Economic design and incentives:

Uniswap, a decentralised cryptocurrency exchange

Introduction

This paper is focused on Uniswap, a decentralised cryptocurrency exchange implementing an automated market maker system deployed on the blockchain. The contents are split into two parts. The first concentrates on the economic design of Uniswap, exploring the mathematical models underpinning a system enabling swaps of cryptocurrency tokens and provision of liquidity. This description is structured around Uniswap's evolution from its beta to the latest version. The second part examines some of the concepts surrounding liquidity provision, notably the generation of fees and idea of impermanent loss. This draws on research formalising these concepts and presents some of main strategies for maximising rewards.

Contents

1. Background

Since Bitcoin came to the fore in 2008 cryptocurrencies have been obtained by users in two major ways: through mining or through exchange. Mining is a process in which tokens are earnt for providing a proof-of-work or proof-of-stake while carrying out some kind of task for a blockchain network. Exchange is the manner in which cryptocurrencies or crypto assets are swapped or traded, one for another, much like the barter of one good for another, or like a traditional stock exchange. We can group these types of exchange into two groups, swapping cryptocurrency tokens for traditional fiat currencies, for example buying Bitcoin for euros, or trading one cryptocurrency token for another, for instance swapping Ether for Bitcoin.

The exchange of crypto assets was initially focused on centralised exchanges much like traditional stock exchanges. These exchanges have evolved as the blockchain ecosystem itself has grown and developed, but this has not been without a number of bumps along the road. From the early days, centralised exchanges have been subject to similar problems and fraud that have dogged traditional financial markets, except that these issues are often woven into the highly technical and sophisticated implementation of blockchain networks.

Mt. Gox, a Japanese cryptocurrency exchange, was one of the earliest centralised bourses to achieve widespread use, reportedly processing the majority of the world's Bitcoin

transactions by 2013. However, in 2014 the website stopped all withdrawals amid the loss or theft of hundreds of thousands of cryptocurrency tokens amounting to hundreds of millions of US dollars. If we fast forward to now, the names of the centralised exchanges are different, but the problems are essentially the same. FTX, an exchange based in the Bahamas, went into liquidation in November 2022 with billions of dollars allegedly siphoned off into another company, closely linked to the exchange, that had been acting as a market maker. The collapse sent shockwaves through the cryptocurrency universe.

The issues with centralised exchanges on the one hand point to some of the time old problems of fraud and criminality. On the other hand, advocates for stronger regulation, would point to the need for stronger rules and laws surrounding cryptocurrency markets. Meanwhile, the flagbearers for the ideals of a decentralised financial system would instead use these problems surrounding centralised exchanges as opportunity to argue for a decentralised solution.

1.1 Centralised exchanges (CEXs)

Before we explore a decentralised exchange, it is important to briefly outline the economic model used by centralised exchanges. The principal way in which centralised exchanges operate is through an order book system or what is known as a double auction. Buyers and sellers set their respective prices to buy or sell, often known as the bid or ask, and a trade takes place when these prices intersect. The double auction is a continuous process and in the era of high frequency trading takes place at a rate of fraction of seconds. Traditional stock exchanges operate within a defined trading period, usually a weekday during office hours, however, the world of cryptocurrencies has paved the way for continuous 24-hour trading.

The economic motivations for various agents are clearly defined, and although there are differences and variations to the model, the overall structure is as follows:

- The person who wants to carry out a trade on the exchange pays a fee to the exchange operator.
- The exchange operator sets a fee structure usually as a percentage of the trade or as a flat rate fee.
- Market makers operate on the exchange earning a difference between the bid and the ask, known as the spread, as compensation for providing liquidity.

There are some differences within centralised crypto currency exchanges, for example, some exchanges may perform market making activities themselves and there are not the same regulatory requirements as on most traditional stock exchanges. But for the most part the payoffs for offering or interacting with the exchange are focused in the same way on the operation of a centralised order book, providing the framework for incentivising agents to interact with the buying and selling of assets.

1.2 Motivation for Decentralised Exchanges (DEXs)

As well as facilitating the trade of cryptocurrencies, centralised exchanges also carry out basic custody. In order to interact with the order book and carry out a trade the assets must reside at some point within the exchange’s system. In the traditional world of finance, at least for retail investors, custody and trading has become somewhat synonymous. If an individual opens a brokerage account for trading shares in a company, they deposit fiat currency, buy

the shares through the broker, and then hold the assets through that same brokerage account. In the future, this individual can receive dividends for their holding or decide to sell the shares. In the retail setting, it is unlikely that the individual will actually pay for the actual share certificates, although brokerages do provide this service. In the cryptocurrency universe, the same system is in place, but crypto assets are held within so-called wallets which store the cryptographic keys that essentially provide ownership of the asset.

An individual using a cryptocurrency centralised exchange may deposit fiat currency in their account and then trade that for Bitcoin, Ether or any number of other crypto assets. Once they carry out that trade, they have the option to keep the assets within the centralised exchange's wallet or transfer it out. They may hold another cryptocurrency wallet or want to transfer the tokens to pay for something. The key point is that whilst the centralised exchange has custody over the cryptographic keys the user only has so much control over the assets.

This idea of custody is one of the central themes and motivations for the rise of DEXs. Given the number of failures impacting centralised exchanges, the clarion call for DEXs has intensified. The idea of a DEX stems from the very decentralised philosophy at the core of cryptocurrencies: having no centralised authority, but instead a system of disparate nodes who arrive at a consensus on transactions interacting with the blockchain. As such, we are only concerned here with exchanges or swaps of pure crypto assets. DEXs do not swap crypto tokens for fiat currency because the blockchain itself is not interacting directly with the traditional banking system. You cannot buy Bitcoin with euros on a DEX, although we must be clear there exist tokens provided by some blockchain protocols that offer a cryptographic representation of fiat currencies such as USD Tether ($USDT$) or USD Circle ($USDC$), that are intended to be backed by real US dollars.

Returning to the custody question, DEXs imply that you retain control over your cryptographic keys. When you interact with a DEX you use your own wallet to sign transactions. So, if you deposit assets or carry out a swap on a DEX you give permission to the protocol by authorising your tokens with your cryptographic keys to be used by the underlying smart contract. We must point out that this is not entirely risk free, instead the complexity of custody is now something you must manage yourself, rather than delegating that custody to a supposedly responsible centralised exchange.

1.3 Uniswap

At this point it is important to understand how a DEX, or in our case Uniswap, works in terms of its basic operation. It is a smart contract which exists on the Ethereum blockchain, a distributed system of computers storing a copy of a ledger. A smart contract resides on the ledger and is a software programme written so that it automatically carries out transactions according to a pre-defined set of criteria or rules. Once the smart contract is deployed on the blockchain the code within it cannot be updated or changed, although it can hold different variables or states. Transactions interacting with the smart contract are also stored on the blockchain and as such there is a level of transparency and recorded history that is immutable.

Uniswap at its core is a software programme and as such developers can interact with it through a number of functions and methods that are exposed by smart contracts. There are actually a number of smart contracts that make up Uniswap, but for our purposes we can

imagine the Uniswap code as acting as one single entity. It also provides an application and user interface that allow people to interact with it visually, just the same as any other web app. Version one was released in 2018 and since then it has gone through two further iterations, with version three marking it as one of the most popular decentralised exchanges, claiming to have facilitated some 1.3 trillion US dollars in trade volume.

There is a company behind Uniswap and a number of top venture capitalists, but the core functionality requires no oversight or centralised authority. If the company or founders disappeared tomorrow, the web app might stop working, but the smart contract would live on, and asset holders could continue to transact as long as the Ethereum blockchain remained intact.

2. How Uniswap works

Uniswap is centred around the idea of a Constant Function Market Maker (CFMM), an idea that surfaced within the crypto community in 2017, having been borrowed from the idea of Hanson's logarithmic market scoring rule for prediction markets (Hanson, 2002). The model removes the need for a centralised order book and instead puts emphasis on pools of liquidity. These types of systems are known as automated market makers since they operate according to a set of pre-defined rules. The liquidity is constrained according to a mathematical model, in this case the constant product function working on the basis of inverse proportionality. The various versions of Uniswap have built upon this core idea and developed the model to add additional features and changed the implicit system of payoffs and penalties. In this chapter, we will describe how each version of Uniswap was designed, outlining the models, and illustrating them with some examples.

2.1 Uniswap v1

The original version of Uniswap was launched in November 2018 and was designated version 0, denoting it as a beta version (Adams, A short history of Uniswap, 2019). An audit and formal specification were written by Runtime Verification Inc, a company working in the blockchain field (Park, Zhang, & Cheng, 2018). We will start our examination of the Uniswap model using this formal specification.

2.1.1 Overview and swaps

Uniswap is based on the concept of an inversely proportional constant function describing the reserves of two cryptocurrency tokens, x and y, with an invariant k.

$$x.y = k$$

This relationship is visually represented on a graph as follows:

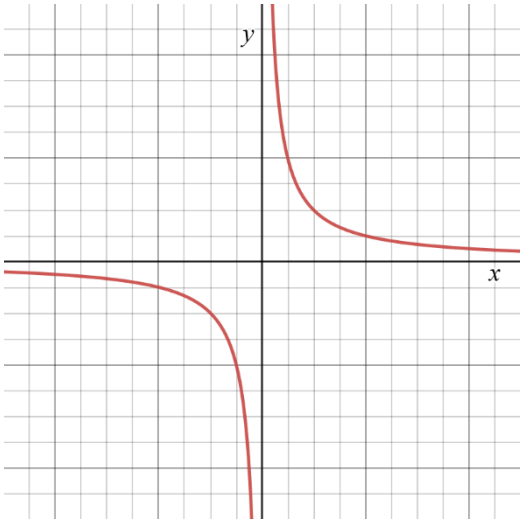

Figure 1 Rectangular hyperbola, via Desmos.com

Since we can only have positive amounts of tokens, we are only interested in Quadrant I.

$$\{x, y > 0\}$$

The main idea involves how a swap between a pair of tokens changes our position on the hyperbola. Given that k will remain constant this provides us with the following relationship:

$$x.y = (x + \Delta x) \times (y - \Delta y) \quad (1)$$

With x being the token added to the pool and y being the token being removed from the pool.

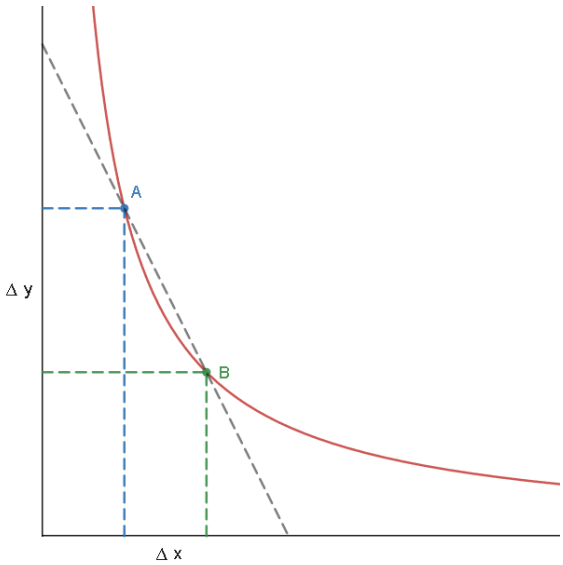

Figure 2 Graph representing a swap, via Desmos graphing tool.

The reserves in the liquidity pool move from point A to point B as the swap is carried out.

The grey dotted line represents the slope between point A and point B and provides us with the price of the swap in units of x.

$$Price = \frac{\Delta y}{\Delta x}$$

Equally, we can calculate the price in units of y using the reciprocal:

$$Price = \frac{\Delta x}{\Delta y}$$

This pricing formula is like one for the marginal cost in economics, but in this case, it is the marginal price. To avoid confusion later we will call this the execution price. The price at a given moment is not actually the price achieved through a swap since it represents a shift along the curve. The price at point A without any swap occuring can be considered the theoretical equilibrium price and is calculated from the liquidity reserves of x and y:

$$Price = \frac{x}{y}$$

The equation (1) can be rearranged to express it as Δx and Δy so that we have a simple formula to calculate the number of tokens resulting from a swap Rearrangement of equation making $\Delta \boldsymbol{x}$ and $\Delta \boldsymbol{y}$ the subject..

$$\Delta x = \frac{\Delta y . x}{y - \Delta y} \quad (2)$$

$$\Delta y = \frac{\Delta x . y}{x + \Delta x} \quad (3)$$

The formal overview by Runtime Verification Inc uses a slightly different approach and rewrites the equation substituting in α and β. The idea here is that these values provide a ratio of the token being added or removed to the overall reserves. Their equations are as follows:

$$x' = x + \Delta x = (1 + \alpha)x = \frac{1}{1 - \beta}x$$
$$y' = y - \Delta y = \frac{1}{1 + \alpha}y = (1 - \beta)y$$

where $\alpha = \frac{\Delta x}{x}$ and $\beta = \frac{\Delta y}{y}$. Additionally, there is also:

$$\Delta x = \frac{\beta}{1 - \beta}x$$
$$\Delta y = \frac{\alpha}{1 + \alpha}y$$

These are rearrangements of the equations presented in (2) and (3) Demonstration that $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ represent the same relationship..

An example of a simple swap is available in Appendix C: 7.1 Simple swap.

2.1.2 Fees

The next part is the inclusion of a fee for each swap. The equations are altered so that the token amounts are adjusted according to the fee, with the amount generated by the fee given to the liquidity provider, a point we will explore later. The fee increases the size of the token reserves with each swap. The fee is represented by ρ:

$$0 \leq \rho < 1$$
$$\rho = 0.003 \text{ for } 0.3\% \text{ fee}$$

The equations are presented below:

$$x'_\rho = x + \Delta x = (1+\alpha)x = \frac{1+\beta\left(\frac{1}{\gamma}-1\right)}{1-\beta}\,x$$
$$y'_\rho = y - \Delta y = \frac{1}{1+\alpha\gamma}y = (1-\beta)y$$

where $\alpha = \frac{\Delta x}{x}$, $\beta = \frac{\Delta y}{y}$ and $\gamma = 1-\rho$. Additionally, there is also:

$$\Delta x = \frac{\beta}{1-\beta}.\frac{1}{\gamma}.x$$
$$\Delta y = \frac{\alpha.\gamma}{1+\alpha.\gamma}.y$$

As fees are generated, we can represent the change to curve of the liquidity pool visually:

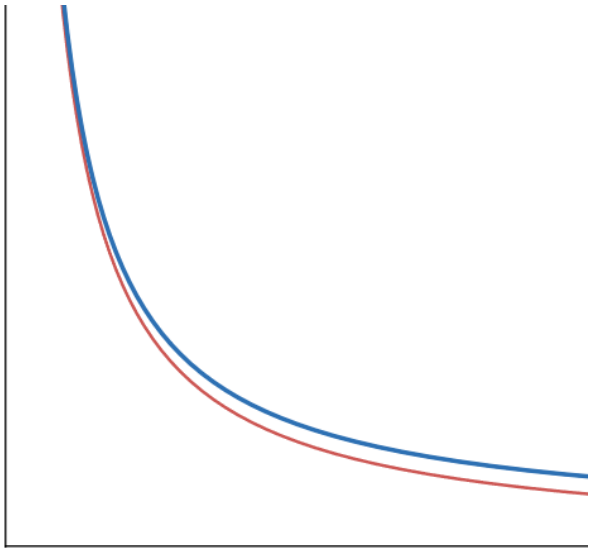

Figure 3 Addition of fees, via Desmos graphing tool

The addition of the fee shifts the hyperbola upwards since the fee is now incorporated into the y reserves. Similarly, if the x token were being bought then the hyperbola would shift to the right.

An example of a swap with fees is available in Appendix C: 7.2 Swap with fees.

2.1.3 Liquidity

Adding and removing liquidity is the next part of the functionality introduced. The formal model uses a tuple to represent the state of the exchange before and after liquidity events.

$$(e, t, l)$$

where e is the amount of Ether, t is the number of exchange tokens, and l is the amount of total liquidity.

Before we examine the liquidity functions it is important to point out that Uniswap v1 was designed based with each liquidity pool using Ether as one of the two tokens. Ether is the numeraire, or if we used the terminology employed by forex markets, then Ether would be the base asset, from a base and quote pair. The other token is a ERC20, which is a type of token standard used by a number of cryptocurrencies. Examples of ERC20 tokens include USD Coin or Shiba Inu, a cryptocurrency named after a popular breed of dog from Japan.

As well as providing for Ether to ERC20 swaps, Uniswap v1 also allows ERC20 to ERC20 swaps, but this uses Ether as an intermediary. Since this requires two swaps to take place, the fee is doubled. We will illustrate the liquidity functionality using a straightforward Ether to ERC20 swap rather than complicating the job with ERC20-to-ERC20 swaps.

Also, a quick word on l which denotes the total liquidity. According to the formal specification, this is a measure of the number of UNI tokens, a token native to Uniswap. This is intended to track liquidity providers so that their rewards can be appropriately allocated depending on the amount of liquidity they have contributed to the pool. Although in later versions of Uniswap the UNI token takes on another role.

Adding liquidity to a pool is outlined as follows:

An input of $\Delta e > 0$

$$(e, t, l) \xrightarrow{addLiquidity\ (\Delta e)} (e', t', l')$$

where:

$$e' = (1 + \alpha)e$$
$$t' = (1 + \alpha)t$$
$$l' = (1 + \alpha)l$$

and $\alpha = \frac{\Delta e}{e}$

You will notice that the equations are like the ones outlined in the section on swaps. Since the reserves of tokens in a liquidity pool is governed by the invariant, adding liquidity first via depositing Ether, then entails the addition of $\Delta t = t' - t$ tokens. Likewise, the number of UNI tokens received for tracking the liquidity providers is $\Delta l = l' - l$, and the initial supply of UNI tokens is set to the initial deposit of ETH. The invariant is preserved using a continued portion ratio $e : t : l$ and $k = e \times t$ increases since we are adding to reserves for each token, so will get a higher value for our constant k.

Three properties are observed:

let $k = e \times t$ and $k' = e' \times t'$

1. $e : t : l = e' : t' : l'$
2. $k < k'$
3. $\frac{k\prime}{k} = (\frac{l'}{l})^2$

Firstly, the new values of e', t' and l' are maintained in the same proportions when liquidity is added. Secondly, the original invariant k is always smaller than the new invariant k'. Finally, the new invariant divided by the old invariant is equal to the square of the new liquidity token amount divided by the old liquidity token amount.

An example of adding liquidity to a pool is available in Appendix C: 7.3 Adding liquidity to a pool.

Removing liquidity from the pool works in the inverse manner. The liquidity token is destroyed or "burnt", and the tokens provided to the pool are returned. The proportion of the tokens returned to the liquidity provider are defined by the current exchange rate and not the rate used when the tokens were added to the pool. This is a key point detailed in the Uniswap Whitepaper (Adams, Uniswap Whitepaper, 2018) and it follows on from how the invariant interacts with the token reserves. Any fees that have accrued while the liquidity provider has been active will be compensated via additional tokens accumulated as e or t.

Removing liquidity is detailed as follows:

An input of $0 < \Delta l < l$

$$(e, t, l) \xrightarrow{removeLiquidity\ (\Delta l)} (e', t', l')$$

where:

$$e' = (1 - \alpha)e$$
$$t' = (1 - \alpha)t$$
$$l' = (1 - \alpha)l$$

and $\alpha = \frac{\Delta l}{l}$

The Δl liquidity tokens are destroyed, $\Delta e = e - e'$ and $\Delta t = t - t'$tokens are withdrawn. Once again, three properties are observed:

let $k = e \times t$ and $k' = e' \times t'$

1. $e : t : l = e' : t' : l'$
2. $k' < k$
3. $\frac{k\prime}{k} = (\frac{l'}{l})^2$

Similar to adding liquidity, we have a continued portion ratio, the invariant this time is smaller after removing liquidity than the state before, and we have the same identity relationship between the invariant and liquidity tokens.

The final thing to note on adding and removing liquidity is to show how this effects our curve.

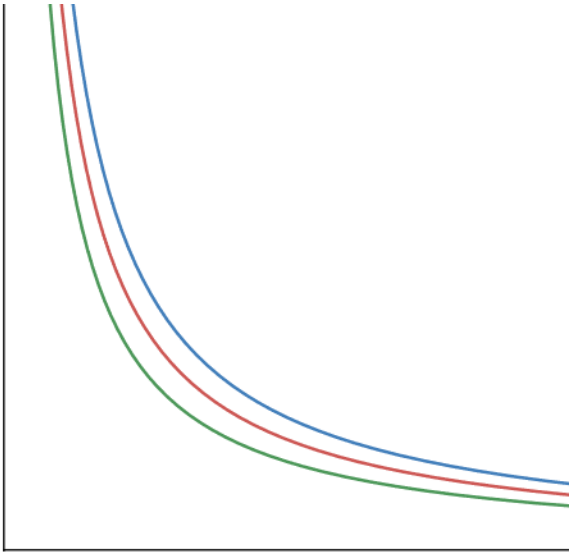

Figure 4 Illustrating addition and removal of liquidity, via Desmos graphing tool.

With the red curve representing the original amount of liquidity, the blue curve demonstrates liquidity being added to the pool, and the green curve shows liquidity being removed.

An example of liquidity being removed from a pool is available in Appendix C: 7.3 Adding liquidity to a pool.

2.2 Uniswap v2

This version of Uniswap really marks the moment when adoption of the decentralised exchange kicked off. There were several changes to the design and in this section, we follow the technical specification laid out by the whitepaper (Adams, Zinsmeister, & Robinson, Uniswap v2 Core, 2020).

2.2.1 ERC20 pairs

The beta version of Uniswap used Ether as the medium of exchange for all assets on the platform. Swaps could be carried out from Ether to ERC20 tokens, the inverse, or from ERC20 to ERC20, but only using Ether as the intermediate token. This had significant implications in terms of the fees, since effectively you would be carrying two trades.

For example, if you had Shiba Inu tokens ($SHIB$), the Japanese dog cryptocurrency, and wanted to exchange them for Wrapped Bitcoin ($WBTC$), the platform provided a smart contract function that would carry out the following:

$$SHIB \xrightarrow{Swap} ETH \xrightarrow{Swap} WBTC$$

All liquidity pools were denoted in Ether versus another token. In Uniswap v2, Ether is no longer the mandatory base asset and pools can be constructed from any combination of ERC20 tokens. This means if a liquidity provider created a $SHIB \leftrightarrow WBTC$ liquidity pool, you would no longer to pay the two-step fees required to carry out the swap. And the liquidity

provider would no longer be exposed to Ether as the medium of exchange, they would solely be interested in the price of the two arbitrary tokens in the pool.

2.2.2 Price oracle

Uniswap v2 creates a new price oracle intended to provide price information for third-party services using data from the exchange. Price oracles have become an important part of the cryptocurrency ecosystem and in particular within Decentralised Finance (DeFi). The general idea is to provide correct prices so that different blockchain services can interact.

We discuss the importance of this later, but for now it is important to understand that a price oracle for Uniswap intends to make reliable price information from the exchange available externally.

We touched upon the execution price in section 2.1.1, outlining the execution price of a particular swap. We also have the notion of a market price, representing the theoretical equilibrium price given the state of token reserves. The whitepaper outlines a formula for this, which we rewrite to maintain the notation we have followed up to this point:

$$Price_t = \frac{y_t}{x_t}$$

where x and y are the reserves of a given token within a liquidity pool, and t is a given time. This does not reflect the actual price received in each swap.

Uniswap v2 constructs the price oracle using a time-weighted average based on the market price stored in an accumulator on the blockchain. We must mention here that a new block is mined on the Ethereum blockchain approximately every 12 seconds. This "block time" has been relatively stable since September 2022, according to analytics (Ethereum Average Block Time Chart, 2023). Also, time on the blockchain is stored in the Unix Epoch format, denoting the number of seconds that have elapsed since 1 January 1970.

A little recap on the functioning of the blockchain: each block contains a batch of transactions, and the blocks are linked together in a sequential chain using a cryptographic key.

The price oracle mechanism works by calculating a value at the end of a given block based upon the market price and the time elapsed since the last block. This value is then accumulated with the idea being that it is in effect the sum of the market price at every second. The formula is as follows for a given block:

$$a_t = a_{t-1} + (p_t \times (t_t - t_{t-1}))$$

where a is the accumulator, t is the timestamp for the block and p is the market price.

Then to estimate the time-weighted average price from t_1 to t_2 the difference of the accumulators is divided by the time elapsed.

$$p_{t1,t2} = \frac{a_{t2} - a_{t1}}{t_2 - t_1}$$

We can illustrate this with an example using data directly from the blockchain using the Uniswap v2 smart contract (Contract: 0xB4e16d0168e52d35CaCD2c6185b44281Ec28C9Dc,

2023). This example shows us the reserves for ETH and $USDC$ tokens in a series of blocks from 09:50:11 to 09:50:59 on 20 April 2023.

Block	Timestamp	Reserves ETH	Reserves USDC	Price (USDC/ETH)	Time Elapsed	Accumulator
17086348	1681977011	18,475.68	36,119,637.53	1,954.98	0	0
17086349	1681977023	18,476.10	36,118,823.91	1,954.89	12	23,458.73
17086350	1681977035	18,475.90	36,119,222.88	1,954.94	12	46,917.98
17086351	1681977047	18,475.90	36,119,222.88	1,954.94	12	70,377.23
17086352	1681977059	18,476.81	36,117,447.08	1,954.74	12	93,834.17

If we want to calculate the time-weighted average price between block 17086348 and 17086352 we apply the formula using the accumulator values and the timestamps:

$$TWAP = \frac{0 - 93{,}834.17}{1681977059 - 1681977011} = 1954.88$$

In practice, the accumulator contains the full price history, but the illustration serves to outline how it functions. It can be queried according to the interval for the TWAP required.

2.2.3 Flash Swaps

This feature allows a user to receive a cryptocurrency token from a liquidity pool and use it before they have actually paid for it or without any collateral. The most important part of this is what is called an “atomic transaction”, which is an operation or series of operations that is carried out to completion, or otherwise fails. The operation must finally result in either the withdrawn token being paid for with the corresponding amount of the other token in the pair or returning the same token, minus the fees. This is enforced programmatically in the smart contract.

Flash swaps enable what the documentation (Core Concepts: Flash Swaps, 2023) calls “capital-free arbitrage”, stating that the intention is to make arbitrage widespread enough to help ensure the exchange maintains a fair market price. It also relies upon the interoperability of Ethereum smart contracts running other DeFi platforms.

We can provide a brief example with a flash swap involving $USDC$ and DAI. We already mentioned $USDC$, it is a stablecoin token pegged to the US dollar with an organisation called Circle backing it with actual US dollars held in reserve outside of the cryptocurrency system. DAI is also a stablecoin token, but this time it is backed by a mix of other cryptocurrency tokens and managed through smart contracts. The taxonomy of the tokens is not important here, we are taking one token from the Uniswap liquidity pool in a flash swap, then carrying out another operation, before returning the other token back to the pool plus the fee. The initial operation in Diagram 2 takes 2000 $USDC$ from a Uniswap liquidity pool in a flash swap and interacts with another DeFi service, then returns the equivalent amount in DAI, and pays the necessary fee. The arbitrageur nets a profit of 23.33 DAI and has not risked any capital. If at any stage one of the operation fails, the whole transaction reverts.

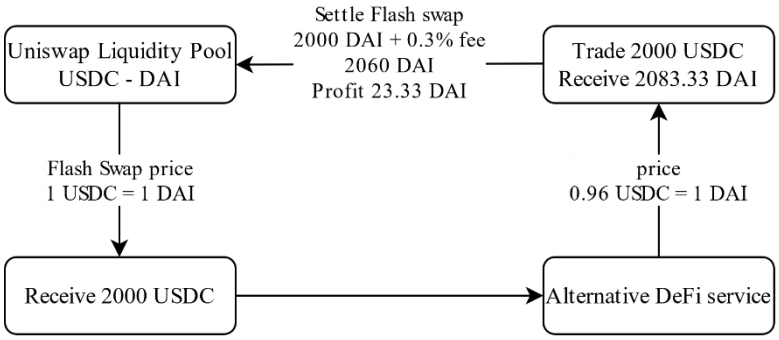

Figure 5 Basic arbitrage using Uniswap flash swap.

The Alternative DeFi service may be a competing DEX, or another service providing margin trading, or a credit and lending platform. The essential point to understand is that the trader has identified some kind of price discrepancy. To simplify this example, we have also assumed that there is no price slippage within the Uniswap liquidity pool, the exchange rate remains the same when the flash swap is repaid. This is not the case as we have seen earlier in our examination of liquidity pools because each trade impacts the subsequent execution price but serves to illustrate the principal.

2.2.4 Protocol fees

We have already discussed fees paid to liquidity providers. Uniswap v2 introduces a protocol fee intended to compensate the platform itself. This is set as 0.05%, or $\frac{1}{6}$ of the 0.3% fee levelled on every swap, effectively reducing the fee earnt by liquidity providers to 0.25%. This fee is hardcoded into the smart contract itself but is not currently switched on. Since it currently does not have a material impact on the economic incentives, we will not go further in our exploration of protocol fees.

2.2.5 Liquidity token supply

Previously in section 2.1.3 we described how the liquidity token UNI was issued to liquidity providers when they added a token pair to a pool and how it was subsequently destroyed when the corresponding liquidity was removed. This token defines the providers' share of the pool. In Uniswap v1, the initial supply was set equal to the initial deposit in ETH and further manipulation of the UNI supply was regulated as such:

$$l' = (1 + \alpha)l$$

$$\alpha = \frac{\Delta e}{e}$$

where l is the amount of liquidity and e is the amount of Ether.

Uniswap v2 changes the way liquidity tokens are managed. And although the nomenclature changed, since UNI tokens became related to governance of the platform, we will continue to refer to UNI tokens as liquidity tokens for the sake of clarity.

The addition of new liquidity to a pool creates UNI tokens as follows:

$$l = \sqrt{\Delta x . \Delta y}$$

where x and y are tokens in a pool l. This is the geometric mean of the tokens deposited.

An example of liquidity tokens is available in Appendix C: 7.4 Uniswap v2 liquidity tokens.

2.3 Uniswap v3

Uniswap v3 launched in May 2021 and was described how it had become a part of "critical infrastructure for decentralized finance", lauding some $135bn in trading volume through the v2 platform (Introducing Uniswap v3, 2021).

2.3.1 Concentrated liquidity

The major change in the latest version of Uniswap is the introduction of concentrated liquidity. This allows liquidity providers to contribute to a liquidity pool within a range of prices called positions. Holding a position between two prices of a token pair means that if the price remains within this range, the liquidity provider will earn fees from their position, outside of this range their liquidity becomes inactive, they no longer earn fees.

The jumping off point is the original formula $x.y = k$, but the curve is translated, representing a shift from "virtual reserves" of the token pair to "real reserves". We will first describe the explanation from the whitepaper (Adams, Zinsmeister, Salem, Keefer, & Robinson, 2021).

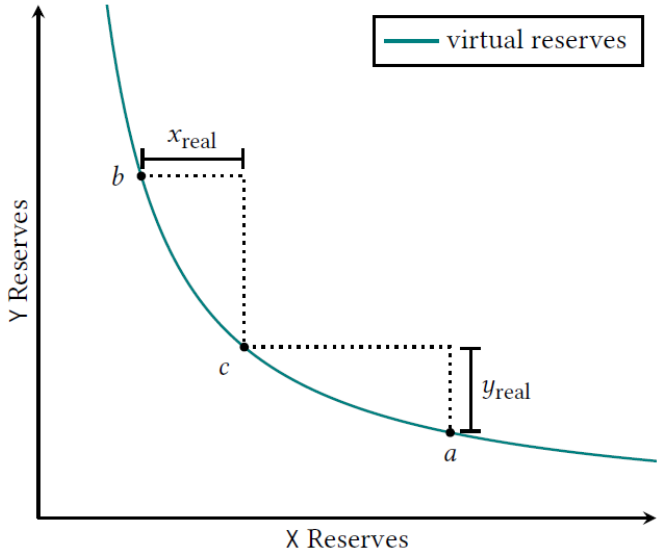

Figure 6 Simulation of virtual liquidity, Uniswap v3 Core whitepaper

In reference to the original model, the virtual reserves are made up of combinations of a token pair from all liquidity providers. If we wish to create a bound range, we no longer wish to hold reserves to account for prices from 0 to ∞ we instead have the $real$ reserves covering the range a to b, with the market price c somewhere in the middle as illustrated in Figure 6. Furthermore, if we reach the limits of a price bound the token balance within that liquidity position would be completely converted to one or the other. Whereas the original Uniswap model would continue to provide swaps along the asymptote with infinitesimally small amounts, Uniswap v3 constrains the curve. We can see this on a graph representing virtual and real reserves:

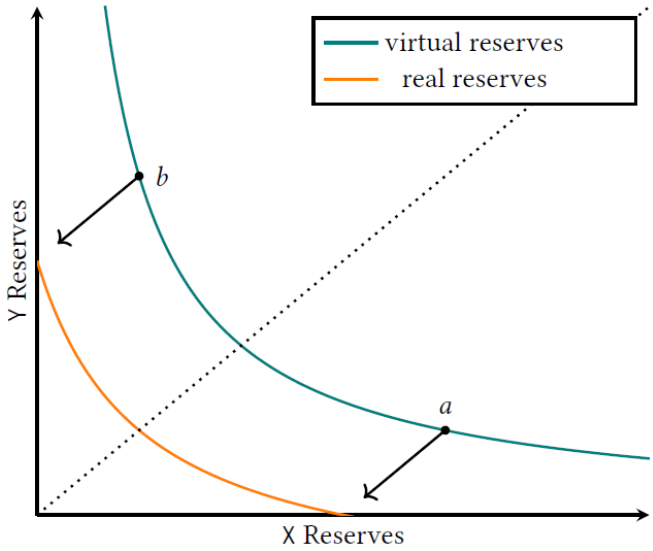

Figure 7 Real reserves, Uniswap v3 Core whitepaper

Unlike in the original model, liquidity providers are determining their own unique position and price bounds and we can think of the real reserves as representing one liquidity provider.

Formally, we have:

$$x.y = k$$
$$L = \sqrt{k}$$
$$L = \sqrt{x.y} \quad (4)$$

where L is liquidity in the position, or real reserves. The original $x.y = k$ relationship tracks the total liquidity or virtual liquidity, and $L = \sqrt{x.y}$, the liquidity position, but it is not yet constrained, for the moment it is simply another representation of the original model. To describe the translation of the curve from virtual to real reserves, as outlined in Figure 7, we can think about the x and y offset.

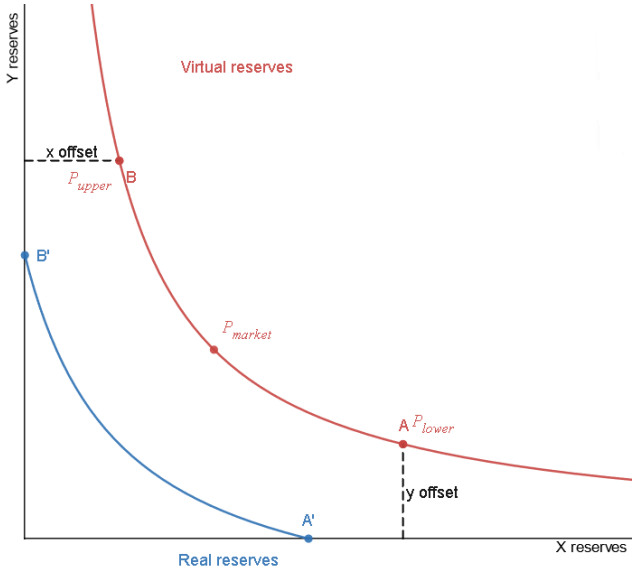

Figure 8 Transforming v2 curve to v3 curve, via Desmos graphing tool.

Here, we will follow the technical note detailing the liquidity formula (Elsts, 2022). Plugging in the x and y offsets into (4) would translate our virtual reserves into real reserves, moving point A to A' and B to B', giving us the following equation:

$$\left(x + x_{offset}\right).\left(y + y_{offset}\right) = L^2$$

The $offset$ allows us to constrain the liquidity position. Now, considering that the price of y in terms of x is given by the slope of the curve $P = \frac{y}{x}$ and that $L = \sqrt{x.y}$, we can substitute into the equation solving for x in terms of price and liquidity:

$$L^2 = x.y, \quad y = P.x$$
$$L^2 = P.x^2$$
$$L = x\sqrt{P}$$
$$x = \frac{L}{\sqrt{P}}$$

Similarly solving for y:

$$L^2 = x.y, \quad x = \frac{y}{P}$$
$$L^2 = \frac{y^2}{P}$$
$$L = \frac{y}{\sqrt{P}}$$
$$y = L\sqrt{P}$$

These two values for x and y represent the offset between the real and virtual reserves, and can replace our offset to provide us with a formula for the real reserves for a liquidity position:

$$\left(x + \frac{L}{\sqrt{P_{upper}}}\right)\left(y + L\sqrt{P_{lower}}\right) = L^2 \quad (5)$$

where P_{lower} and P_{upper} represents the minimum and maximum price of a position. This is consistent with the formula presented in the whitepaper.

However, equation (5) is no good if we want to determine how many y tokens we need for a given number of x tokens at a certain price range. Much like earlier in Uniswap v1 we need to solve this equation for the different terms. To do so, we need to take three scenarios:

1. $P_{market} \leq P_{lower}$
2. $P_{market} \geq P_{upper}$
3. $P_{lower} < P_{market} < P_{upper}$

So, using the fact that at the x and y intercepts the other token quantity will be 0, we can rearrange to solve for x, y and L (below) to deal with the first two scenarios:

$$x = L.\frac{\sqrt{P_{upper}} - \sqrt{P_{lower}}}{\sqrt{P_{lower}}.\sqrt{P_{upper}}} \quad (6)$$

$$L = x.\frac{\sqrt{P_{lower}}.\sqrt{P_{upper}}}{\sqrt{P_{upper}} - \sqrt{P_{lower}}} \quad (7)$$

$$y = L.\sqrt{P_{upper}} - \sqrt{P_{lower}} \quad (8)$$

$$L = \frac{y}{\sqrt{P_{upper}} - \sqrt{P_{lower}}} \quad (9)$$

The third scenario requires a bit of gymnastics. If the market price is in range of the liquidity position, we can imagine that either side of P_{market} the token reserves are both contributing equally to the liquidity. Depending on the price, the reserves might be any configuration of different ratios, but they are both going to be used to provide for swaps at whatever prevalent price.

To say this another way, if we have an amount of a token pair and either an upper or lower price bound in mind, then the market price will constrain what amount of the corresponding token we can put into the liquidity position at a given price.

If we therefore introduce, P_{market} we can assert that $L_x = L_y$ with (P_{market}, P_{upper}) providing a part of the liquidity for x and (P_{lower}, P_{market}) providing a part of it for y. This

allows us to manipulate the equations (7) and (9), replacing what would be either the lower or upper part of this imaginary division with P_{market}.

Equation (7) becomes:

$$L_x = x.\frac{\sqrt{P_{market}}.\sqrt{P_{upper}}}{\sqrt{P_{upper}} - \sqrt{P_{market}}}$$

and equation (9) is:

$$L_y = \frac{y}{\sqrt{P_{market}} - \sqrt{P_{lower}}}$$

giving us:

$$x.\frac{\sqrt{P_{market}}.\sqrt{P_{upper}}}{\sqrt{P_{upper}} - \sqrt{P_{market}}} = \frac{y}{\sqrt{P_{market}} - \sqrt{P_{lower}}} \quad (10)$$

This equation enables us to determine what amount of x or y tokens we need to deposit with a given price range at a certain market price without any recourse to liquidity. We can rearrange (10) to solve for x and y:

$$y = \frac{(x.\sqrt{P_{market}}.\sqrt{P_{upper}})(\sqrt{P_{market}} - \sqrt{P_{lower}})}{\sqrt{P_{upper}} - \sqrt{P_{market}}}$$

$$x = \frac{y(\sqrt{P_{upper}} - \sqrt{P_{market}})}{\sqrt{P_{market}}.\sqrt{P_{upper}})(\sqrt{P_{market}} - \sqrt{P_{lower}}}$$

Further manipulation of P_{lower} and P_{upper} can also enable us to get a range from an amount of tokens and we can also rearrange to find the amount of tokens after a price change.

To conclude, a word about how the liquidity positions come together to form a liquidity pool. In previous versions, this was simply one curve with tokens deposited to represent the 0 to ∞ range of prices. With Uniswap v3, as we have seen, the individual liquidity positions with price bounds are paramount. To visualise this, we can illustrate how several segments of the different shaped hyperbolas represent each liquidity position:

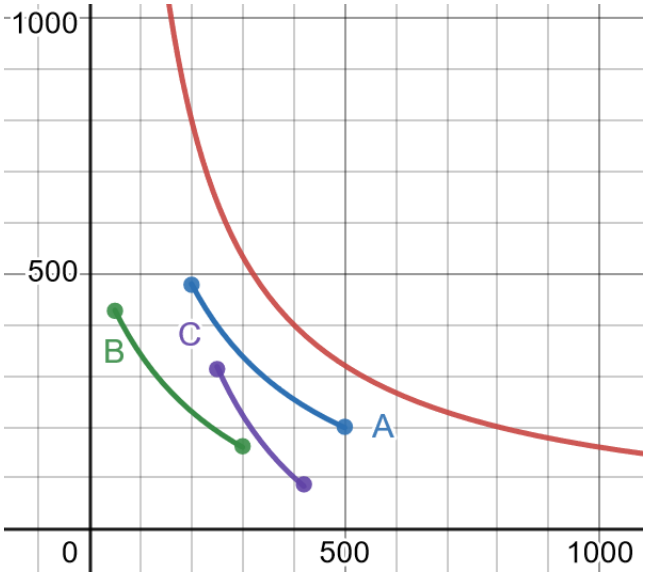

Figure 9 Liquidity positions, Desmos Graphing Calculator

In Figure 9, three different liquidity providers hold positions A, B and C, with the red curve representing the virtual reserves. The three positions are all set at different prices, but they also overlap one another at different points, for instance with $\{250 \leq x \leq 300\}$ both A, B and C positions will be contributing liquidity.

2.3.2 Ticks

To implement concentrated liquidity, Uniswap v3 uses a system of ticks. The range of possible prices for a token pair is split up into discrete ticks such that every price p is an integer power $p(i) = 1.0001^i$. This means that the difference between two ticks is 0.01%.

A liquidity position sets the aforementioned lower and upper price in a range of ticks. This range of ticks for a liquidity position can either be in or out of range depending on the current price. According to price movements, the appropriate ticks for all liquidity positions will be initialised for use and there is an accounting mechanism to ensure that each liquidity position gets paid the proportional share of fees.

For swaps that are completed within a specified tick, the operation is treated as in Uniswap v1 outlined earlier. However, given that Uniswap v3 is using L and $\sqrt{P}$ the formulas are adjusted accordingly. Once liquidity in a specified tick is used up, the next tick is activated, and the liquidity within this tick used to complete the swap. Given the nature of the implementation, if the swap uses up the liquidity in a specified swap and crosses into the adjacent tick, then it will increase the cost of the transaction because a further call will be made within the system of smart contracts.

A further constraint on the system of ticks is tick spacing which is defined by the fee tier of the liquidity pool. The tick spacing limits which ticks can be initialised. Ticks are mainly an implementation detail so we do not further examine this mechanism.

2.3.3 Fees and tier structure

Fees from swaps were automatically included into the liquidity pool as they accrued in previous versions of Uniswap. With this latest version the fees are not automatically included

into the pool. The implication of this is that the token reserves do not grow as fees are generated.

The other change with fees is the creation of a fee tier structure. Liquidity pools can be created with four fee levels: 0.01%, 0.05%, 0.30% and 1%. Likewise, the same token pair can be included in different pools with different fee tiers. The idea is that certain token pairs will gravitate towards certain fee tiers according to the volatility. In their documentation, Uniswap said they expect less volatile token assets, such as stablecoins, to be pooled with the lowest fee tier pool, while more exotic token pairs, will gather in higher fee level pools.

3. Incentives

In this part we will explore the incentive system provided by Uniswap and its implications. For individuals wanting to swap tokens, two main considerations are apparent: the fees and direct interaction for a trade without a custodian. The fees are composed of the fees paid to liquidity providers and the gas fees paid to the blockchain miners to process the transaction. These fees are presented differently compared to CEXs since a swap on the exchange is not directly interacting with the blockchain. While carrying out the trade without a custodian is a unique characteristic of DEXs. It can be viewed as a positive property for DeFi purists, or a risk for those who see the need for a central authority as necessary to protect consumers. Despite the importance of these two features, we will instead focus on the incentive system for liquidity providers.

Earning fees from depositing tokens into a liquidity pool is similar to earning a spread for a market maker in a traditional limit order book. It is the main incentive for a liquidity provider. However, it is not as simple as depositing two sets of tokens and earning fees because the underlying price of the tokens can change. Before discussing fees and how these factor into a decision to provide liquidity, we must understand how changes in price impact a liquidity provider.

3.1 Impermanent Loss (IL)

This is the potential loss in a liquidity position compared to simply holding the original assets. It originates from the functioning of the constant product function. Each swap occurring in a liquidity pool has a price impact and as subsequent swaps are settled this also changes the composition of the liquidity pool. If the liquidity pool experiences a prolonged period of increased demand for one of the tokens (or reduced demand for the other token), the execution price will adjust upwards (or downwards). Simultaneously, the proportion of the tokens in the liquidity pool will be modified. There will be less of the demanded token in the pool, and more of the unwanted. Likewise, the price of the demanded token will increase, and price of unwanted token will decrease. These two mechanisms are the source of impermanent loss.

Examples are available in Appendix C: 7.5 Impermanent loss.

It is said to be impermanent because this could simply be a temporary price change and if the price snaps back to the initial amount, then no actual loss will occur. But if the price change is not a temporary phenomenon and the liquidity provider removes their tokens, then a permanent loss will be realised. Contrary to the fees earnt by providing liquidity, impermanent loss represents a risk.

Several authors have analysed the impact of impermanent loss. We follow a scenario focused on Uniswap v2 with a risk neutral liquidity provider operating in a perfectly competitive liquidity pool, not accounting for fees earnt (Barbon & Ranaldo, 2022). We follow the authors' notation to avoid ambiguity.

A liquidity pool of X and Y tokens, containing quantities x_0 and y_0 at time $t = 0$, with a liquidity provider owning ψ of the pool, and a quoted price of $P_0 = \frac{y_0}{x_0}$. A liquidity provider's initial position is described by:

$$\psi(x_0 P_0 + y_0) = 2\psi y_0$$

We arrive at $2\psi y_0$ by substituting $\frac{y_0}{x_0}$ into P_0 and expanding the equation.

Later, the value of the liquidity position changes such that:

$$\psi(x_1 P_1 + y_1) = 2\psi y_1$$

To calculate the percentage change between $t = 0$ and $t = 1$ we can express the return for a liquidity provider as:

$$R_{LP} = \frac{2\psi y_1}{2\psi y_0} = \frac{y_1}{y_0}$$

(11)

We can rewrite the constant product rule $x.y = k$ and substitute into the equation for price, giving us:

$$P_0 = \frac{y_0^2}{k} \Rightarrow\ y_0 = \sqrt{kP_0}$$

In the same way, at $t = 1$, $y_1 = \sqrt{kP_1}$ and if we return to equation (11) we can substitute in:

$$R_{LP} = \frac{y_1}{y_0} = \frac{\sqrt{kP_1}}{\sqrt{kP_0}} = \sqrt{\Delta P}$$

(12)

Since $\Delta P = \frac{P_1}{P_0}$ we are left with an equation showing that the change in value rests upon the square root of the price change.

Our comparison is with the return for simply holding the tokens. Because we are representing our values accounted for in units of Y the change in value equals 1 and ΔP for X, divided by 2 to provide the average return for the two tokens.

$$R_H = \frac{1}{2}(\Delta P + 1)$$

(13)

The impermanent loss is therefore defined as the change in value of the liquidity pool, equation (12), minus what they would have been worth if the tokens were simply held, equation (13), giving us the net opportunity cost.

$$IL = R_H - R_{LP} = \frac{1}{2}(\Delta P + 1) - \sqrt{\Delta P} \quad (14)$$

The first order derivative with respect to ΔP is:

$$\frac{dIL}{d\Delta P} = \frac{1}{2} - \frac{1}{2\sqrt{\Delta P}}$$

which if we solve for 0 gives us $\Delta P = 1$ indicating that liquidity providers will always be worse off if $\Delta P \neq 1$ compared to simply holding the tokens. If the price change is temporary and reverts to the initial amount, then $IL = 0$, but if it is permanent then IL increases in magnitude with the price change given the convexity of the constant product function.

If we go back to the example in Appendix C: 7.5 Impermanent loss, the fees generated by providing liquidity must be greater than the impermanent loss, otherwise the liquidity provider will not make a net profit. If the change in price is small, the impermanent loss could be more than compensated by the liquidity fees generated. If there is no price change, there is no impermanent loss, and the liquidity provider simply profits from the generated fees.

Equation (14) formalises this for us giving us this relationship, spelling out that any price change is going to be detrimental to the liquidity provider's profits. It is in the liquidity provider's interest that the price does not move. Furthermore, if the fees do not compensate for the impermanent loss, then liquidity providers will always be penalised compared to simply holding the tokens. Since every swap involves price impact, impermanent loss is a key characteristic.

3.2 Equilibrium liquidity

The authors describe this impermanent loss as a measure of "adverse selection" faced by liquidity providers, saying it is similar to market makers in a traditional limit order book market. They go on to show how impermanent loss is a major factor in the levels of liquidity provided to a pool demonstrating it empirically. In order to do so, they tie it into a larger model of equilibrium liquidity and fees, describing the problem of providing an optimal quantity of liquidity to the DEX. The conditions remain the same as previously outlined with some additions.

At $t = 0$ the total liquidity is equal to x and the liquidity provider can add or remove ξ liquidity to the pool. Trading occurs with $t > 0$ and stops at $t = 1$. A random variable V represents the total traded volume in the pool denoted in units of x and $\Delta P = \frac{P_1}{P_0}$ is the change in quoted price within the interval.

$E[V]$ represents the expected trading volume and $E[IL]$ represents the expected IL at time $t = 0$. We now incorporate the fees earnt by the liquidity provider denoted by f, the fee rate. Given that these fees are earnt according to the share of the liquidity pool owned by the provider we can represent this as follows:

$$\frac{\xi}{x + \xi} f E[V]$$

In other words, the share of the liquidity pool, multiplied by the fee percentage, multiplied by the trading volume. This is simply a reconstruction of the mechanism outlined earlier (Fees) but expressed as a function of trading volume rather than a single swap and in units of X. The fees can be expressed as a percentage return if we divide by ξ:

$$\frac{fE[V]}{x+\xi}$$

Then we can define the total expected percentage return $E[R]$ considering both the fees and the IL previously defined.

$$E[R] = \frac{f}{x+\xi}E[V] - E[IL] \quad (15)$$

Because the authors are assuming a liquidity pool in perfect competition, the liquidity provider is expected to make zero profit. So, they outline an equilibrium level of total liquidity $x^* = \xi + x$ and plugging this into equation (15) and solving for 0 gives us:

$$x^* = \frac{fE[V]}{E[IL]} \quad (16)$$

This relationship gives us the equilibrium level of total liquidity, which is a function of trading volume and fees as the numerator, and impermanent loss as the denominator. Total liquidity increases with expected trading volume and decreases with expected impermanent loss. To put this another way, with more trading volume expected within the liquidity pool, liquidity providers will deposit more tokens to take advantage of the higher levels of fees generated. On the flipside, more volatile prices increase the likelihood of impermanent loss, therefore countering the likelihood of liquidity providers participating in the pool.

To test this empirically, the authors use data liquidity data from Uniswap v2 for 100 token pairs between April 2020 and April 2021, with 42,293 observations. They calculate a rolling average of daily traded volume, which acts as a proxy for $E[V]$ and a rolling average of $E[IL]$ estimated over the prior two weeks. In their regression, observed liquidity is the dependent variable and predicted liquidity, based on equation (16), is the independent variable. The results show high levels of significance and an $R^2 > 0.92$ across four panels, demonstrating that the predicted values explain almost all the variance in the model, with one of the panels finding $R^2 = 0.98$. They conclude that the equilibrium model fits empirically and captures the risk-reward relationship liquidity providers face in earning fees but risking impermanent loss.

“Our simple equilibrium model is empirically relevant, as it is able to capture the main economic trade-off faced by LPs in AMM-based DEXs,” says Barbon & Ranaldo, referring to liquidity providers. To interpret this a different way, we can say that liquidity providers are analysing token prices, estimating the potential impermanent loss, and making their decisions on liquidity provision by weighing the possible fees and expected volumes of trading. The results remain relevant with different rolling time windows since the study considered 5-day and 20-day periods reflecting the median and average length of time for a liquidity position.

3.3 Uniswap v3 and impermanent loss

The previous analysis focuses on Uniswap v2 and fits impermanent loss into a larger framework of equilibrium liquidity. Other authors have studied impermanent loss in Uniswap v3, arriving at similar results when the curve is bounded by an upper and lower price range (Heimbach, Schertenleib, & Wattenhofer, 2022). Furthermore, the structure of Uniswap v3 actually leads to larger impermanent loss for the same price change than in Uniswap v2. And the size of the price range also impacts the magnitude of the impermanent loss.

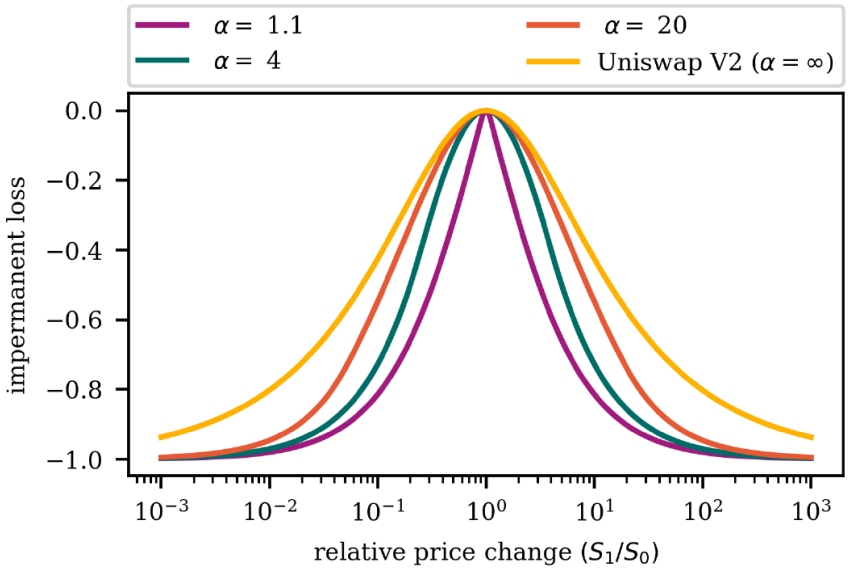

Figure 10: Simulation of impermanent loss of liquidity provider, Heimbach, Schertenleib & Wattenhofer, 2022

Figure 10 shows the impermanent loss at a given price change, with α representing the size of the price bounds. Uniswap v2 in yellow, bounded from 0 to ∞, demonstrates the least severe outcome from a price change. In Uniswap v3, the tightest price bounds in purple leads to the steepest impermanent loss, and orange, the widest price limits, offers something closer to Uniswap v2.

The graph spells this out clearly. We can also think about it intuitively if we consider the example in Appendix C: 7.5 Impermanent loss. With an increasing demand for one token on the Uniswap v2 platform, the price increases, and the amount of the demanded token in reserve in the pool is diminished. However, it is never completely exhausted. The constant product function in Uniswap v2 will continue to sell the demanded token and reserves in the pool will drop to infinitesimally smaller amounts.

With Uniswap v3 the bounding of the constant product function results in a different situation. The upper and lower price bounds for a liquidity provider result in the reserves for one of the tokens being completely exhausted. If one of the tokens is in high demand, the price increases, and the reserves of the demanded token are depleted until the price hits the upper bound at which point the liquidity provider is left with reserves only containing the unwanted token. Since each liquidity provider has their own customised price bounds, they define what change in price will entail this conversion. Bounding the constant product curve on the intersection with the x and y axis, using the offset described in 2.3.1, results in a greater magnitude of impermanent loss illustrated by Figure 10.

3.4 Behaviour and strategies of liquidity providers

The trade-off between impermanent loss and fees results in liquidity providers acting somewhat like market makers for limit order books in traditional financial markets. Subsequent authors have modelled effective hedging strategies for automated market makers like Uniswap to limit the potential impermanent loss (Khakhar & Xi, 2022) (Deng, Zong, & Wang, 2023). Besides hedging the impermanent loss, the other option involves changing the liquidity position accommodating price changes. In this section, we will briefly consider some possible strategies that the system of Uniswap liquidity provision permits. We will ignore hedging strategies initiated outside of Uniswap. Since Uniswap operates via a smart contract, we must also point out that this results in gas fees each time a change to the liquidity position is made, unlike changes to a limit order book on a CEX.

3.4.1 Uniswap v2

Uniswap v2 does not feature price bounds and therefore liquidity providers are not in direct competition on judging the volatility of prices. Nevertheless, it still suffers from the phenomenon of impermanent loss. For a given liquidity provider we can visualise the liquidity position as flat across a price range from 0 to ∞:

Figure 11 Liquidity provision across price range in Uniswap v2

Despite not being able to set price bounds, the liquidity provider can of course set their own arbitrary price bounds for which liquidity provision is acceptable for them. Outside of this price range the liquidity provider can withdraw their tokens from the pool. This results in a binary strategy with either liquidity provision on or off. A liquidity provider may well decide to withdraw from the pool if certain price limits are exceeded. Or they may have a longer-term vision on the price evolution of a particular token and be willing to leave their tokens in the pool despite short term fluctuations in price.

Additionally, Uniswap v2 uses a flat rate fee for all pools so liquidity providers do not have to select an optimum fee level. Entering and withdrawing from a liquidity pool will result in gas fees so this must be weighed up against the generation of fees and the potential impermanent loss.

3.4.2 Uniswap v3

Liquidity providers in Uniswap v3 are incentivised to choose a narrow price range in order to earn the largest proportion of fees. Yet by doing so they also expose themselves to the most significant impact of impermanent loss.

3.4.2.1 Simple lower and upper bounds

Uniswap v3 allows liquidity providers to use the full 0 to ∞ price range as in Uniswap v2, or to select a specific price range. The simplest strategy is to select a lower and upper price range.

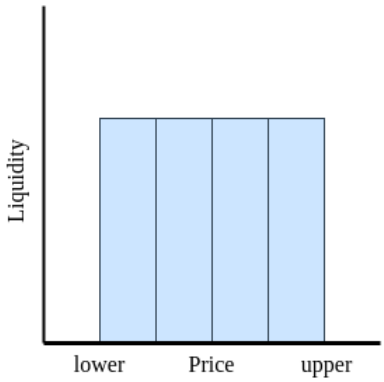

Figure 12 Liquidity provision across simple price range in Uniswap v3

The liquidity is distributed across the ticks described in 2.3.2. The strategy assumes that there is an equal probability of the price moving within the lower and upper bounds. It might be prudent to select a price range by estimating how the price will evolve over a given time period using past data.

3.4.2.2 Anticipating increases or decreases in price

Taking the simple strategy to the next level of sophistication involves shifting the distribution to the left or the right of the current price.

In Figure 13, the liquidity provider has an expectation that the price will fall and therefore weighs their tokens to the lower end. Similarly, in Figure 14, the liquidity provider weighs their tokens to the upper price limit. If the price does indeed move in the direction expected, the liquidity provider will benefit from continuing to earn fees. But if the price does not

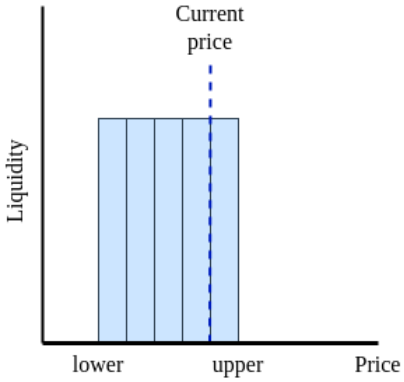

Figure 14 Liquidity provision weighted towards lower prices in Uniswap v3

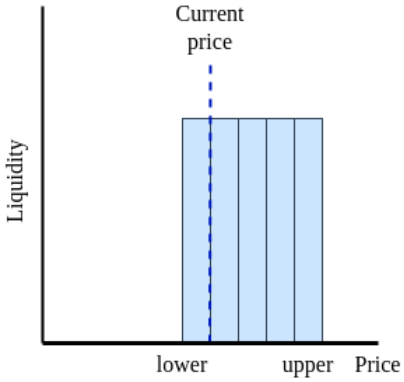

Figure 13 Liquidity provision weighted towards higher prices in Uniswap v3

increase or decrease as expected, and pass the range bounds, the liquidity position will move out of range.

3.4.2.3 Out of range

A similar strategy involves setting the price range outside of the current price. This has the characteristic of only needing to deposit of one of the token pairs.

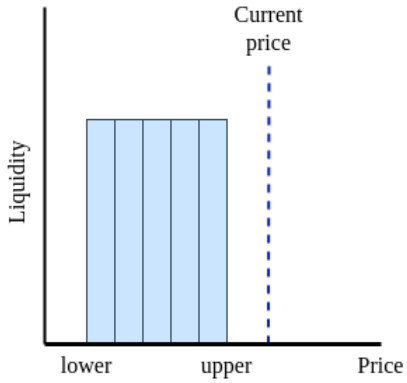

Figure 15 Liquidity provision out of range in Uniswap v3

In Figure 15, the upper and lower price bounds are outside of the current price with the liquidity provider expecting the price to drop. If the price drops, the liquidity position will become active. On creating the position, the liquidity provider will only need to deposit one of the tokens. Then if the position becomes active the amount of deposited token will start to be converted into the other token, giving them a mix of the token pair.

3.4.2.4 Multiple liquidity positions

With multiple liquidity positions a provider can create much more elaborate distributions to model different expectations of the price action.

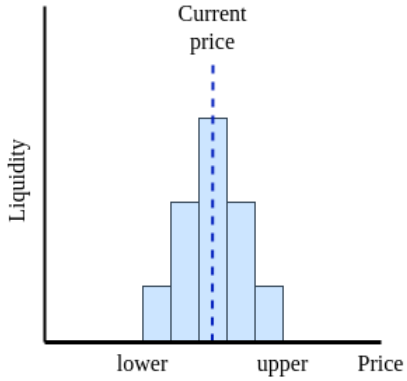

Figure 16 Liquidity provision with five positions concentrated around the current price on Uniswap v3

In Figure 16, the distribution is centred around the current price similar to a bell curve. The liquidity provider could have the expectation the price is most likely to remain around the current price. But is also anticipating more unlikely price movements by concentrating less liquidity in the tails.

Many more possibilities are available with multiple liquidity positions and a provider might, for example, want to use a distribution such as in Figure 16 but skewed positively or negatively, in order to better anticipate price movements. We could also imagine a U-shaped or V-shaped distribution with the majority of liquidity allocated towards the upper and lower bounds and less allotted to the middle.

3.4.2.5 Resetting positions

As well as the shape of the liquidity distribution, another important factor in managing impermanent loss and fee generation is time. A profit-maximising liquidity provider will want to monitor their position and make adjustments to avoid impermanent loss and maximise fees. They therefore might be calculating the expected returns and standard deviation of token prices in order to inform their positions. In this case, it is vital to have a framework in place to manage a liquidity position or positions over particular time horizons. Some authors have developed classes of strategies defining a reset condition that is periodically evaluated in conjunction with expected price ranges based on historical data (Neuder, Rao, Moroz, & Parkes, 2021).

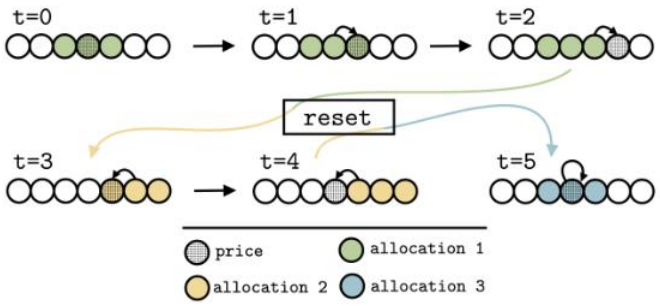

Figure 17 A reset strategy, Neuder, Rao, Moroz & Parkes, 2021

Figure 17 shows the basic idea behind the "uniform τ-reset strategy" outlined by Neuder, Rao, Moroz, & Parkes. It defines a number of price bins, corresponding to price intervals. Once the current price moves out of the current allocation, the strategy resets, re-centreing itself and re-allocating across a new set of price bins. The authors present three different reset strategies: proportional, uniform and optimal. This work therefore takes in account impermanent loss through both the dimension of price change and with time.

3.4.2.6 Fee structure

As outlined in 2.3.3, Uniswap v3 introduces four fee tiers that the Uniswap team expect to better represent rewards according to the volatility of the assets in the liquidity pool. This presents a choice to liquidity providers coupling the rewards with the possibility of impermanent loss. With more volatile tokens, they can select a higher fee tier, therefore earning more fees to compensate for the increased probability of impermanent loss. For more stable assets, for instance stablecoins linked to real world assets like USD Coin ($USDC$) or Tether ($USDT$), a lower fee tier can be selected, since the assets have a near $1{:}\,1$ relationship. This element also introduces competition for fee tier selection between liquidity providers. Those less risk averse liquidity providers may be willing to accept lower fees if they think they can capture a bigger slice of the generated fees. Or perhaps, they reason that their strategies for selecting price ranges and resetting positions is superior to other liquidity providers and they can therefore better avoid impermanent loss.

In reality, from a cursory analysis of the data, it appears that the distribution of liquidity in pools for certain token pairs settles in particular fee tiers. The impression is that this is a function of both a consensus around the volatility of a token pair and competition between liquidity providers. The $WBTC - WETH$ token pair shows how liquidity providers centre themselves on certain fee tiers:

Table 1 Data on WBTC-WETH liquidity pools sourced from Uniswap v3 web interface, 12 June 2023

Fee tier	**Total Value Locked**	**Price (1 $WBTC$ in $WETH$)**	**Contract**
0.01%	$5.06k	14.8571	(0xe6ff8b9a37b0fab776134636d9981aa778c4e718, 2023)
0.05%	$98.38m	14.8612	(0x4585fe77225b41b697c938b018e2ac67ac5a20c0, 2023)
0.3%	$213.66m	14.8522	(0xcbcdf9626bc03e24f779434178a73a0b4bad62ed, 2023)
1%	$136.43k	14.7672	(0x6ab3bba2f41e7eaa262fa5a1a9b3932fa161526f, 2023)

Table 1 illustrates how liquidity providers have concentrated their provision on the two middle fee tiers 0.05% and 0.3% pools. The total value locked is the equivalent value in USD of all the tokens deposited in the pool, and for two largest pools represents over 99.9% of the total liquidity for the token pair across the four pools in the given snapshot. The price is slightly deceptive since it represents the equilibrium price, whereas for an actual swap the execution price is going to be affected by the price impact on the pool, which for a smaller pool is bigger. Therefore, the small pools will offer less competitive pricing for someone wanting to carry out a swap.

Figure 18 Total Value Locked within WBTC-WETH pool for 0.05% fee tier, May 2021 to June 2023, via Dune.com analytics

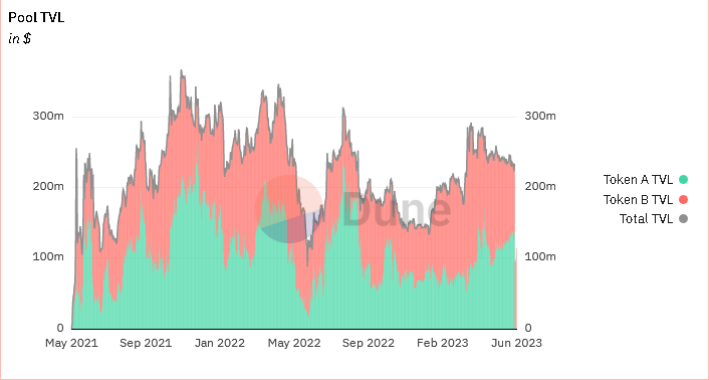

Figure 19 Total Value Locked within WBTC-WETH pool for 0.3% fee tier, May 2021 to June 2023, via Dune.com analytics

Figure 18 and Figure 19 illustrate how the total value locked changed over time for the two largest pools. The 0.3% fee pool initially saw the greatest total value locked. The 0.05% fee pool then increased, reaching a peak in November 2021. This increase did not correspond with a drop in the total value locked within the 0.3% fee pool. However, later on in June 2022

we see a drop in the liquidity in 0.3% fee pool and a corresponding increase in the 0.05% fee pool.

This example demonstrates that certain fee tiers emerge as the most popular choice for liquidity providers. This consensus for liquidity providers can change over time as token prices become more or less volatile, and with expected trading volumes. Furthermore, that support for more than one fee tier can coexist from liquidity providers settling on a tier they are comfortable with.

3.4.2.7 Changing positions and gas fees

We have previously mentioned how fees for swaps are composed of fees paid to liquidity providers as well as transaction fees for executing the trade on the blockchain. Changing liquidity positions also entails a fee for changing the contract on the blockchain, therefore incurring gas fees. According to a widely used Ethereum analytics website (Ethereum Gas Tracker, 2023), the gas fees for adding liquidity to a Uniswap v3 pool is currently about $6.80. Unlike providing liquidity on a CEX, the fees for the management of liquidity positions must also be taken into account. It seems reasonable that regular, active management of a liquidity position is desirable especially given the importance of selecting price ranges and avoiding impermanent loss. Yet, by altering a liquidity position too frequently a provider will be spending gas fees that eat into their profit. As such it is appears logical that the risk of impermanent loss against fee generation must also be balanced with the cost of active management of a position.

4. Conclusion

Some cryptocurrency fans mistakenly believed that the innovation offered by Uniswap offered some sort of magic passive income (MacDonald, 2021). This is clearly not the case, and in fact, price volatility is amplified thanks to the constant product function. With Uniswap v3, this behaviour is further augmented, leaving liquidity providers with a fundamental balancing act between potential impermanent loss and possible fees. It seems no coincidence that changes to the latest version resulted in four different fee tiers, acting as a classification of the volatility of the underlying token pair. This implicitly attempts to rebalance equation (16), enabling liquidity providers to earn more fees for a volatile pair that has potentially more impermanent loss.

The other changes to Uniswap v3's economic design, notably the ability to select price ranges, introduce the possibility of much more sophisticated liquidity strategies. These resemble those undertaken by market makers in electronic limit order books operated in traditional financial markets for several decades. However, there are a two key differences. Firstly, operations on Uniswap are completely transparent given the nature of the blockchain. Secondly, liquidity providers do not need a relationship with the exchange. On the one hand this allows users to explore and interrogate the behaviour of profitable liquidity providers and attempt to emulate their strategies. It also allows retail investors to compete on a level playing field with institutional investors, where the barriers to entry are purely based on the resources at their disposal and capacity to understand and mobilise the available mechanisms. And not the relationship between a given centralised exchange and market maker. In fact, a cottage industry of cryptocurrency projects has emerged catering to exactly this. So-called "liquidity

mining" where decentralised solutions offer the opportunity to target the most profitable strategies on decentralised exchanges.

This does appear to somewhat democratise the idea of market making, fitting into the ideals of decentralised finance, removing third parties, and opening up financial infrastructure to the masses. That being said, some of the risks involved in providing liquidity do not seem to be fully understood by less sophisticated investors. Furthermore, it appears that big players and those with a first-mover advantage carry outsized benefits over smaller players or retail investors. When weighing up the potential for impermanent loss and fee generation, having a sophisticated staff able to effectively monitor cryptocurrency markets, parsing news events quickly or understanding the flow of capital within the markets, means that the "whales" will almost always dominate the "fish".

Nevertheless, the innovation offered by Uniswap is unique. It offers competition to the economic design of traditional limit order books that have long reigned. The feat of programming such functionality into a restrictive environment such as a decentralised blockchain must also be recognised. Given the recent regulatory focus on cryptocurrencies in general, it remains to be seen whether decentralised exchanges will completely replace their centralised counterparts. But there are also questions about the impact of designs such as Uniswap on the greater financial industry at large. Not just with respect to their decentralised nature, but rather with the implementation of an automated market maker that everyone can participate in.

5. References

0x4585fe77225b41b697c938b018e2ac67ac5a20c0. (2023, June 12). Retrieved from Pool WBTC - WETH 0.05%, Uniswap v3 web interface: https://info.uniswap.org/#/pools/0x4585fe77225b41b697c938b018e2ac67ac5a20c0

0x6ab3bba2f41e7eaa262fa5a1a9b3932fa161526f. (2023, June 12). Retrieved from Pool WBTC - WETH 1%, Uniswap v3 web interface: https://info.uniswap.org/#/pools/0x6ab3bba2f41e7eaa262fa5a1a9b3932fa161526f

0xcbcdf9626bc03e24f779434178a73a0b4bad62ed. (2023, June 12). Retrieved from Pool WBTC - WETH 0.3%, Uniswap v3 web interface: https://info.uniswap.org/#/pools/0xcbcdf9626bc03e24f779434178a73a0b4bad62ed

0xe6ff8b9a37b0fab776134636d9981aa778c4e718. (2023, June 12). Retrieved from Pool WBTC - WETH 0.01%, Uniswap v3 web interface: https://info.uniswap.org/#/pools/0xe6ff8b9a37b0fab776134636d9981aa778c4e718

Adams, H. (2018, April 29). *Uniswap Whitepaper*. Retrieved from HackMD: https://hackmd.io/@HaydenAdams/HJ9jLsfTz

Adams, H. (2019, February 11). *A short history of Uniswap*. Retrieved from Uniswap Labs Blog: https://blog.uniswap.org/uniswap-history

Adams, H., Zinsmeister, N., & Robinson, D. (2020, March 2020). Uniswap v2 Core. *Uniswap*, 1-10. Retrieved from Uniswap: https://uniswap.org/whitepaper.pdf

Adams, H., Zinsmeister, N., Salem, M., Keefer, R., & Robinson, D. (2021, March). Uniswap v3 Core. *Uniswap*, 1-9. Retrieved from Uniswap: https://uniswap.org/whitepaper-v3.pdf

Barbon, A., & Ranaldo, A. (2022, October 3). On the quality of cryptocurrency markets - Centralized versus decentralized exchanges. *arXiv*, 15-36. Retrieved from https://arxiv.org/pdf/2112.07386.pdf

Contract: 0xB4e16d0168e52d35CaCD2c6185b44281Ec28C9Dc. (2023, June 12). Retrieved from Etherscan: https://etherscan.io/address/0xb4e16d0168e52d35cacd2c6185b44281ec28c9dc

Core Concepts: Flash Swaps. (2023, June 12). Retrieved from Uniswap Docs: https://docs.uniswap.org/contracts/v2/concepts/core-concepts/flash-swaps

Deng, J., Zong, H., & Wang, Y. (2023). Static replication of impermanent loss for concentrated liquidity provision in decentralised markets.

Elsts, A. (2022, January 9). *Uniswap v3 liquidity formula explained*. Retrieved from Medium: https://atise.medium.com/uniswap-v3-liquidity-formula-explained-de8bd42afc3c

Ethereum Average Block Time Chart. (2023, June 12). Retrieved from Etherscan: https://etherscan.io/chart/blocktime

Ethereum Gas Tracker. (2023, June 12). Retrieved from Etherscan: https://etherscan.io/gastracker

Hanson, R. (2002). Logarithmic market scoring rules for modular combinatorial information aggregation.

Heimbach, L., Schertenleib, E., & Wattenhofer, R. (2022, September 21). Risks and returns of Uniswap v3 liquidity providers. *arXiv*, 3-4. Retrieved from https://arxiv.org/pdf/2205.08904.pdf

Introducing Uniswap v3. (2021, March 23). Retrieved from Uniswap Labs Blog: https://blog.uniswap.org/uniswap-v3

Khakhar, A., & Xi, C. (2022). Delta hedging liquidity positions on automated market makers.

MacDonald, C. (2021). *3 reasons to buy uniswap*. Retrieved from The Motley Fool: https://www.fool.com/investing/2021/11/18/3-reasons-to-buy-uniswap/

Neuder, M., Rao, R., Moroz, D. J., & Parkes, D. C. (2021). Strategic Liquidity Provision in Uniswap v3. *arXiv*, 4. Retrieved from https://arxiv.org/pdf/2106.12033.pdf

Park, D., Zhang, Y., & Cheng, X. (2018, October 24). Formal specification of constant product (x.y = k) market maker model and implementation. *Runtime Verification, Inc*, 1-14. Retrieved from Runtime Verification Inc: https://github.com/runtimeverification/verified-smart-contracts/blob/uniswap/uniswap/x-y-k.pdf

5. Appendix A

1. Rearrangement of equation making Δx and Δy the subject.

$$x.y = (x + \Delta x) \times (y - \Delta y)$$
$$x.y = x.y + \Delta x.y - \Delta y.x - \Delta x.\Delta y$$
$$0 = \Delta x.y - \Delta y.x - \Delta x.\Delta y$$
$$\Delta y.x = \Delta x(y - \Delta y)$$
$$\Delta x = \frac{\Delta y.x}{y - \Delta y}$$

$$x.y = (x + \Delta x) \times (y - \Delta y)$$
$$x.y = x.y + \Delta x.y - \Delta y.x - \Delta x.\Delta y$$
$$0 = \Delta x.y - \Delta y.x - \Delta x.\Delta y$$
$$\Delta x.y = \Delta y(x + \Delta x)$$
$$\Delta y = \frac{\Delta x.y}{x + \Delta x}$$

2. Demonstration that α and β represent the same relationship.

$$y - \Delta y = \frac{1}{1 + \alpha} y$$

substituting in $\alpha = \frac{\Delta x}{x}$

$$y - \Delta y = \frac{1}{1 + \frac{\Delta x}{x}} y$$
$$\frac{y - \Delta y}{1} \Big/ \frac{1}{1 + \frac{\Delta x}{x}} = y$$
$$(y - \Delta y)\left(1 + \frac{\Delta x}{x}\right) = y$$
$$y - \Delta y + y\frac{\Delta x}{x} - \Delta y\frac{\Delta x}{x} = y$$
$$y\frac{\Delta x}{x} - \Delta y\frac{\Delta x}{x} = \Delta y$$
$$\frac{\Delta x}{x}(y - \Delta y) = \Delta y$$
$$\frac{\Delta x}{x} = \frac{\Delta y}{y - \Delta y}$$
$$\Delta x = \frac{\Delta y.x}{y - \Delta y}$$

$$x + \Delta x = \frac{1}{1-\beta}x$$

substituting in $\beta = \frac{\Delta y}{y}$

$$x + \Delta x = \frac{1}{1-\frac{\Delta y}{y}}x$$

$$\frac{x+\Delta x}{1} \Big/ \frac{1}{1-\frac{\Delta y}{y}} = x$$

$$(x + \Delta x)\left(1 - \frac{\Delta y}{y}\right) = x$$

$$x + \Delta x - x.\frac{\Delta y}{y} - \Delta x.\frac{\Delta y}{y} = x$$

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

$$x.\frac{\Delta y}{y} + \Delta x.\frac{\Delta y}{y} = \Delta x$$

$$\frac{\Delta y}{y}(x + \Delta x) = \Delta x$$

$$\frac{\Delta y}{y} = \frac{\Delta x}{x + \Delta x}$$

$$\Delta y = \frac{\Delta x.y}{x + \Delta x}$$

Therefore, the two equations for Δx and Δy are the same as detailed in Appendix A1.

6. Appendix B

Solving equation (5) for x, y and L:

For simplicity, using the same notation as the whitepaper where P_a, P_b are used instead of P_{lower}, P_{upper}

when $x = 0$:

$$\left(\frac{L}{\sqrt{P_b}}\right)\left(y + L\sqrt{P_a}\right) = L^2$$

$$\frac{Ly}{\sqrt{P_b}} + \frac{L^2\sqrt{P_a}}{\sqrt{P_b}} = L^2$$

$$Ly + L^2\sqrt{P_a} = L^2\sqrt{P_b}$$

$$\frac{Ly}{L^2} + \sqrt{P_a} = \sqrt{P_b}$$

$$y.\frac{1}{L} = \sqrt{P_b} - \sqrt{P_a}$$

$$\therefore y = L\left(\sqrt{P_b} - \sqrt{P_a}\right)$$
$$\therefore L = \frac{y}{\sqrt{P_b} - \sqrt{P_a}}$$

when $y = 0$:

$$\left(x + \frac{L}{\sqrt{P_b}}\right)\left(L\sqrt{P_a}\right) = L^2$$
$$x.L\sqrt{P_a} + \frac{L^2\sqrt{P_a}}{\sqrt{P_b}} = L^2$$
$$x.L\sqrt{P_a}\sqrt{P_b} + L^2\sqrt{P_a} = L^2\sqrt{P_b}$$
$$x.\frac{1}{L}\sqrt{P_a}\sqrt{P_b} + \sqrt{P_a} = \sqrt{P_b}$$
$$x.\frac{1}{L} = \frac{\sqrt{P_b} - \sqrt{P_a}}{\sqrt{P_a}\sqrt{P_b}}$$
$$\therefore x = L.\frac{\sqrt{P_b} - \sqrt{P_a}}{\sqrt{P_a}\sqrt{P_b}}$$
$$\therefore L = x\frac{\sqrt{P_a}\sqrt{P_b}}{\sqrt{P_b} - \sqrt{P_a}}$$

7. Appendix C: Examples

7.1 Simple swap

To demonstrate this idea, we can take an example cryptocurrency pair using Ether (ETH) and USD Coin ($USDC$). We will use these representations in place of x and y in this illustration.

If the pool contains 80 ETH and 2000 $USDC$ locked in the liquidity pool our constant is as such:

$$80\ ETH \times 2000\ USDC = 160{,}000$$

Let us imagine a user called Vitalik who has 10 ETH they want to exchange for $USDC$. This swap would result in the number of $USDC$ in the pool dropping and the number of ETH increasing, but the value of the constant k would remain unchanged. The operation would be as follows:

$$(80\ ETH + 10\ ETH) \times (2000\ USDC - equivalent\ USDC)\ = 160{,}000$$
$$equivalent\ USDC = 222.22$$

So, Vitalik swaps 10 ETH and gets 222.22 $USDC$ in return. Following the swap, the invariant remains the same.

$$90\ ETH\ \times 1777.78\ USDC = 160{,}000$$

The price obtained by Vitalik is obtained as follows:

$$Price_{ETH} = \frac{\Delta y}{\Delta x}$$
$$Price_{ETH} = \frac{222.22}{10}$$
$$Price_{ETH} = 22.22\ USDC$$

Equally, we can calculate the price of $USDC$ by using the reciprocal:

$$Price_{USDC} = \frac{\Delta x}{\Delta y}$$
$$Price_{USDC} = \frac{10}{222.22}$$
$$Price_{USDC} = 0.045$$

7.2 Swap with fees

Continuing the example swap with Vitalik incorporating fees:

$$Liquidity\ pool{:}\ 80\ ETH\ and\ 2000\ USDC. Swapping\ 10\ ETH$$

$$\alpha = \frac{10}{80}$$

$$\Delta y = \frac{\frac{1}{8} \times 0.997}{1 + \frac{1}{8} \times 0.997} \times 2000$$

$$\Delta y = 221.63$$

So Vitalik receives 221.63 $USDC$ for his 10 ETH compared to the 222.22 $USDC$ he received in the last example without the fee. He paid 0.59 $USDC$ in fees which was retained by the liquidity pool for providing the service.

We can also see how this has impacted the reserves in the liquidity pool and the invariant.

$$90\ ETH\ \times 1778.37\ USDC = 160{,}053.3$$

The invariant has grown. So, any subsequent swaps will take place using a marginally bigger invariant and slightly larger token reserves. Depending on the nature of the trade, selling ETH and buying $USDC$, or the opposite, the token reserves of the asset being bought will increase.

7.3 Adding liquidity to a pool

Taking a liquidity pool containing tokens of Ether (ETH) and Wrapped Bitcoin ($WBTC$). Wrapped Bitcoin is a representation of Bitcoin using the ERC20 token standard. For all intents and purposes, we can imagine it as being equal to a normal Bitcoin token. In this example we will imagine Satoshi has both a stock of ETH and $WBTC$ in his possession that he wants to add to a liquidity pool. This liquidity pool had been initialised by another liquidity provider who added what they believed was an equivalent amount of the token pair, and subsequently received 5,000 UNI tokens, since the initial supply of liquidity tokens is set to the initial deposit of ETH. The pool currently looks as follows:

$$e = 5000\ ETH$$
$$t = 15\ WBTC$$
$$l = 5000\ UNI$$

Satoshi has 88 ETH

$$\alpha = \frac{88}{5000}$$
$$e' = \left(1 + \frac{88}{5000}\right) \times 5000 = 5088$$
$$t' = \left(1 + \frac{88}{5000}\right) \times 15 = 15.26$$
$$l' = \left(1 + \frac{88}{5000}\right) \times 5000 = 5{,}088$$
$$\Delta t = 15.26 - 15 = 0.26$$
$$\Delta l = 5{,}088 - 5{,}000 = 88$$

So, to maintain the invariant Satoshi must deposit 0.26 $WBTC$ alongside the 88 ETH he possesses. He receives 88 UNI tokens representing his portion of the liquidity in the pool. In the terminology used within the cryptocurrency ecosystem, Satoshi has “minted” 88 UNI tokens.

We can validate the statements in the theorem as such:

1. $5000 : 15 : 5000 = 5088 : 15.26 : 5088$
   $\frac{5000}{15} = 333.33, \quad \frac{5088}{15.26} = 333.33, \quad \frac{15}{5000} = 0.003, \quad \frac{15.26}{5088} = 0.003$
   $\therefore e : t : l = e' : t' : l'$
2. $k = 5000 \times 15$
   $k = 75{,}000$
   $k' = 5088 \times 15.26$
   $k' = 77{,}642.88$
   $\therefore k < k'$
3. $\frac{k'}{k} = \frac{77{,}642.88}{75{,}000} = 1.04$
   $\left(\frac{l'}{l}\right)^2 = \left(\frac{5088}{5000}\right)^2 = 1.04$
   $\therefore \frac{k'}{k} = (\frac{l'}{l})^2$

7.3 Removing liquidity from a pool and accrued fees

Returning to the previous example with Satoshi who added liquidity to the pool. But this time, before removing liquidity we will carry out two transactions to help demonstrate how his token balances have been impacted by the fees accrued in the interim.

A reminder of the way the pool currently stands:

$$e = 5088, \qquad t = 15.26, \qquad l = 5088$$

Satoshi added $e = 88$ to the pool in the form of Ether and $t = 0.26$ in the form of Wrapped Bitcoin. In return, to represent his share of the fee revenue, he received $l = 88$, which is the UNI liquidity token.

Transaction 1: Buy 20 ETH worth of $WBTC$ with fee $\rho = 0.003$ (0.3%)

$$\alpha = \frac{20}{5088} = 0.0039$$
$$\Delta y = \frac{0.0039 \times (1 - 0.003)}{1 + (0.0039 \times (1 - 0.003))} \times 15.26 = 0.059$$

This purchase results in 0.059 $WBTC$ and the pool is adjusted as follows:

$$e' = 5088 + 20 = 5108, \qquad t' = 15.26 - 0.059 = 15.20$$

Transaction 2: Buy 17 ETH worth of $WBTC$ with fee $\rho = 0.003$ (0.3%)

$$\alpha = \frac{17}{5108} = 0.0033$$
$$\Delta y = \frac{0.0033 \times (1 - 0.003)}{1 + (0.0033 \times (1 - 0.003))} \times 15.20 = 0.05$$

This purchase results in 0.05 $WBTC$ and the resulting pool looks like this:

$$e'' = 5108 + 17 = 5125, \qquad t'' = 15.20 - 0.05 = 15.15$$

We will now withdraw Satoshi's liquidity:

$$\alpha = \frac{88}{5088} = 0.017$$
$$e''' = (1 - 0.017) \times 5125 = 5037.88$$
$$t''' = (1 - 0.017) \times 15.151 = 14.89$$
$$l''' = (1 - 0.017) \times 5088 = 5000$$

After removing the liquidity, the pool contains the same number of UNI tokens, $l''' = 5000$, as before Satoshi added his tokens. He will receive:

$$ETH = 5088 - 5037.88 = 50.12$$
$$WBTC = 15.26 - 14.89 = 0.37$$

His holding of Ether has been reduced, but he has more Wrapped Bitcoin. It is important to note here the fees he has accrued. The calculation of the fees accrued by a liquidity provider are not covered in the Runtime Verification Inc formal specification, neither are they detailed in the Uniswap Whitepaper, but we can nevertheless deduce this using the equation (*3*).

Transaction 1. saw 20 ETH exchanged for 0.059 $WBTC$. Given the small fee size we will have to demonstrate this using more decimal places.

Without fee:

$$\Delta y = \frac{20 \times 15.264}{5088 + 20} = 0.059765074$$

With fee revisited:

$$\Delta y = \frac{\frac{20}{5088} \times (1 - 0.003)}{1 + (\frac{20}{5088} \times (1 - 0.003)} \times 15.264 = 0.059586479$$

Therefore, the fee is worth:

$$0.059765074 - 0.059586479 = 0.000178595$$

Transaction 2.

Without fee:

$$\Delta y = \frac{17 \times 15.20441352}{5108 + 17} = 0.050434152$$

With fee revisited:

$$\Delta y = \frac{\frac{17}{5108} \times (1 - 0.003)}{1 + (\frac{17}{5108} \times (1 - 0.003)} \times 15.20441352 = 0.050283344$$

Therefore, the fee is worth:

$$0.050434152 - 0.050283344 = 0.000150808$$

The fees generated from the two transactions are therefore:

$$0.000178595 + 0.000150808 = 0.000329403$$

Since both transactions have added ETH to and removed $WBTC$ from the pool, the generated fees have accrued in $WBTC$.

Satoshi's share of the liquidity pool fees is given by the portion of UNI tokens he holds:

$$0.000329403 \times \frac{88}{5088} = 0.000005697\ WBTC$$

If we review Satoshi's portfolio, we can see how much of his new $WBTC$ balance has been come from fees.

$$ETH = 50.12$$
$$WBTC = 0.37$$
$$fees = 0.000005697$$
$$WBTC - fees = 0.369994303$$

As the example illustrates, the fee is accrued at the exchange rate of each swap, plus, as we have already detailed, the exchange rate of the swap is determined by the invariant at a point in time.

We could once again validate the theorem presented for withdrawing liquidity.

1. $5125 : 15.15 : 5088 = 5037.88 : 14.89 : 5000$
   $\frac{5125}{15.15} = 338.28, \quad \frac{5037.88}{14.89} = 338.34, \quad \frac{15.15}{5088} = 0.003, \quad \frac{14.89}{5000} = 0.003$
   $\therefore e : t : l = e' : t' : l'$
2. $5125 \times 15.15 = 77643.75,\ 5037.88 \times 14.89 = 75014.03$
   $\therefore k' < k$
3. $\frac{75014.03}{77643.75} = 0.97,\ (\frac{5000}{5088})^2 = 0.97$
   $\therefore \frac{k'}{k} = (\frac{l'}{l})^2$

Note the rounding error we are starting to introduce in statement 1. Nevertheless, these properties hold true for the withdrawal of liquidity. The formal specification also outlines a state using the addition and subsequent removal of liquidity, where the amount removed is equal to the amount added. This shows egality between the first and last state, although it does not appear to add much to our explanation here.

7.4 Uniswap v2 liquidity tokens

Taking another example with Vitaly who has just added 54,000 $USDC$ and 2 $WBTC$ tokens to a liquidity pool. He will receive the following UNI tokens to represent his share of the liquidity pool:

$$l = \sqrt{54{,}000 \times 2}$$
$$l = 328.63$$

Now if Satoshi adds 20,000 $USDC$ and 0.54 $WBTC$, maintaining the invariant and assuming no swaps have taken place, we will receive the following UNI tokens:

$$l = \sqrt{20{,}000 \times 0.54}$$
$$l = 103.92$$

So Vitaly has $\frac{328.63}{328.63+103.92} = 76\%$ of the liquidity pool and Satoshi has $\frac{103.92}{328.63+103.92} = 24\%$.

7.5 Impermanent loss in Uniswap v2

We will show how impermanent loss impacts liquidity provider Satoshi in a $USDC - WBTC$ pool. The pool initially contains:

$$USDC = 75{,}000$$

$$WBTC = 3$$

$$Price_{equilibrium} = \frac{75{,}000}{3} = 25{,}000\ USDC$$

Satoshi wants to participate in the pool and has 10,000 $USDC$. The equivalent amount of $WBTC$ tokens is determined as in 2.1.3 and owns 11.8% of the pool.

Number of $USDC$ tokens	**Number of $WBTC$ tokens**
10,000	0.4

Now, imagine that $WBTC$ in is high demand, some news has hit cryptocurrency news websites, with Acme Pizza Corporation says it is going to accept payments in $WBTC$. We can simulate three swaps chronologically occurring in the liquidity pool all paying in $USDC$ to receive $WBTC$. We will ignore the fees earnt by liquidity providers.

The amount received in $WBTC$ is calculated using equation (3)

Swapper	**Paid $USDC$ (Δx)**	**Received $WBTC$ (Δy)**	**$USDC$ in pool after swap (x)**	**$WBTC$ in pool after swap (y)**	**$Price\ \frac{\Delta x}{\Delta y}$**
Tyler	150	0.0059894	85,150	3.39	25,044.12
Cathie	10,000	0.3567011	95,150	3.04	28,034.69
Zhao	325	0.0103391	95,475	3.03	31,434.07

It is quite clear that as subsequent swaps take place, the execution price goes up, but what about the impact on Satoshi? The composition of the pool looks as follows:

$$USDC = 95{,}475$$

$$WBTC = 3.03$$

Satoshi's share	**$USDC$ tokens**	**$WBTC$ tokens**
11.8%	11232.35	0.36

We express Satoshi's portfolio using the same numeraire and the equilibrium price of the pool:

$$Price = \frac{x}{y}$$

$$Price = \frac{84{,}243}{2.67} = 31{,}541.44$$

Number of $USDC$ tokens	**Value of $WBTC$ tokens in $USDC$**	**Total value (in $USDC$)**
11,232.35	11,232.35	22,464.71

At first glance it appears as if Satoshi has profited from the increasing demand for $WBTC$ and higher prices since his original portfolio upon depositing it into the pool was worth 20,000 $USDC$. But what about if Satoshi had simply held his tokens and not participated in the pool? Evaluating his original portfolio with the latest equilibrium price is much more revealing:

Number of $USDC$ tokens	**Value of $WBTC$ tokens (in $USDC$)**	**Total value (in $USDC$)**
10,000	12,616.58	22,617

So, with the increasing price of $WBTC$ Satoshi would have made the equivalent of 2,617 $USDC$ of profit if he had simply held his tokens. But by providing liquidity in the pool his tokens are now worth 22,464.71 $USDC$ rather than 22,617 $USDC$. The difference between these two values is the impermanent loss and, in this example, represents a loss of 0.68%. He would not have been subject to this loss had he not provided liquidity and just held his tokens.

The effect is similar if we simulate the inverse situation where demand for $WBTC$ drops. Imagine, regulatory authorities have said they are placing new restrictions on $WBTC$ tokens and those holding the tokens now want to exchange them for $USDC$. We will assume the same conditions in the liquidity pool and Satoshi's liquidity position. And calculate three swaps approximating the same amount paid in $WBTC$ to receive $USDC$. Once again, we will ignore fees earnt by liquidity providers.

The amount received in USDC is calculated using equation (2)

Swapper	**Paid $WBTC$ (Δy)**	**Received $USDC$ (Δx)**	**$WBTC$ in pool after swap (y)**	**$WBTC$ in pool after swap (y)**	**$Price\ \frac{\Delta x}{\Delta y}$**
Tyler	0.006	150.2651738	3.41	84,850	25,044.20
Cathie	0.36	10028.20241	3.77	74,822	27,856.12
Zhao	0.010	199.2053579	3.78	74,622	19920.54

Afterwards the pool looks as follows:

$$USDC = 74{,}622$$

$$WBTC = 3.78$$

Satoshi's share	**$USDC$ tokens**	**$WBTC$ tokens**
11.8%	8,779.10	0.44

Following the same logic, we can once again express Satoshi's portfolio using the same numeraire and the equilibrium price:

$$Price = \frac{x}{y}$$

$$Price = \frac{74{,}622}{3.78} = 19{,}762.27$$

Number of $USDC$ tokens	**Value of $WBTC$ tokens in $USDC$**	**Total value (in $USDC$)**
8,779.10	8,779.10	17,558.19

Satoshi's original portfolio before providing liquidity would look as follows had he simply held his tokens:

Number of $USDC$ tokens	**Value of $WBTC$ tokens (in $USDC$)**	**Total value (in $USDC$)**
10,000	7,904.91	17,905

So, he loses 346.71 $USDC$ suffering an impermanent loss of 1.93%.