Timeswap V2 - Time Preference Protocol

Designed by Ricsson Ngo

Timeswap is a fixed time preference protocol for users to manage their ERC20 tokens over a fixed period. It works as a zero liquidation money market and options market in one. Users can lend tokens into the pool to earn a fixed income. They can also borrow or leverage tokens against other tokens, without any liquidation risk. Liquidity providers (different from lenders) create markets for any pair of tokens, adding liquidity, and acting as the counterparty to both lenders and borrowers of the protocol. In return, they earn transaction fees from both sides of the market. Timeswap utilizes a unique constant sum options specification and an ingenious duration weighted constant product automated market maker (AMM) similar to the Uniswap AMM. It is designed to work without oracles, is capital efficient, permissionless to use, game theoretically sound in any state of the market, and is easy to use. It becomes the fundamental time preference primitive lego to build exotic and interesting DeFi products that need fixed period preference.

Fundamentally, any non-liquidation fixed term lending and borrowing is an option implementation under the hood. Timeswap utilises a proprietary mechanism to value its options.

The protocol follows a unique AMM formula for each pool, where a pool has a pair, strike, and maturity. We will use ETH/USD pair, K strike, and 1 year maturity as an example.

$$(x + y)z = l$$

where x is the number of Long ETH options in the pool, y is the number of Long USD options in the pool, and z is the number of Short options (ETH or USD) per second in the pool (per second to account for pool duration).

Explanation of the AMM:

Let us consider $x + y$ which corresponds to Timeswap Constant Sum Option where, single option has locked tokens in it that follows a constant sum formula.

$$x_1 + y_1 = 1$$

where x_1 is the number of ETH locked, and y_1 is the number of USD locked divided by strike K. Therefore, a single option may have either 1 ETH locked or K USD locked in it. Note that this option can be fractionalized.

The constant sum option gives the holder the right but not the obligation to withdraw and deposit any of the tokens from and to it, for a fixed duration, as long as the resulting locked tokens follow the constant sum formula.

Let us define a constant sum option with 1 ETH locked as 1 Long ETH, a long call position. We also define a constant sum option with K USD locked as 1 Long USD, a long put position. The constant sum mechanism gives the owner the ability to transform Long ETH to and from Long USD at any time, following the principles of modified Put-Call Parity (removing the present value calculation of strike).

$$Long\ ETH\ +\ Strike\ =\ Long\ USD\ +\ Spot$$

Finally we define Short as the counterparty position who receives ETH or USD left in the option at maturity. Both Short put and Short call are the same in this protocol.

To create these positions, any contract can mint 1 Long ETH and 1 Short, when 1 ETH is locked. Any contract can also mint 1 Long USD and 1 Short, when K USD is locked. Burning these positions goes the opposite way, any contract can withdraw 1 ETH by burning 1 Long ETH and 1 Short. Any contract can withdraw K USD by burning 1 Long USD and 1 Short. These can only be done before maturity.

With the understanding of constant sum option, we then have $x\ +\ y$ where x is the amount of Long ETH, and y is the amount of Long USD in the pool.

Due to the constant sum formula, whenever Long ETH in the pool is in the money, arbitrageurs will withdraw the Long ETH from the pool, transform it to Long USD for profit, and deposit the Long USD back into the pool. Similarly as well for when Long USD is in the money. Therefore it is expected that only either Long ETH or Long USD exists in the pool at all times. This implies that either $x\ =\ 0$ or $y\ =\ 0$.

Therefore we can simplify the AMM to the following

$$xz\ =\ l\ (y\ =\ 0 \text{ i.e. when pool has only long ETH})$$

or

$$yz\ =\ l\ (x\ =\ 0 \text{ i.e. when pool has only long USD})$$

These two formulas follow a duration weighted constant product formula, where $x\ +\ y$ will be the amount of Long and where z is the amount of Short per second in the pool (z times duration is the amount of Short in the pool).

Since it also follows the constant product formula as shown in the equations above, similar to Uniswap, the marginal price ratio between Short and the out-of-the-money Long is from $[0, \infty)$. It is designed this way, because if the pricing is between Short and the in-the-money Long, the pricing of $[0, \infty)$, may lead the out-of-the-money Long to have negative pricing, which is bad pricing that leads to inefficient AMM slippage.

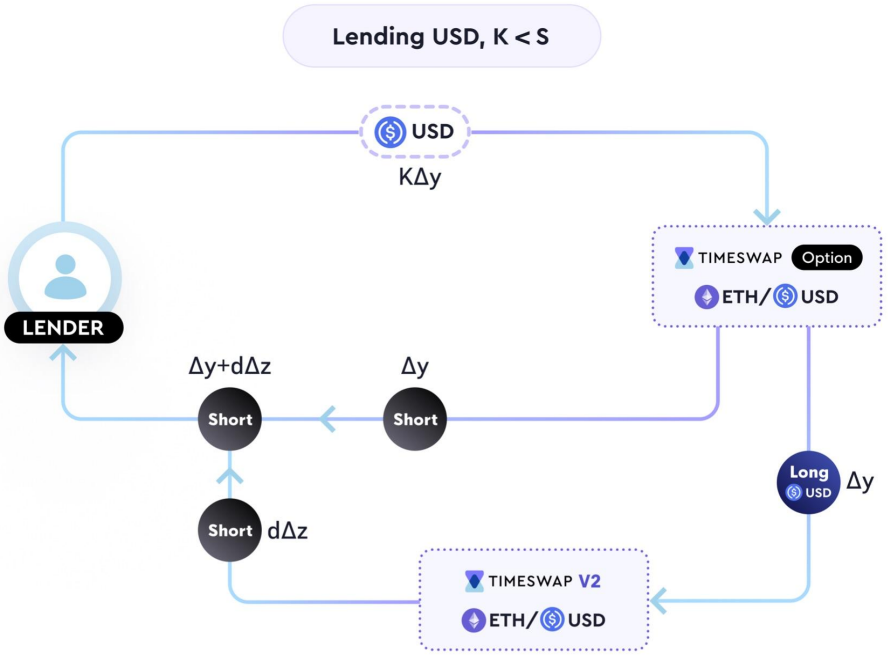
$$Short\ +\ Long\ ETH\ =\ ETH$$

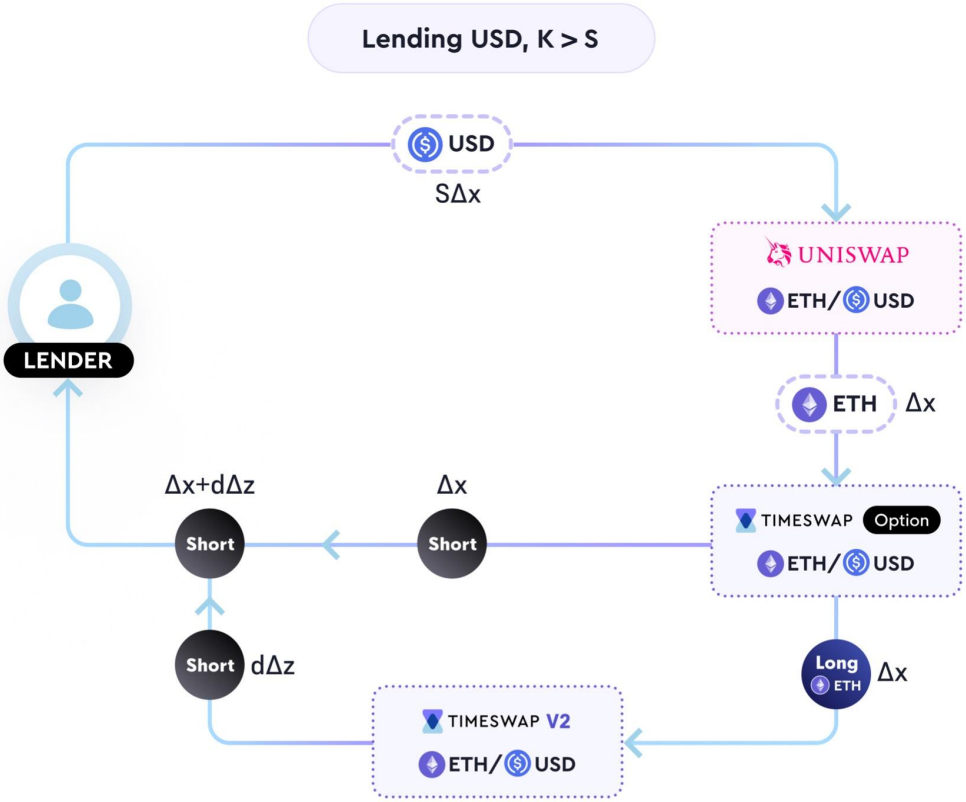
$$Short + Long\ USD = K{\cdot}USD$$

We only need to price Short and the out-of-the-money Long, since we can price the in-the-money Long from the out-of-the-money Long through the Put-Call-Parity transformation. So even if only out-of-the-money Long is in the pool. Any contract that wants in-the-money Long, can get the out-of-the-money Long and transform it. This makes a Timeswap pool doubly more capital efficient, as it can serve both fully collateralized call and put options.

Another capital efficiency improvement is that instead of pricing the option against a base token like USD, we price Long versus Short instead. Since Long and Short are minted from the same base token, the amount of capital required is much less for the same liquidity. Also by pricing Long versus Short, pricing slippage also becomes more efficient.

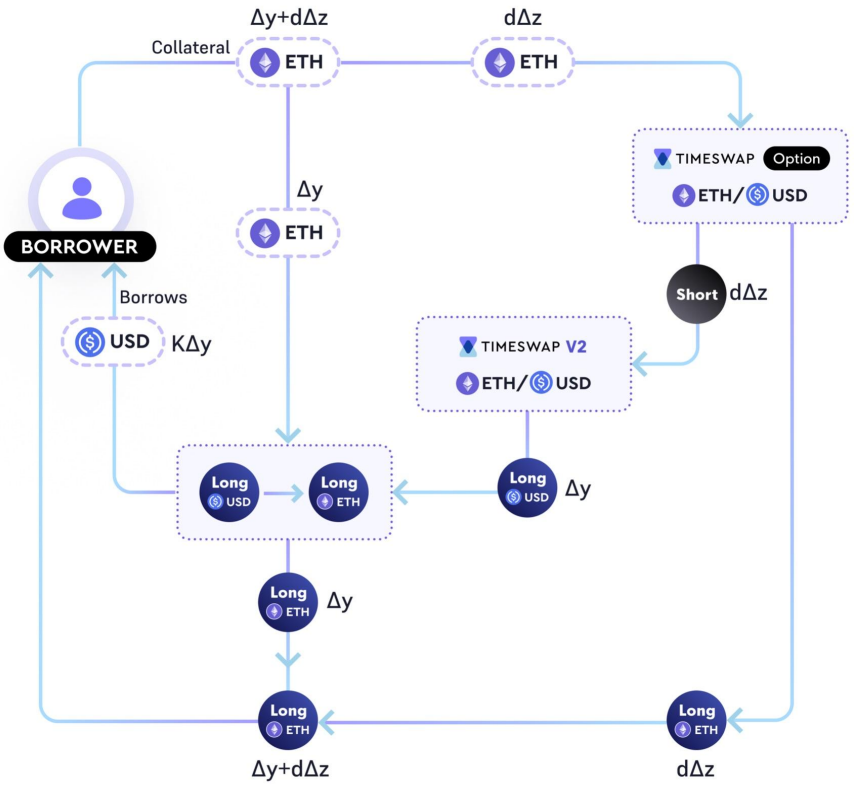
Lenders/option sellers deposit ETH or USD into Timeswap and receive a Short position. Under the hood, the contract first mints Short and out-of-the-money Long (It will swap ETH for USD and USD for ETH through Uniswap whenever necessary to get the tokens required to mint the out-of-the-money Long). Then out of the money Long will be deposited into the Timeswap pool to withdraw Short. The Short minted and the Short withdrawn from the pool go to the lenders/option sellers. From the two diagrams below, $K\Delta y$ and $S\Delta x$ are the amount of USD deposited by lenders. $d\Delta z$ is the amount of Short withdrawn from the Timeswap V2 pool.



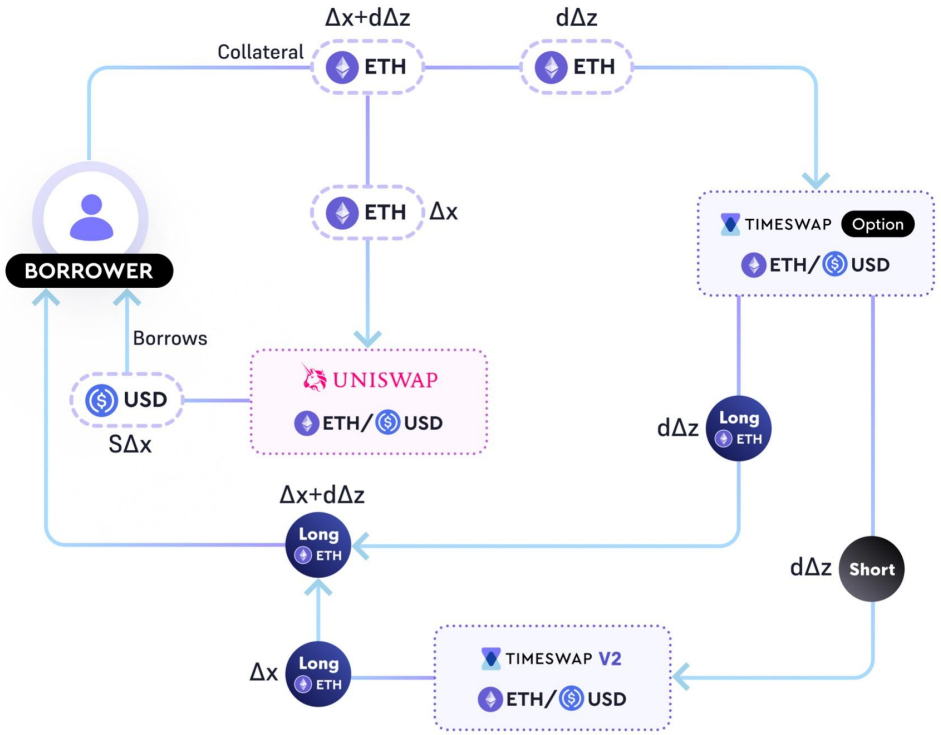


Borrowers deposit ETH or USD as collateral, receive USD or ETH respectively as principal, and obtain the desired Long position. Under the hood, the contract first mints Short and the desired Long. Then, the Short will be deposited into the Timeswap pool to withdraw the out-of-the-money Long. If the borrowers desire in-the-money Long, the out-of-the-money Long withdrawn will be transformed to the in-the-money Long, which leads to borrowers receiving USD and ETH as principal, and depositing more ETH or USD respectively. If the borrowers desire the out-of-the-money Long, then the contract will simply swap some ETH or USD collateral to USD or ETH respectively through Uniswap as the principal borrowed. The Long minted and the Long withdrawn (transformed if necessary) from the pool go to the borrower. The ETH or USD deposited to mint and the ETH or USD used to transform the Long or swapped through Uniswap is the total collateral locked by the borrower. The USD or ETH withdrawn from the transformed Long or received from Uniswap is the principal borrowed. Options buyers will follow similar steps as the borrower, with some configured steps. From the two diagrams below, $\Delta y + d\Delta z$ and $\Delta x + d\Delta z$ are the amount of ETH deposited as collateral. $K\Delta y$ and $s\Delta x$ are the amount of USD borrowed.

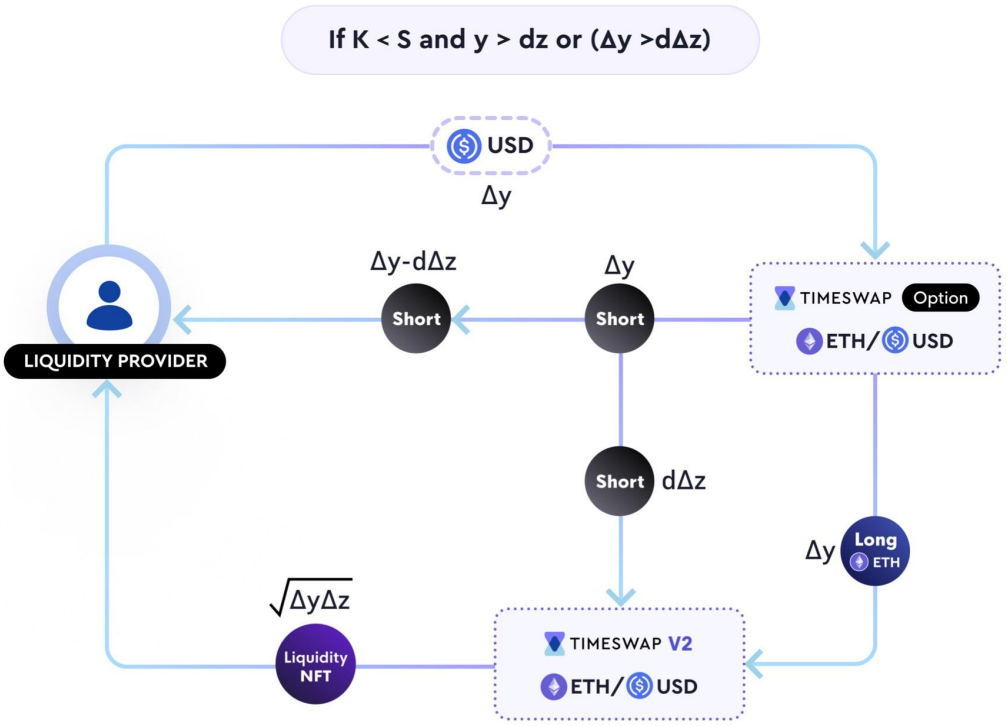
Borrowing USD for Long ETH, K < S

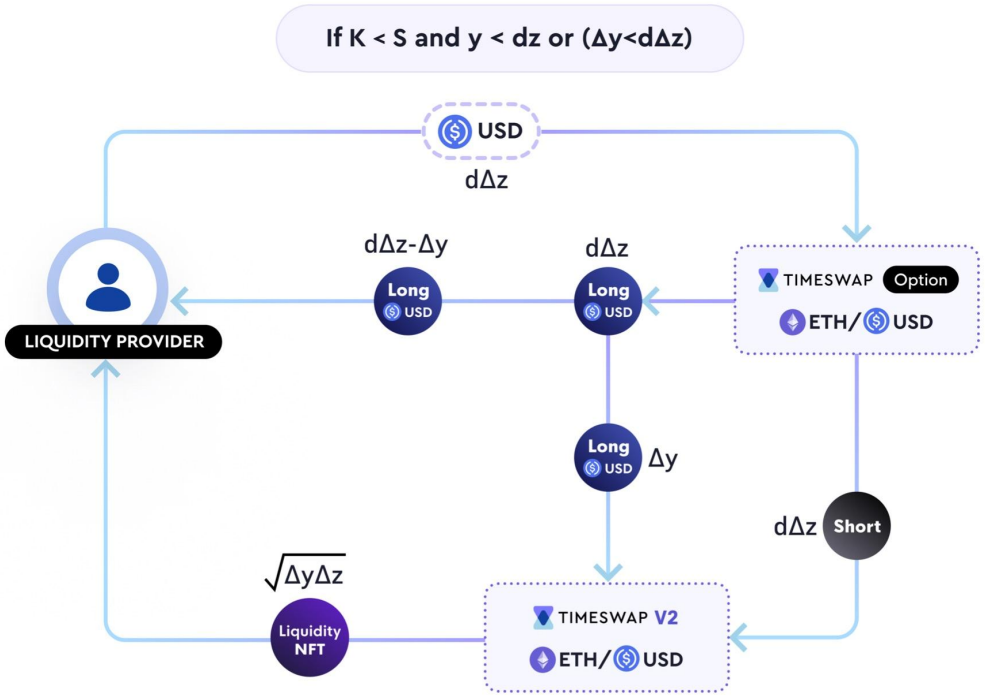


Borrowing USD for Long ETH, K > S



Liquidity providers deposit either ETH or USD to mint Short and the out-of-the-money Long, and deposit some ratio of both into the pool. This means that the Liquidity provider will have some excess Short or Long in their balance (which they could hold, or sell back into the pool). Liquidity providers earn Short or out-of-the-money Long whenever users transact with the pool. The goal of the Liquidity provider is to earn enough fees to cover possible divergent loss, similar to how Uniswap liquidity providers work. The capital efficiency of adding liquidity is maximized, as only ETH or USD is added as liquidity, whichever mints the out-of-the-money Long. The constant sum option arbitrage mechanism will always change the collateral required in the pool for whatever market state we are in. Note that concentrated liquidity may be added into the constant product for greater capital efficiency and price control. From the two diagrams below, Δy and $d\Delta z$ are the amount of USD deposited.





More in-depth information will be available in the Whitepaper.