Banker Joe: DeFi Lending for DEX leveraged trading

Cryptofish & 0xMurloc
www.traderjoexyz.com
V1.0, Aug 2021

Abstract

Trader Joe is a one-stop decentralized trading platform on Avalanche. We introduce *Banker Joe*, a lending protocol based on the Compound[1] protocol. We then show how our lending protocol can be combined with our decentralized exchange to offer non-custodial leveraged trading.

Introduction

Motivation

The market for cryptocurrencies within Avalanche has progressed significantly since the first dApps were conceived back in February 2021, creating a vibrant ecosystem of investors, speculators and traders. However, when it comes to incentivizing token holding, there is currently only one option: staking them in liquidity mining programs ("farms").

Decentralized lending presents many opportunities as the next stage of DeFi evolution. Similar to decentralized exchanges, lending has two parties involved: the *borrower* and the *supplier*. Suppliers deposit assets into a pool to earn interest. Borrowers deposit assets for collateral, and borrows other assets. In order to borrow an asset, the borrower pays interest *over time*, paid to suppliers, with a share to the protocol.

This presents new opportunities for DeFi participants to further the growth of an ecosystem:

- Instead of staking tokens in liquidity mining programs, users can now lend out their tokens. This has the added advantage of earning yield on their tokens *without impermanent loss* and not having to hold two tokens at a fixed ratio as you must do when you liquidity mine.
- Users can deposit their tokens as collateral to borrow more tokens of another class. This presents the opportunity to trade *on leverage.*

The second point is crucial to our overall plan of turning Trader Joe into a *one-stop trading platform*; lending and collateralization enables leveraged spot trading, and unlocks possibilities for derivatives.

However, these benefits are not risk-free. For borrowers, the main risk is *liquidation* and for suppliers, the main risk is not being able to withdraw your deposited tokens in a *shortfall event*.

[1] Compound: The money market protocol, Leshner & Hayes, 2019

Lending

Protocols such as *Aave*, *Compound* and *Uniswap* spurred the growth of DeFi with algorithmic, pool-based liquidity and lending strategies. Suppliers provide liquidity by depositing tokens into a pool contract. The pool of tokens can also be borrowed by placing tokens to the pool contract as collateral. Loans can instantly be issued without matching lenders to borrowers, instead relying on pooled tokens and state of the pool to calculate interest rates.[2]

Leveraged Trading

Crypto exchanges such as Binance and FTX offer leveraged trading via margin or derivative trading. In this paper, we will focus on the leveraged spot trading use case. Leveraged spot trades can be decomposed to asset collateralization, borrowing, and then a trade.

For example:

- Margin long trades can be made by using tokens as collateral, and borrowing USDT for the long trade.
- Margin short trades can be made by using tokens or USD as collateral, and borrowing tokens for the short trade.

Exchanges such as FTX also allow users to participate in lending[3], by offering a two-sided market for lending and borrowing interest rates. We observe that leverage trading is a major use case for crypto token borrowing, and that similar synergy can also provide value for decentralized exchanges.

Decentralized leveraged trading

Sushiswap

Sushiswap is an Automated Market Maker (AMM) that also offers integrated lending, via their *Kashi* feature[4]. Leveraged trades can be performed by providing collateral to and borrowing from Kashi, and then swapping tokens on the AMM. A key distinction of *Kashi* is its isolated lending pools.

Rather than borrowing tokens from a shared pool such as in *Aave*, *Kashi* loans are made from individual token pairs. For example, to borrow USDT against ETH collateral, the borrower would have to post ETH tokens against the specific ETH/USDT pool. The interest rate and loan risks are specific only to the ETH/USDT pool. Although Kashi is less prone to loan risk, it is less

[2] Aave Protocol Whitepaper, Jan 2020
[3] FTX spot margin lending, https://ftx.com/intl/spot-margin/lending
[4] Sushiswap, https://docs.sushi.com/products/kashi-lending

convenient for traders as there is added friction in comparing interest rates between different pairs.

Example:

- Assuming 1 ETH = $3000 and collateral factor is 65%.
- Deposit and collateralize 1 ETH to ETH/USDT pool.
- Borrow 1500 USDT from the ETH/USDT pool against the collateralized ETH.
- Swap 1500 USDT for 0.5 ETH
- User has used 1 ETH and swapped his borrowed amount for 0.5 ETH. So he is now 0.5x leveraged long on ETH.

WOWswap

WOWswap[5] is a decentralized leveraged trading protocol that allows users to borrow tokens for leverage trades. The protocol itself facilitates token lending and borrowing, and then performs trades on behalf of users on integrated AMMs such Pancakeswap. In this design, the swapped tokens are held by the protocol as collateral against borrowed assets. This strategy may appear to be more convenient to the trader, as no additional collateral is provided before performing the trade. However, the custodial design with the trader receiving a proxy-token is effectively a derivative contract, with traders and lenders sharing exposure to the swapped asset.

Example:

- Send 1 BNB to contract and borrow 4 BNB.
- Contract swaps 5 BNB for 100 CAKE.
- Contract mints 300 proxy-CAKE tokens to the trader.

Design

Since its inception in 2020, many lending protocol variants have emerged. They can be largely classified into two categories:

- **Isolated lending pool (Sushiswap Kashi, Impermax[6]):** Similar to a Uniswap v2 liquidity pair, an isolated lending pool only allows borrowing and lending between two assets, for example, AVAX-USDT. Furthermore, each pair requires two separate markets, each denoting which asset is the collateral asset and which is the borrowable asset. This has the added benefit of limiting risk to a certain pair, but at the cost of fragmenting liquidity significantly.
- **Grouped lending pool (Compound, CREAM[7], Aave):** All assets are grouped into a single pool, which means liquidity is not fragmented. However, the disadvantage is that if a risky asset is added then that could jeopardize the entire pool. As a result, protocols using this approach usually have a strict criteria for listing new assets.

[5] WOWswap, https://docs.wowswap.io/about/wowprotocol
[6] Impermax, https://www.impermax.finance/
[7] CREAM finance, https://docs.cream.finance/

We believe that a grouped lending pool provides several advantages for the purposes of offering leveraged trading. For example, if a trader wants to borrow tokens, a grouped lending pool offers more concentrated liquidity without needing to compare interest rates, compared to isolated lending pools. In order to offer leveraged trading for as many tokens as possible, we also considered an approach that can reasonably scale for long-tail assets.

Protocol Architecture

Markets

We adopt a lending system similar to *Compound Protocol*[5], where each asset is defined by a *market*, which is implemented by a *jToken* contract, which is itself an ERC-20 contract. The *jToken* contract is where suppliers supply assets and borrowers borrow the same asset.

Suppliers

Using AVAX as an example, a supplier wanting to supply AVAX will deposit AVAX into the jAVAX contract. In return, the contract will mint jAVAX tokens which act as a receipt of deposit, similar to an LP token. Internally, the contract has an exchange rate that defines how much AVAX each jAVAX is worth. This exchange rate is ever-increasing due to interest accrued, so over time, the supplier will be able to redeem the jAVAX for more AVAX tokens than when he first deposited.

Borrowing

In order to borrow an asset, a borrower must supply collateral first in the same way the supplier supplied an asset. This does *not* have to be the same asset as the one the borrower wishes to borrow. For example, a borrower can supply ETH to the jETH contract and then borrow AVAX from the jAVAX contract. The amount of AVAX the borrower can borrow is defined by the **collateral factor** of ETH. If for example, the borrower has supplied 1 ETH as collateral and the collateral factor of ETH is 65%, then he can borrow up 0.65 ETH worth of AVAX.

The protocol would have safety measures to restrict users from taking actions (borrowing more tokens or withdrawing collateral) that may exceed their borrowing limit.

Interest Rate Model

The Compound protocol introduced an algorithmic interest rate model that is based on supply and demand. We define ***U***, the utilization ratio for each token market, which is the amount of underlying asset that has been borrowed.

$$U = Borrowing / (Liquidity + Borrowing)$$

The interest rate model can then be expressed as a function of the utilization ratio:

$$Borrowing\ Interest\ Rate = m^* \boldsymbol{U} + b$$

Where the parameters *m* is a multiplier that defines the slope of the line and *b* is the base rate; both of these variables are customizable for each market.

To account for the risk profile of long-tail assets, the *CREAM protocol*[8] introduced an interest rate model with three slopes instead.

Broadly speaking, *CREAM* categorizes each asset into different categories: *major*, *stable*, *governance/seed and Sushi LP*. They all adopt the same triple slope interest rate model, but the parameters differ.

For example, the *major* interest rate model as defined by *CREAM* has such a yield curve:

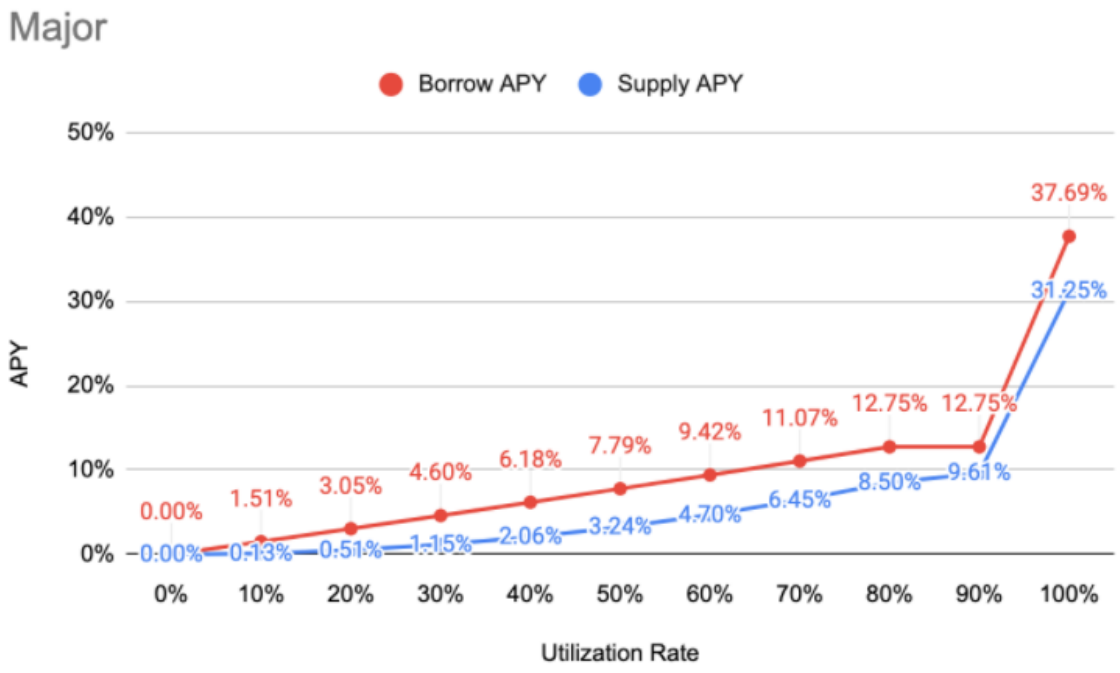

Fig 1. CREAM interest rate model

In this graph, there are three slopes:

1. From 0-80% utilization ratio, the interest rate follows a straight line with a 0.15 multiplier.
2. From 80-90% utilization ratio, the line is completely flat.
3. From 90-100% utilization ratio, the line steeps greatly with a multiplier of 2.

[8] CREAM finance, interest rate model. https://docs.cream.finance/lending/interest-rate-model

The rationale behind this interest rate model is to *dramatically* increase borrowing interest rate when a certain utilization ratio is met in order to *disincentivize* borrowing. This makes sense given a strategy of focusing on long-tail assets where the risk to the protocol is greater.

Repayment

Borrowers can repay tokens to the market up to the **borrow balance**. If partial repayment is made, the **borrow balance** may be non-zero, and continue to accrue interest. Repayment is a transfer of tokens from the borrower back to the token market.

Risk management

Reserves

The total borrowing balance of each token asset is increased every block based on the borrowing interest rate per second *r*, and the time per block *t*.

$$TotalBorrowBalance_{(n)} = TotalBorrowBalance_{(n-1)} * (1 + r * t)$$

A **reserve factor** is a percentage applied to accrued interest and retained by the protocol as a **reserve fund**. For example, if the reserve factor is 20% for a given asset, then 20% of all accrued interest goes to the reserve fund for that market and the rest is paid to the supplier. These funds serve to shield protocol from losses that may result even if liquidation does not recover sufficient obligation (**shortfall event)**.

In addition to the reserve factor, **2.8%** percent of liquidated collateral also goes to the reserves. In the liquidation example provided in the following section, a percentage of liquidated collateral is sent to protocol reserves. The reserve factor is configured for each market to account for its risk and liquidity.

Liquidation

If a borrower's borrowing balance exceeds their total collateral value, it means that the protocol is at risk of suffering a loss if the borrower defaults on repayment. We call this a **shortfall event**. In order to reduce this risk, the protocol provides a public *liquidate* function which can be called by any user.

- The callers of the liquidate function are called **liquidators**.
- The liquidate function can only be called when the borrowing balance exceeds the total collateral value.
- When called, a portion of the borrower's collateral are sold to pay back the borrowed tokens to the protocol.

Liquidators provide a risk management service to the protocol; it is important that liquidation is performed swiftly to prevent further losses. If a single market is in a **shortfall** state, i.e. the amount borrowed for an asset is greater than the amount supplied, then the *entire protocol is at*

risk. This can occur when liquidators are slow to liquidate shortfall accounts. To incentivize an ecosystem of liquidators to perform this quickly, the jToken collateral is sold at oracle price minus a **liquidation discount**. The default liquidation discount is 8%.

The portion of collateral eligible to be liquidated is defined as the **close factor,** which is the maximum repayment amount when liquidating a borrow for an underwater account. By default, the close factor for all assets will be **50%.**

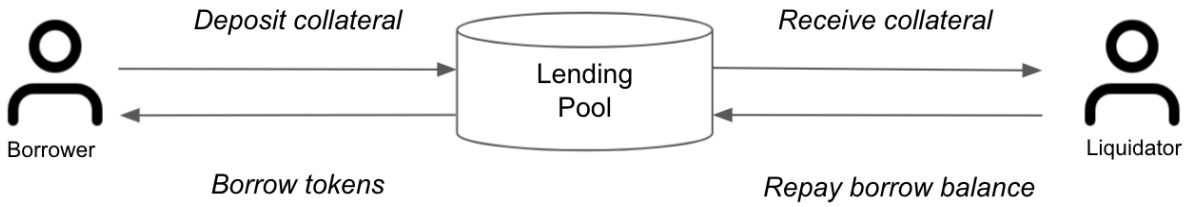

Fig 2. Liquidation flow

Liquidation Example

In this example, a borrower has collateral that encounters a shortfall event. The liquidator then liquidates some of the collateral to cover for the exposure.

1. The borrower deposits 10 AVAX as collateral.
    a. Oracle price is $100
    b. Collateral Factor is 65% for AVAX
    c. The collateral value is $650
2. The borrower borrows 650 USDT.
    a. The borrow balance is $650
    b. The borrow limit is now $0, and borrow limited used is 100%
3. The price of the collateral, AVAX drops to $90
    a. The collateral value is now $585
    b. The borrow limit is now negative: $585 - $650 = -$65
    c. The borrow limit is 111%, as the borrow balance is over the collateral value.
4. An amount of collateral is liquidated.
    a. Close factor is 50%, so $325 of collateral will be sold to the liquidator.
    b. Liquidator receives 3.9 AVAX at a liquidity discount of 8%.
    c. 2.8% of 3.9 AVAX or 0.1092 AVAX is sent to reserves.
    d. Liquidator repays 325 USDT to borrowers balance
    e. Borrow balance is now 325 USDT
    f. The liquidity is now $31.85, with 91.1% limit used

	1. Deposit collateral	2. Borrows 650 USDT	3. Shortfall event	4. Liquidator buys $325 of collateral
Collateral	10 AVAX	10 AVAX	10 AVAX	**6.1 AVAX**
Oracle Price	$100	$100	**$90**	$90
Liquidity Value	$1000	$1000	$900	**$549**
Collateral Factor	65%	65%	65%	65%
Collateral Value	$650	$650	**$585**	**$356.85**
Borrow Balance	$0	**$650**	$650	**$325**
Borrow Limit	$650	$650	**-$65**	**$31.85**
Borrow Limit used	0%	100%	**111%**	**91.1%**
Liquidation Discount	8%	8%	8%	8%
Close Factor	50%	50%	50%	50%
Reserve	0	0	0	**0.1092 AVAX**

To summarize:

- Liquidation happens when borrowing balance exceeds collateral value.
- **Liquidators** are incentivized to liquidate unhealthy loans by a **liquidity discount**.
- The amount of collateral to be liquidated is determined by the **close factor**.

Aftermath

The borrower suffered loss from collateral being sold at discount to repay their borrowing obligation. The value of the collateral is transferred to the liquidator and protocol reserves.

Borrower

- Liquidity value decreased -$351
- Borrow balance decreased +$325
- Net loss: -$26

Liquidator

- Paid -$325 USDT
- Received 3.7908 AVAX with $341.172 market value
- Net profit: $16.172

Protocol

- Received reserve 0.1092 AVAX with $9.828 market value
- Net profit: $9.828

Parameterization

Each token market can have different **reserve factor** and **collateral factor** configurations. The market level configuration allows the protocol to manage market risk. For example, stable coin markets will have a *lower* reserve factor and collateral factor. These parameters will be customized by the team at launch, and then eventually managed by community governance.

Protocol Fees

The **reserves** denote the quantity of underlying assets for each market. For example, the reserves of jAVAX will be the number of AVAX held by the jAVAX contract. Usually the reserves act as an insurance fund in the event that the jAVAX contract experiences a **shortfall event.**

However, the reserve is also a source of protocol fees. We propose that if voted by governance, then 50% of reserves for each jToken market goes to xJOE holders as **revenue** for participating in the protocol.

Price Oracles

Price oracles provide an exchange rate between two token assets. The pricing of the assets, which impact users collateral value and borrowing limit, are delegated to *Chainlink*[9], which provides robust and secure price feeds.

LP Fair Pricing

We plan to allow collateralization of LP tokens, such as AVAX-ETH LP, in the future. In order to price LP tokens securely, we will use the Fair LP Pricing Algorithm[10] outlined by Alpha Finance. The price of the underlying assets will still be priced by *Chainlink* price feeds.

Comptroller

The *Compound* protocol introduced the *Comptroller*, a smart contract device that manages token markets available for lending and borrowing. This is the contract that borrowers, lenders and admin will interact with. The contract validates collateral and liquidity before a user may interact with the token market.

- E.g. Borrowing requires a valid price from the *Price Oracle*.
- E.g. Collateralization requires a valid price and collateral factor.
- E.g. Lending requires a valid interest rate model.
- E.g. Adding new markets

[9] Chainlink, https://chain.link/
[10] Alpha Finance, Fair LP Pricing Algorithm, https://blog.alphafinance.io/fair-lp-token-pricing/

Listing of Assets

Before any asset is listed to the lending protocol, it must be whitelisted. At launch, we will whitelist the following assets:

- WETH
- USDT
- DAI
- AVAX
- LINK
- WBTC

The absolute minimum requirement for an asset to be listed on the *Banker Joe* protocol is that there must exist a *Chainlink* price feed for it. At the start, the team will decide which assets to list, but this will be transitioned to governance once it goes live.

Leveraged Trading

We propose a modular approach to leveraged trading as a combination of lending, borrowing and trading. In our proposed user flow, the borrowing can occur at the time when the trader is making the token swap. Another way to view this synergy is that the leveraged trade generates demand for token borrowing, which in turn generates demand for token lending.

On centralized exchanges where the tokens are held in custody, leverage is defined with respect to the total collateral value of the entire user account. This doesn't apply for a decentralized, non-custodial system where the user assets are held in their own wallet. Instead, we define leverage based on spot trade itself.

$$Leverage = 1 + BorrowAmount / SpotAmount$$

E.g. Trader borrows 100 USDT in addition to spending 100 USDT to perform a swap. In this case the leverage is 2X.

Key benefits include:

- **Non-custody of swapped tokens**
    - The tokens that the trader intends to gain leveraged exposure to end up in the trader's wallet.
- **Cross-margining of lending assets**
    - Lending and borrowing of a token should be from a single pool.
- **Modularized lending**
    - Both lender and protocol are exposed only to the loan risk in the borrowing flow. The subsequent trades don't affect lenders.

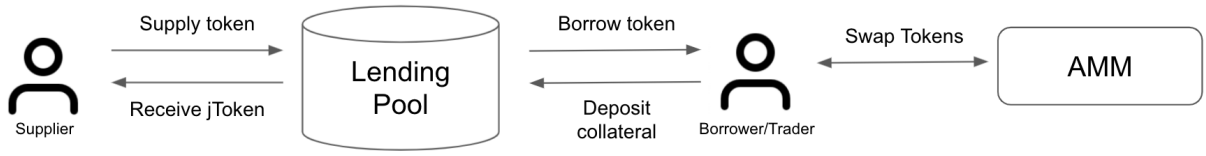

Fig 1. High-level user flow

Implementation

At its core, the leveraged trading feature will be implemented by a *LeverageTradeRouter* contract. This contract encapsulates the logic that will call the *Comptroller* contract to borrow the desired asset and call the *JoeRouter* contract to make the swap.

Version 1 of our leveraged trading feature will allow borrowing up to the user's available collateral value. Future versions may enable higher leverage trading, such as by offering a high risk lending pool.

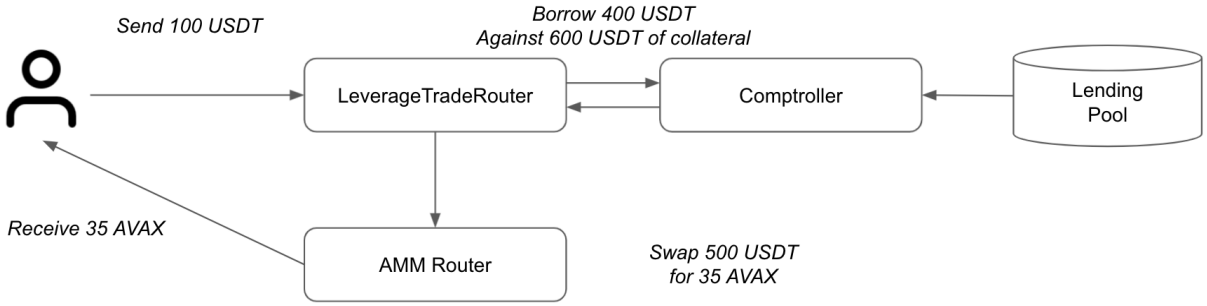

Fig 2. LeverageTradeRouter flow

Leveraged Long 2x Example

In this example, a trader wants to buy AVAX against USDT, with 2X leverage.

- The current price of AVAX is 100 USDT.
- Trader has 500 USDT in his wallet, and $1000 collateral value deposited.
    - Collateral factor: 75%
    - Borrow limit: $650
- Trader will buy 10 AVAX by sending 500 USDT from his wallet and borrowing 500 USDT from the protocol (he has $1000 of collateral value).
    - Leverage = 1 + BorrowedAmount / SpotAmount = 2X
    - Borrow balance: $500
    - Borrow limit used: 77%
- Trader has liquidation risk from borrowing 500 USDT (pls see *Liquidation Example* above for details).

Now we imagine some scenarios:

1. AVAX increases from 100 to 110 USDT
    a. He swaps 10 AVAX to 1100 USDT
    b. Pays back the borrowed balance of 500 USDT
    c. Profit is 600 USDT - 500 USDT = 100 USDT
2. AVAX decreases from 100 to 90 USDT
    a. He swaps 10 AVAX to 900 USDT
    b. Pays back the borrowed balance of 500 USDT
    c. Loss is 400 USDT - 500 USDT = -100 USDT

Summary

- We introduce *Banker Joe*, a decentralized lending protocol. Its approach follows closely to the *CREAM* and *Compound* protocols, with support for long-tail assets.
    - The whitelisted assets at launch are: WETH, USDT, DAI, AVAX, LINK, WBTC.
    - This will be expanded to long-tail assets as well as LP tokens in the future.
    - Protocol fees may be paid to xJOE holders, and will be managed by governance.
- Leveraged trading will be implemented as a combination of DEX token swapping and token borrowing.

Reference

1. Leshner & Hayes, Compound: The money market protocol, 2019
2. Aave Protocol Whitepaper, Jan 2020
3. FTX spot margin lending, https://ftx.com/intl/spot-margin/lending
4. Sushiswap, https://docs.sushi.com/products/kashi-lending
5. WOWswap, https://docs.wowswap.io/about/wowprotocol
6. Impermax, https://www.impermax.finance/
7. CREAM finance, https://docs.cream.finance/
8. CREAM finance, interest rate model. https://docs.cream.finance/lending/interest-rate-model
9. Chainlink, https://chain.link/
10. Alpha Finance, Fair LP Pricing Algorithm, https://blog.alphafinance.io/fair-lp-token-pricing/