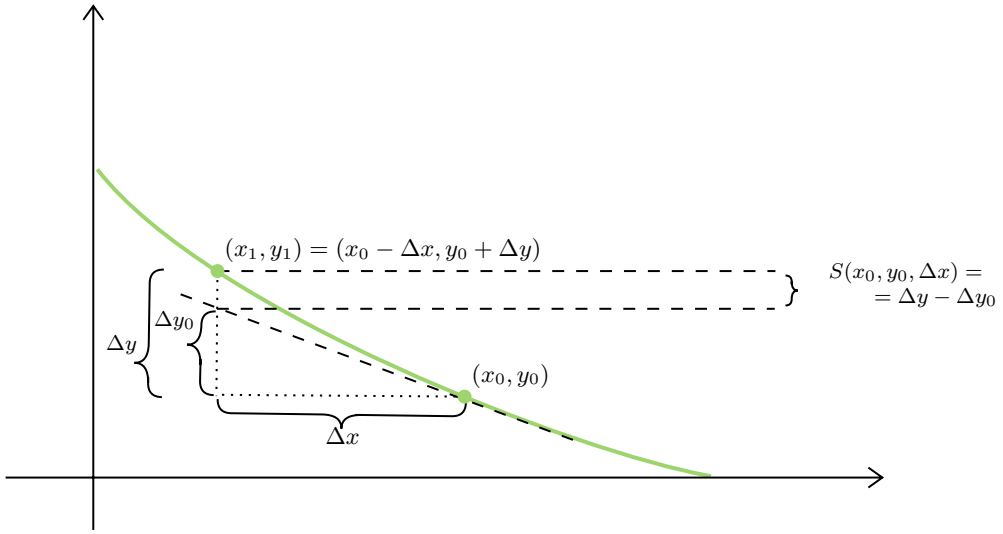

Figure 4: Illustration of price slippage on a trade

Slippage is an important consideration when approaching swaps with the *Invariant* protocol, and it should not be overlooked. Slippage is the term we use to describe price changes that may occur while a transaction is in the process of being processed (see Figure 4 for an example). However, only within a specific range of $[p_l, p_u]$ should the slippage be of the same magnitude as in the reserve curve. When $p_u - p_l \to \infty$, the slippage function approximates a reserve curve in the same way. One more instance: If the $p_u - p_l \to 0$, then slippage is reduced to 0. Of course, slippage and liquidity are inversely proportional to one another.

Slippage tolerances define a range of change that is acceptable to the user beyond the influence on the pricing. Whenever the execution price falls within the slippage range, for example, 0.5%, the deal will be completed. If the execution price falls outside of the acceptable slippage range, the transaction will fail, and the swap will not take place as a result of the failure.

2.3 Fees

Fee, φ is a mathematical representation of the fee paid by swappers in hundredths of a percent, and it is initialized with a predetermined value in each pool. Two numbers `fee_growth_global_x` and

`fee_growth_global_y` are the global fees (in x and y) that LPs have accrued. When a swap occurs, the values of all of the above variables change. However, only L changes when liquidity is provided or removed.

When the tick is crossed, the contract must keep track of the amount of gross liquidity that should be added or withdrawn, as well as the fees received above and below the tick, in order to be effective. When the tick indexes are updated, the variables in the tick-indexed state are updated. Consider that, after updating the contract's global state, the pool changes the fees collected and liquidity at the precise price point, which is τ_u (upper tick) and τ_l (lower tick) in the contract's global state (lower tick).

It also keeps track of the current protocol fee, denoted by the symbol Φ. It is set to a constant value of 0.1 and generates a portion of the swapper fees that are currently going to protocol rather than liquidity providers.

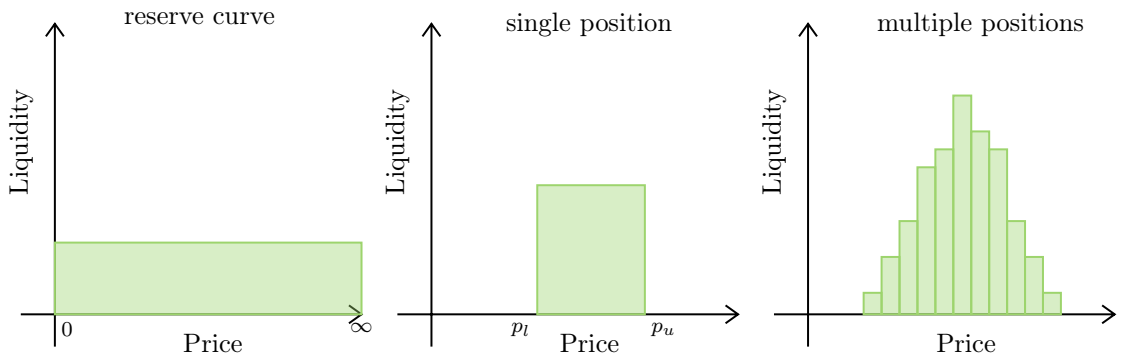

Figure 5: Liquidity distributions

References

[1] Anatoly Yakovenko, Solana: A new architecture for a high performance blockchain v0.8.13.

[2] Hayden Adams, Noah Zinsmeister, Moody Salem, River Keefer and Dan Robinson. Uniswap v3 Core (2021).

[3] Hayden Adams, Noah Zinsmeister and Dan Robinson. Uniswap v2 Core (2020).

[4] Bhaskar Krishnamachari, Qi Feng and Eugenio Grippo. Dynamic Curves for Decentralized Autonomous Cryptocurrency Exchanges. `arXiv:2101.02778v1`

[5] D. Senchenko. Impernament Losses in Uniswap-Like Markets. (2020). `dsenchenko.medium.com`