Wombat Cross Chain AMM

Wombat Team

Wombat Exchange

Abstract. This paper describes how we expand Wombat's algorithm to accommodate cross-chain swaps. Using Wombat's innovative open pool liquidity design and algorithm and Wormhole's cross-chain messaging system, the cross-chain pools can break down barriers and enable capital-efficient swapping between two chains.

1 Re-Introduction

Observing Wombat's success in displaying capital and slippage efficiency, it is natural to ask whether it is possible to have *token x* on *chain A* and *token y* on *chain B* and how users could swap between them. This paper proposes a method for swapping assets between different chains.

Wombat's foundational algorithm for the swaps has remained largely unchanged. To keep the robustness, this is the obvious choice. The following paper will be a quick overview. Refer to *Wombat's Whitepaper* and the *Wombat Side Pool and Dynamic Whitepaper* to better understand Wombat's mechanics.

Recalling the Wombat invariant for a single chain:

$$\sum_i L_i(r_i - \frac{A}{r_i}) = D$$

Wombat's design has the following assumptions:

- Maintaining our global equilibrium coverage ratio, i.e., r* =1;
- Wombat's stableswap invariant design does not require asset price input, i.e., oracle;
- Assumption of pegged assets priced at 1;
- Assumption of pegged assets reverting to their pegged prices over time.

Invariant in the Context of Single-Chain

To facilitate cross chain transactions, we modify our stableswap invariant with an additional term, *Credit*, which serves as a network-neutral representation of value. *Credit* is equivalent to the value of 1 token in the pool when the coverage ratio of the token is 100%. In other words, 1 credit can swap into 1 token when its coverage ratio is 100%, vice versa. Users will only obtain credit when the transaction of the designated chain is not executed. Let's define the cross-chain pool invariant for a *single chain* as follows:

$$\sum_i L_i(r_i - \frac{A}{r_i}) = D + (1+A)C \tag{1}$$

C denotes the amount of outstanding credit, which represents the net amount of unrealized credit for that network. In a scenario where all credits are settled (swapped into tokens), C equals zero and it is identical with our stableswap invariant. We can justify the coefficient term, $(1 + A)$, for credit by considering the marginal slippage when swapping credit to token x. Based on our cross-chain AMM invariant:

$$L_x(\frac{A_x}{L_x} - \frac{A}{\frac{A_x}{L_x}}) + \sum_{i \neq x} L_i(\frac{A_i}{L_i} - \frac{A}{\frac{A_i}{L_i}}) = D + (1+A)C$$

A_x and L_x represent the asset and liability of token x. A denotes the amplification factor. We can observe the marginal slippage through the following:

$$\begin{aligned}
\frac{dA_x}{dC} + \frac{AL_x^2}{A_x^2}\frac{dA_x}{dC} + \sum_{i \neq x}(\frac{dA_i}{dC} + \frac{AL_i^2}{A_i^2}\frac{dA_i}{dC}) &= \frac{dD}{dC} + (1+A)\frac{dC}{dC} \\
\frac{dA_x}{dC}(1 + \frac{A}{r_x^2}) &= (1+A) \\
\frac{dA_x}{dC} &= \frac{(1+A)}{(1+\frac{A}{r_x^2})}
\end{aligned} \tag{2}$$

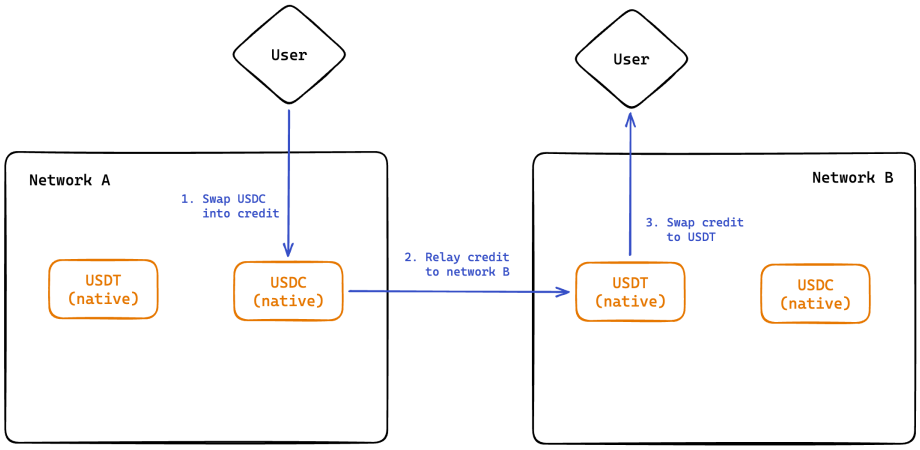
As such, with the coefficient term, (1 + A), a credit can be swapped for (or from) a token with a coverage ratio of 100%.

2 Invariant in the Context of Cross-Chain

$$\sum_{n}^{\text{Network}} [\sum_{i}^{\text{Token}} L_{ni}(r_{ni} - \frac{A_n}{r_{ni}})] = \sum_{n}^{\text{Network}} D_n + (1 + A_n)C_n \tag{3}$$

Users can swap between native assets on different chains using a cross-chain pool, as indicated by equation (3). In a multi-networks setting, we allow for network-specific amplification factors, A_n, and Credit, C_n. Additionally, users can obtain gas tokens on the designated network by sending more gas from the source network, supported by the Wormhole relayer.

If a user wants to swap token x from Network A to token B on Network B, we can understand the process in three steps. First, the user swaps token x to credit by putting a certain amount of token x into the pool on chain A, causing an increase in RHS, $[D_A + (1 + A_A)C_A]$, by the amount C. Second, credit is relayed from Network A to Network B with an identical amount of credit. Finally, the user swaps credit to token and get token y on Network B, causing a decrease in RHS by the amount C.



3 Wormhole's Messaging Layer

Wormhole is used as the message layer to relay messages in cross-chain transactions. The *WormholeAdaptor* contract serves as an adapter between the pool and the Wormhole Relayer, publishing messages to the Wormhole Network. It also completes the swap at the designated chain when it receives the signed message, also known as Verified Action Approvals (VAA).

4 Credit, Mitigating MEV, and Slippage

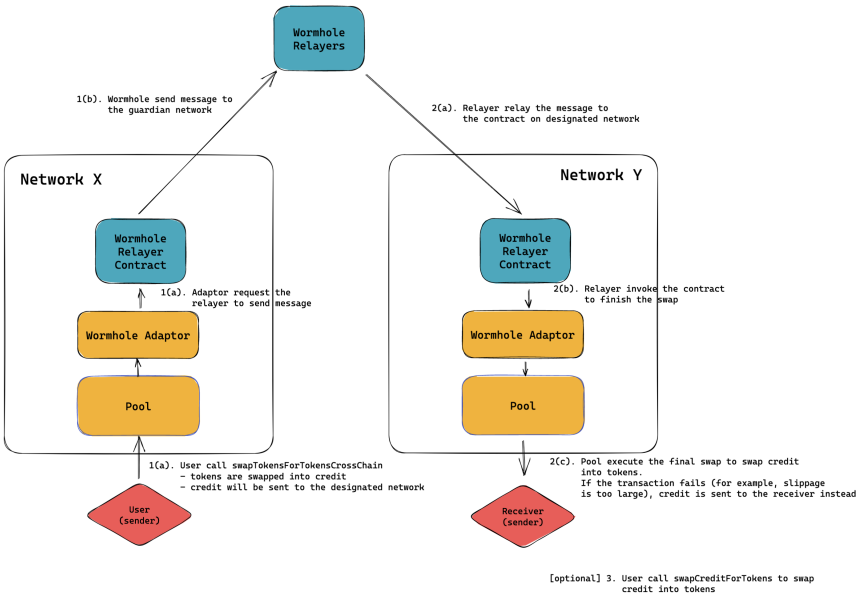
As the transaction between the source chain and the designated chain is asynchronous, the final received amount is not deterministic when the transaction at the source chain is executed. To address this issue, we introduce the concept of "credit" – an imaginary intermediate token. The value of 1 credit is equivalent to 1 token when the token's coverage ratio is 100%.

The transaction in the source chain swap fromToken into credit. The amount of credit is then relayed to the designated chain, where it is swapped into the toToken. If no intermediate transactions are performed on the toToken, the final amount received will equal that of a same-chain swap.

Users can specify a minimum receive amount for the first and second steps. If the slippage in the first step exceeds the specified amount, the transaction will be reverted to the source chain. However, if the slippage at the second step exceeds the specified amount, the user will receive credit instead. Users can swap credit for native tokens on the designated network or bridge the credit back to the source chain and swap it for native tokens in the source network.

5 Steps of a Cross-Chain Swap

1. To initiate a cross-chain swap, the sender will call *CrossChainPool.swapTokensForTokensCrossChain*.
    - The Wormhole Relayer requires extra gas tokens to relay the message to the contract on the designated chain. These tokens should be attached in the value field of the message.
    - To determine the amount of value required, use *WormholeAdaptor.estimateDeliveryFee*.
2. The Guardian Network observes and signs messages.
3. Wormhole Relayers are responsible for relaying the signed message (the VAA) to the designated chain. Once complete, they will invoke the pool to swap tokens and send them to the receiver.
4. Users will receive credit if the slippage is too significant and the swap is reverted on the designated chain. Users can then exchange their credit for native tokens.



6 Security Mechanisms

6.1 Operational

1. A high coverage ratio fee is charged to prevent one token from being dumped into the pool.
2. To prevent too many tokens from being bridged from or to one chain, Set *reasonable values* for *maximumOutboundCredit / maximumInboundCredit*
3. To pause a chain from cross-chain swap immediately, Wombat can set *swapCreditForTokensEnabled / swapTokensForCreditEnabled*
4. Limits of the pool size are considered for each environment's liquidity availability.

6.2 Smart Contract

1. To prevent message replay, *deliveredMessage* is recorded.
2. *WormholeAdaptor* specifies one designated chain to deliver messages in order to prevent message replay across different chains. On the designated network, the *WormholeAdaptor* verifies that a trusted contract in the source network emits the message.

Reference

Wombat Team. (2022). Wombat - An Efficient Stableswap Algorithm.

Wombat Team. (2022). Wombat Side Pool and Dynamic Pool.