UniBoost: A Trustless and Permissionless Liquidity Mining Tool for Uniswap v3

Venson Liou, John Guan, PC Chen, Guan-Ting Su

1 Introduction

UniBoost is a groundbreaking decentralized finance (DeFi) application that offers a trustless and permissionless liquidity mining tool for all Uniswap v3 pools. The application has two primary roles: the project owner and the user.

1.1 Roles

The project owner's primary objective is to enhance the liquidity of a specific token, referred to as "A," which is paired with a supported token "S." Supported tokens include BTC, ETH, USDC, USDT, DAI, and FRAX. Users, on the other hand, provide liquidity on Uniswap v3.

To increase liquidity, the project owner offers a reward distributed among liquidity providers who stake their Uniswap v3 liquidity position Non-Fungible Tokens (NFTs) on the project. The distributed rewards are proportional to the trading fee earned for each position, ensuring that only effective positions receive the reward.

UniBoost also provides insurance to protect users who provide liquidity. The project owner locks in an amount of the supported token "S" as insurance for the user. This insurance protects users from the risk of losing all their funds if the price of token "A" crashes. If the price falls below a predetermined threshold, the project will be closed, and the insurance will be distributed to all users.

UniBoost's unique features, such as boosting liquidity and providing insurance, encourage users to provide liquidity to token "A" while minimizing the risk of loss.

2 Project Setup

To initiate a project, anyone can use UniBoost to boost a liquidity pool on Uniswap v3. Uniswap pools contain two tokens, and at least one of the tokens must be a supported token: BTC, ETH, USDC, USDT, DAI, and FRAX, referred to as "S." The other token is the one the project owner wants to boost, referred to as "A." There are six parameters required to start a project:

1. Pool address: The Uniswap v3 pool to be boosted.
2. Boost factor: The incentive will be distributed to users according to their token A trading fee multiplied by the boost factor.
3. Close time: The time when the project ends. If liquidation does not occur, the insurance will be returned to the project owner.
4. Insurance amount: The amount of insurance to be locked into the insurance vault. Note that the insurance token must be the supported token S.
5. Incentive amount: The amount of incentive to be added to the incentive vault.
6. Liquidation price: The project will be liquidated when the pool price falls below the liquidation price.

3 User Guide

To use UniBoost as a user, follow these steps:

1. Provide liquidity on Uniswap v3.
2. Stake the position NFT into the project corresponding to the vault. Note that there will be a minimum time to unstake the position NFT.
3. Whenever your position earns a fee, you can claim the fee and incentive through the "claim reward" button.
4. When the project is liquidated, you can get the insurance share by pushing the "claim insurance" button.

As a project owner, you can start a project and add incentive to the incentive vault anytime. When the project is complete, the owner can claim the remaining incentive and insurance.

4 Distribution of the Incentive

When a user stakes their Uniswap v3 position NFT into UniBoost, UniBoost initially collects the swap fee. Consequently, all positions in UniBoost start with zero swap fee. When someone swaps token A into token S on Uniswap v3, these positions may earn a swap fee fee_A in token A. The user can then claim the reward with fee_A multiplied by the Boost factor. We only count on fee_A because most of the risk originates from token A, not the other supported token S, and this simple calculation has an implementation advantage, making it possible to handle on a smart contract.

Though the design is simple, there is a scenario in which the incentive vault is empty, and the user cannot claim the reward anymore. However, the project owner can add incentives to the incentive vault anytime. If the project owner wants to keep promoting this pool, they should add incentives.

5 Distribution of the Insurance

When the project is liquidated, the insurance will be distributed to the users by the following rule:

$$I_k = I \times \frac{w_k}{W},\ w_k = s_k \times (t - t_k) \tag{1}$$

where I is the total insurance amount; I_k is the insurance distributed to position k; s_k is the amount of token S when position k is staked in the project; t_k is the timestamp when the position k is staked; t is the timestamp when the project is liquidated; and $W = \sum_k W_k$. w_k represents the risk that position k takes when staking liquidity. We only count on the token S amount because S is the supported valuable token. When providing token S into the liquidity pool, the user suffers the risk of S being swapped into token A. In addition, this setup also encourages the user to provide liquidity that includes the current pool price, not just providing liquidity with 100% token A and under the current pool price. Moreover, the weight is multiplied by the time remaining until the end of the project, which incentivizes the user to provide liquidity early.

The calculation is also feasible on-chain. We use a trick to avoid recording every liquidity position's staking time by updating the total amount S_k each time a new position is staked. Let $W^{(k)}$, $t^{(k)}$ denote the total weight

and timestamp when the k-th position is staked, respectively.

$$W^{(k)} = \sum_{i=1}^{k} s_i \times (t^{(k)} - t_i), \tag{2}$$

$$W^{(k+1)} = \sum_{i=1}^{k} s_i \times (t^{(k+1)} - t_i) + s_{i+1} \times (t^{(k+1)} - t^{(k)}) \tag{3}$$

$$= \sum_{i=1}^{k} s_i \times (t^{(k+1)} - t^{(k)} + t^{(k)} - t_i) + s_{i+1} \times (t^{(k+1)} - t^{(k)}) \tag{4}$$

$$= W^{(k)} + \sum_{i=1}^{k+1} s_i \times (t^{(k+1)} - t^{(k)}) \tag{5}$$

Hence we can calculate the weights by recording $W^{(k)}$ and $\sum_{i=1}^{k} s_i$.