Loopring:
A Decentralized Token Exchange Protocol

Daniel Wang
daniel@loopring.org

Jay Zhou
jay@loopring.org

Alex Wang
alex@loopring.org

Matthew Finestone
matt.finestone@gmail.com

https://loopring.org

September 8, 2018

Abstract

Loopring is an open protocol for building decentralized exchanges. Loopring operates as a public set of smart contracts responsible for trade and settlement, with an off-chain group of actors aggregating and communicating orders. The protocol is free, extensible, and serves as a standardized building block for decentralized applications (dApps) that incorporate exchange functionality. Its interoperable standards facilitate trustless, anonymous trading. An important improvement over current decentralized exchange protocols is the ability for orders to be mix-and-matched with other, dissimilar orders, obviating the constraints of two-token trading pairs and drastically improving liquidity. Loopring also employs a unique and robust solution to prevent front-running: the unfair attempt to submit transactions into a block quicker than the original solution provider. Loopring is blockchain agnostic, and deployable on any blockchain with smart contract functionality. At the time of writing, it's operable on Ethereum [1] [2] and Qtum [3] with NEO [4] under construction.

1 Introduction

With the proliferation of blockchain-based assets, the need to exchange these assets amongst counterparties has significantly increased. As thousands of new tokens are introduced - including the tokenization of traditional assets - this need is magnified. Whether exchanging tokens for speculative trading motivations, or converting to access networks via their native utility tokens, the ability to exchange one cryptoasset for another is foundational for the larger ecosystem. Indeed, there is a potential energy in assets [5], and realizing this energy - unlocking capital - requires not only asserting ownership, which blockchains have immutably allowed for, but the ability to freely transfer and transform these assets.

As such, the trustless exchange of tokens (value) is a compelling use case for blockchain technology. Until now, however, crypto enthusiasts have largely settled for trading tokens on traditional centralized exchanges. The Loopring protocol is needed because, just as Bitcoin [6] dutifully emphasized that, in regards to peer-to-peer electronic cash, "the main benefits are lost if a trusted third party is still required to prevent double-spending", so too are the main benefits of decentralized assets lost if they must pass through trusted, gated, centralized exchanges.

Trading decentralized tokens on centralized exchanges doesn't make sense from a philosophical perspective, as it fails to uphold the virtues these decentralized projects espouse. There are also numerous practical risks and limitations in using centralized exchanges which are described below. Decentralized exchanges (DEXs) [7] [8] [9] have sought to address these issues, and in many cases have succeeded in alleviating security risks by using blockchains for disintermediation. However, as DEX capability becomes crucial infrastructure for the new economy, there is substantial room for performance improvement. Loopring aims to provide modular tools for said infrastructure with its dApp agnostic open protocol.

2 Current Exchange Landscape

2.1 Inadequacies of Centralized Exchanges

The three primary risks of centralized exchanges are; 1) Lack of security, 2) Lack of transparency, and 3) Lack of liquidity.

Lack of Security arises from users typically surrendering control of their private-keys (funds) to one centralized entity. This exposes users to the possibility that centralized exchanges fall prey to malicious hackers. The security and hacking risks facing all centralized exchanges are well known [10] [11], yet are often accepted as "table stakes" for token trading. Centralized exchanges continue to be honeypots for

hackers to attack as their servers have custody over millions of dollars of user funds. Exchange developers can also make honest, accidental errors with user funds. Simply, users are not in control of their own tokens when deposited at a centralized exchange.

Lack of Transparency exposes users to the risk of dishonest exchanges acting unfairly. The distinction here is by the exchange operator's malicious intentions, as users are not truly trading their own assets on centralized exchanges, but rather, an IOU. When tokens are sent to the exchange's wallet, the exchange takes custody, and offers an IOU in its place. All trades are then effectively between users' IOUs. To withdraw, users redeem their IOU with the exchange, and receive their tokens to their external wallet address. Throughout this process there is a lack of transparency, and the exchange can shutdown, freeze your account, go bankrupt, etc. It is also possible that they use user assets for other purposes while in custody, such as lending them out to third parties. Lack of transparency can cost users without a total loss of funds, such as in higher trading fees, delays at peak demand, regulatory risk, and orders being front ran.

Lack of Liquidity. From the point of view of exchange operators, fragmented liquidity inhibits entry by new exchanges because of two winner-takes-all scenarios. First, the exchange with the greatest number of trading pairs wins, because users find it desirable to conduct all their trades on one exchange. Second, the exchange with the largest order book wins, because of favorable bid-ask spreads for each trading pair. This discourages competition from newcomers because it is difficult for them to build up initial liquidity. As a result, many exchanges command a high market share despite user complaints and even major hacking incidents. It's worth noting that as centralized exchanges win market share, they become an ever-larger hacking target.

From the point of view of users, fragmented liquidity significantly reduces user experience. In a centralized exchange, users are only able to trade within the exchange's own liquidity pools, against its own order book, and between its supported token pairs. To trade token `A` for token `B`, users must go to an exchange that supports both tokens or register at different exchanges, disclosing personal information. Users often need to execute preliminary or intermediate trades, typically against BTC or ETH, paying bid-ask spreads in the process. Finally, the order books may not be deep enough to complete the trade without material slippage. Even if the exchange purports to process large volumes, there is no guarantee that this volume and liquidity is not fake [12].

The result is disconnected silos of liquidity and a fragmented ecosystem that resembles the legacy financial system, with significant trading volume centralized on few exchanges. The global liquidity promises of blockchains hold no merit within centralized exchanges.

2.2 Inadequacies of Decentralized Exchanges

Decentralized exchanges differ from centralized exchanges in part because users maintain control of their private-keys (assets) by performing trades directly on the underlying blockchain. By leveraging the trustless technology of cryptocurrencies themselves, they successfully mitigate many of the abovementioned risks surrounding security. However, problems persist in regards to performance and structural limitations.

Liquidity often remains an issue as users must search for counterparties across disparate liquidity pools and standards. Fragmented liquidity effects are present if DEXs or dApps at large don't employ consistent standards to interoperate, and if orders are not shared/propagated across a wide network. The liquidity of limit order books, and, specifically, their resiliency – how fast filled limit orders are regenerated – can significantly affect optimal trading strategies [13]. The absence of such standards has resulted not only in reduced liquidity, but also exposure to an array of potentially insecure proprietary smart contracts.

Furthermore, since trades are performed on chain, DEXs inherit the limitations of the underlying blockchain, namely: scalability, delays in execution (mining), and costly modifications to orders. Thus, blockchain order books do not scale particularly well, as executing code on the blockchain incurs a cost (gas), making multiple order-cancellation cadences prohibitively expensive.

Finally, because blockchain order books are public, the transaction to place an order is visible by miners as it awaits being mined into the next block and placed into an order book. This delay exposes the user to the risk of being front run and having the price or execution move against him.

2.3 Hybrid Solutions

For the above reasons, purely blockchain-based exchanges have limitations that make them uncompetitive with centralized exchanges. There is a tradeoff between on-chain inherent trustlessness, and centralized exchange speed and order flexibility. Protocols such as Loopring and 0x [14] extend a solution of on-chain settlement with off-chain order management. These solutions revolve around open smart contracts, but navigate scalability limitations by performing several functions off-chain and giving nodes flexibility in fulfilling critical roles for the network. However, drawbacks remain for the hybrid model as well [15]. The Loopring protocol proposes meaningful differences in our approach to a hybrid solution throughout this paper.

3 Loopring Protocol

Loopring is not a DEX, but a modular protocol for building DEXs on multiple blockchains. We disassemble the component parts of a traditional exchange and offer a set of public

smart contracts and decentralized actors in its place. The roles in the network include wallets, relays, liquidity-sharing consortium blockchains, order book browsers, Ring-Miners, and asset tokenization services. Before defining each, we should first understand Loopring orders.

3.1 Order Ring

Loopring orders are expressed in what we call a Unidirectional Order Model (UDOM)[16]. UDOM expresses orders as token exchange requests, `amountS/amountB`, (amount to sell/buy) instead of bids and asks. Since every order is just an exchange rate between two tokens, a powerful feature of the protocol is the mixing and matching of multiple orders in circular trade. By using up to 16 orders instead of a single trading pair, there is a dramatic increase in liquidity and potential for price improvement.

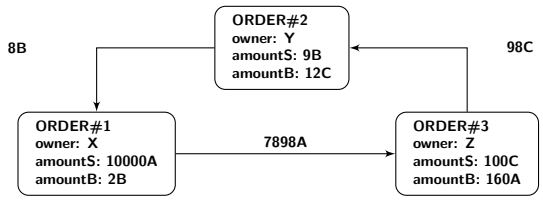

Figure 1: An order-ring of 3 orders

The above figure shows an order-ring of 3 orders. Each order's token to sell (`tokenS`) is another order's token to buy (`tokenB`). It creates a loop that allows each order to exchange their desired tokens without requiring an opposing order for its pair. Traditional order pair trades can, of course, still be executed, in what is essentially a special case of an order-ring.

Definition 3.1 (order-ring) *Let C_0, C_1, $\cdots$, C_{n-1} be n different tokens, $O_{0\to1}$, $\cdots$, $O_{i\to i\oplus1}$, $\cdots$, $O_{n-1\to0}$ be n orders. Those orders can form a order-ring for trading:*

$$O_{0\to1} \to \cdots \to O_{i\to i\oplus1} \to \cdots \to O_{n-1\to0},$$

where n is the length of the order-ring, and $i \oplus 1 \equiv i + 1 \mod n$.

An order-ring is valid when all component transactions can be executed at an exchange rate equal to or better than the original rate specified implicitly by the user. To verify order-ring validity, Loopring protocol smart contracts must receive order-rings from ring-miners where the product of the original exchange rates of all orders is equal to or greater than 1.

Let's assume Alice and Bob want to trade their token `A` and `B`. Alice has 15 token `A` and she wants 4 token `B` for them; Bob has 10 token `B` and he wants 30 token `A` for them.

Who is buying and who is selling? This depends only on the asset we fix to give price quotations. If token `A` is the reference, then Alice is buying token `B` for the price of $\frac{15}{4} = 3.75$`A`, while Bob is selling 10 token `B` for the price of $\frac{30}{10} = 3.00$`A`. In the case of fixing token `B` as reference, we say that Alice is selling 15 token `A` for the price of $\frac{4}{15} = 0.26666667$`B` and Bob is buying 10 token `A` for the price of $\frac{10}{30} = 0.33333334$`B`. Hence, who's the buyer or seller is arbitrary.

In the first situation Alice is willing to pay a higher price (3.75`A`) than the price Bob is selling his tokens for (3.00`A`), while in the second situation Bob is willing to pay a higher price (0.33333334`B`) than the price Alice is selling her tokens for (0.26666667`B`). It is clear that a trade is possible whenever the buyer is willing to pay an equal or higher price than the seller's price.

$$\frac{\frac{15}{4}}{\frac{30}{10}} = \frac{\frac{10}{30}}{\frac{4}{15}} = \frac{15}{4} \cdot \frac{10}{30} = 1.25 > 1 \tag{1}$$

Thus, for a set of n orders to be able to be filled, fully or partially, we need to know if the product of each one of the exchange rates as buy orders results in a number greater or equal to 1. If so, all the n orders can be either partially, or totally filled [17].

If we introduce a third counterparty, Charlie, such that Alice wants to give x_1 token `A` and receive y_1 token `B`, Bob wants to give x_2 token `B` and receive y_2 token `C`, and Charlie wants to give x_3 token `C` and receive y_3 token `A`. The necessary tokens are present, and the trade is possible if:

$$\frac{x_1 \cdot x_2 \cdot x_3}{y_1 \cdot y_2 \cdot y_3} \geq 1 \tag{2}$$

See section 7.1 for more details about Loopring's orders.

4 Ecosystem Participants

The following ecosystem participants jointly provide all functionalities a centralized exchange has to offer.

- **Wallets**: A common wallet service or interface that gives users access to their tokens and a way to send orders to the Loopring network. Wallets will be incentivized to produce orders by sharing fees with ring-miners (see section 8). With the belief that the future of trading will take place within the safety of individual user's wallets, connecting these liquidity pools through our protocol is paramount.
- **Consortium Liquidity Sharing Blockchain/Relay-Mesh**: A relay-mesh network for order & liquidity sharing. When nodes run Loopring relay software, they are able to join an existing network and share liquidity with other relays over a consortium blockchain. The consortium blockchain we are building as a first implementation has near real time order sharing (1-2 second blocks), and trims old history to allow for faster download by new nodes. Notably, relays need not join this consortium; they can act alone and not share liquidity with others, or, they can start and manage their own liquidity sharing network.

- **Relays/Ring-Miners**: Relays are nodes that receive orders from wallets or the relay-mesh, maintain public order books and trade history, and optionally broadcast orders to other relays (via any arbitrary off-chain medium) and/or relay-mesh nodes. Ring-mining is a feature – not a requirement – of relays. It is computationally heavy and is done completely off-chain. We call relays with the ring-mining feature turned on "Ring-Miners", who produce order-rings by stitching together disparate orders. Relays are free in (1) how they choose to communicate with one another, (2) how they build their order books, and (3) how they mine order-rings (mining algorithms).

- **Loopring Protocol Smart Contracts (LPSC)**: A set of public and free smart contracts that checks order-rings received from ring-miners, trustlessly settles and transfers tokens on behalf of users, incentivizes ring-miners and wallets with fees, and emits events. Relays/order browsers listen to these events to keep their order books and trade history up to date. See appendix **??** for details.

- **Asset Tokenization Services (ATS)**: A bridge between assets that cannot be directly traded on Loopring. They are centralized services run by trustworthy companies or organizations. Users deposit assets (real, fiat or tokens from other chains) and get tokens issued, which can be redeemed for the deposit in the future. Loopring is not a cross-chain exchange protocol (until a suitable solution exists), but ATS enable trading of ERC20 tokens [18] with physical assets as well as assets on other blockchains.

5 Exchange Process

1. **Protocol Authorization**: In figure 2, user `Y` who wants to exchange tokens authorizes the LPSC to handle `amountS` of token `B` the user wants to sell. This does not lock the user's tokens, who remains free to move them while the order is processed.

2. **Order Creation**: The current rate and order book for token `B` vs token `C`, are provided by relays or other agents hooked up to the network, such as order book browsers. User `Y` places an order (limit order) specifying `amountS` and `amountB` and other parameters through any integrated wallet interface. An amount of LRx can be added to the order as a fee for ring-miners; higher LRx fee means a better chance to be processed earlier by ring-miners. The order's hash is signed with user `Y`'s private-key.

3. **Order Broadcast**: The wallet sends the order and its signature to one or more relays. Relays update their public order book. The protocol doesn't require order books to be built in a certain way, such as first-come-first-serve. Instead, relays have the power to make their own design decisions in building their order books.

4. **Liquidity Sharing**: Relays broadcast the order to other relays through any arbitrary communication medium. Once again, there is flexibility how/whether nodes interact. To facilitate a certain level of network connectivity, there is a built-in liquidity sharing relay-mesh using a consortium blockchain. As mentioned in the prior section, this relay-mesh is optimized for speed and inclusivity.

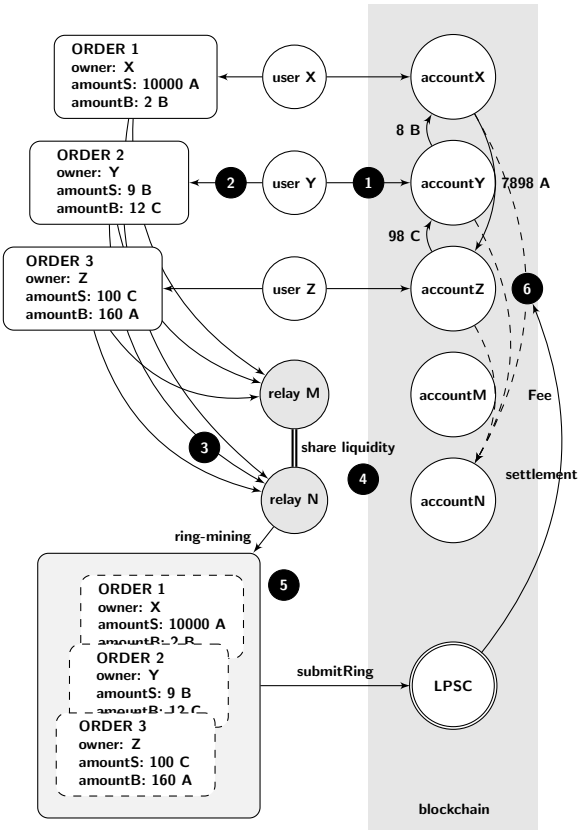

Figure 2: Loopring Exchange Process

5. **Ring-Mining (Order Matching)**: Ring-miners try to fill the order fully or partially at the given exchange rate or better by matching it with multiple other orders. Ring-mining is the main reason why the protocol is able to provide high liquidity over any pair. If the executed rate is better than what user Y specified, margin is shared amongst all orders in the order-ring. As a reward, the ring-miner chooses between claiming part of the margin (Margin-Split, and giving back the LRx to the user), or simply keeping the LRx fee.

6. **Verification & Settlement**: The order-ring is received by LPSC. It makes multiple checks to verify the ring-miner supplied data and determines if the

order-ring can be settled fully or partially (depending on the fill rate of orders in-ring and tokens in users' wallets). If all checks are successful, the contract atomically transfers the tokens to users and pays the ring-miner and wallet fees at the same time. If user `Y`'s balance as determined by the LPSC is insufficient, it will be considered scaled-down: a scaled-down order will automatically scale up to its original size if sufficient funds are deposited to its address, unlike a cancellation, which is a one way manual operation and cannot be reversed.

6 Operational Flexibility

It's important to note that Loopring's open standard allows participants significant flexibility in how they operate. Actors are free to implement novel business models and provide value for users, earning LRx fees on volume or other metrics in the process (if they so choose). The ecosystem is modular and meant to support participation from a multitude of applications.

6.1 Order Book

Relays can design their order books in any number of ways to display and match users' orders. A first implementation of our own order book follows an OTC model, where limit orders are positioned based on price alone. Timestamps of orders, in other words, have no bearing on the order book. However, a relay is free to design their order book in such a way as to emulate a typical centralized exchange's matching engine, where orders are ranked by price, while respecting timestamps as well. If a relay was inclined to offer this type of order book, they can own/integrate with a wallet, and have those wallet orders sent solely to the single relay, who would then be able to match orders based on time. Any such configuration is possible.

Whereas other DEX protocols at times require Relays to have resources - initial token balances to place taker orders - Loopring Relays need only find matchable orders to consummate a trade, and can do so without initial tokens.

6.2 Liquidity Sharing

Relays are free to design how they share liquidity (orders) with each other. Our consortium blockchain is but one solution to accomplish this, and the ecosystem is free to network and communicate as they wish. Besides joining a consortium blockchain, they can build and manage their own, creating rules/incentives as they see fit. Relays can also work alone, as seen in the time-sensitive wallet implementation. Of course, there are clear advantages in communicating with other Relays in pursuit of network effects, however, different business models could merit peculiar sharing designs and split fees in any number of ways.

7 Protocol Specification

7.1 Anatomy of an Order

An order is a pack of data that describes the intent of the user's trade. A Loopring order is defined using the Uni-Directional Order Model, or UDOM, as follows:

```
message Order {
  address protocol;
  address owner;
  address tokenS;
  address tokenB;
  uint256 amountS;
  uint256 amountB;
  unit256 lrcFee
  unit256 validSince; // Seconds since epoch
  unit256 validUntil; // Seconds since epoch
  uint8   marginSplitPercentage;  // [1-100]
  bool    buyNoMoreThanAmountB;
  uint256 walletId;
  // Dual-Authoring address
  address authAddr;
  // v, r, s are parts of the signature
  uint8   v;
  bytes32 r;
  bytes32 s;
  // Dual-Authoring private-key,
  // not used for calculating order's hash,
  // thus it is NOT signed.
  string  authKey;
  uint256 nonce;
}
```

To ensure the origin of the order, it is signed against the hash of its parameters, excluding `authAddr`, with the user's private-key. The `authAddr` parameter is used for signing order-rings that this order is part of, which prevents front-running. Please reference section 9.1 for more details. The signature is represented by the `v`, `r`, and `s` fields, and is sent alongside the order parameters over the network. This guarantees the order stays immutable during its whole lifetime. Even though the order never changes, the protocol can still compute its current state based on the balance of its address along with other variables.

UDOM doesn't include a price (which must be a floating-point number by nature), but, instead uses the term `rate` or r, which is expressed as `amountS/amountB`. The rate is not a floating-point number but an expression that will only be evaluated with other unsigned integers on demand, to keep all intermediate results as unsigned integers and increase calculation accuracy.

7.1.1 Buy Amounts

When a ring-miner ring-matches orders, it's possible that a better rate will be executable, allowing users to get

more `tokenB` than the `amountB` they specified. However, if `buyNoMoreThanAmountB` is set to `True`, the protocol ensures users receive no more than `amountB` of `tokenB`. Thus, UDOM's `buyNoMoreThantokenB` parameter determines when an order is considered completely filled. `buyNoMoreThantokenB` applies a cap on either `amountS` or `amountB`, and allows users to express more granular trade intentions than traditional buy/sell orders.

For example: with `amountS` = 10 and `amountB` = 2, the rate $r = 10/2 = 5$. Thus the user is willing to sell 5 `tokenS` for each `tokenB`. The ring-miner matches and finds the user a rate of 4, allowing the user to receive 2.5 `tokenB` instead of 2. However, if the user only wants 2 `tokenB` and set the `buyNoMoreThanAmountB` flag to `True`, the LPSC performs the transaction at a rate of 4 and the user sells 4 `tokenS` for each `tokenB`, effectively saving 2 `tokenS`. Keep in mind this does not take into account mining fees (See section 8.1).

Indeed, if we use

```
Order(amountS,tokenS,
      amountB,tokenB,
      buyNoMoreThantokenB)
```

to represent an order in a simplified form, then for ETH/USD markets on a traditional exchange, traditional buy-sell modeling can express the 1st and the 3rd order below, but not the other two:

1. Sell 10 ETH at price of 300 USD/ETH. This order can expressed as: `Order(10, ETH, 3000, USD, False)`.

2. Sell ETH at price of 300 USD/ETH to get 3000 USD. This order can expressed as: `Order(10, ETH, 3000, USD, True)`.

3. Buy 10 ETH at price of 300 USD/ETH, This order can expressed as: `Order(3000, USD, 10, ETH, True)`.

4. Spend 3000 USD to buy as many ETH as possible at price of 300 USD/ETH, This order can expressed as: `Order(3000, USD, 10, ETH, False)`.

7.2 Ring Verification

The Loopring Smart Contracts do not perform exchange rate or amount calculations, but must receive and verify what the ring-miners supply for these values. These calculations are done by ring-miners for two main reasons: (1) the programming language for smart contracts, such as solidity[19] on Ethereum, does not have support for floating point math, especially $pow(x, 1/n)$ (calculating the n-th root of a floating point number), and (2) it is desirable for the computation to be made off-chain to reduce blockchain computation and cost.

7.2.1 Sub-Ring Checking

This step prevents arbitrageurs from unfairly realizing all the margin in an order-ring by implementing new orders within it. Essentially, once a valid order-ring is found by a ring-miner, it could be tempting to add other orders to the order-ring to fully absorb the users' margin (rate discounts). As illustrated by figure 3 below, carefully calculated $x1$, $y1$, $x2$ and $y2$ will make the product of all orders' rate be exactly 1 so there will be no rate discount.

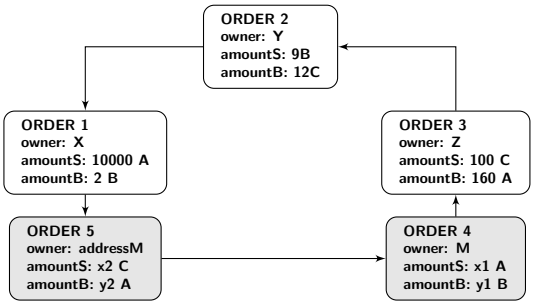

Figure 3: An order-ring with sub-ring

This is zero-risk, zero-value add to the network, and is considered unfair conduct by the ring-miner. To prevent this, Loopring requires that a valid loop cannot contain any sub-rings. To check this, the LPSC ensures a token cannot be in a buy or sell position twice. In the above diagram, we can see that token `A` is a sell token twice and a buy token twice, which would be disallowed.

7.2.2 Fill Rate Checking

The exchange rate calculations in the order-ring are made by ring-miners for reasons stated above. It is the LPSC that must verify they're correct. First, it verifies that the buy rate the ring-miner can execute for each order is equal to or less than the original buy rate set by the user. This ensures the user gets at least the exchange rate they asked for or better on the transaction. Once the exchange rates are confirmed, the LPSC ensures that each order in the order-ring shares the same rate discount. For instance, if the discounted rate is γ, then the price for each order will be:

$r_{0\to1} \cdot (1-\gamma)$, $r_{1\to2} \cdot (1-\gamma)$, $r_{2\to0} \cdot (1-\gamma)$, and satisfy:

$$r_{0\to1} \cdot (1-\gamma) \cdot r_{1\to2} \cdot (1-\gamma) \cdot r_{2\to0} \cdot (1-\gamma) = 1 \tag{3}$$

hence:

$$\gamma = 1 - \frac{1}{\sqrt[3]{r_{0\to1} \cdot r_{1\to2} \cdot r_{2\to0}}}. \tag{4}$$

If the transaction crosses n orders, the `discount` is:

$$\gamma = 1 - \frac{1}{\sqrt[n]{\prod_{i=0}^{n-1} r^i}}, \tag{5}$$

where r^i is the order turnover rate of i-th order. Obviously, only when the discount rate is $\gamma \geq 0$, can these orders

be filled; and the i-th order (O^i)'s actual exchange rate is $\hat{r^i} = r^i \cdot (1-\gamma)$, $\hat{r^i} \leq r^i$.

Recall our prior example where Alice has 15 token `A` and wants 4 token `B` for them, Bob has 10 token `B` and wants 30 token `A` for them. If token `A` is the reference, then Alice is buying token `B` for $\frac{15}{4} = 3.75$`A`, while Bob is selling token `B` for $\frac{30}{10} = 3.00$`A`. To calculate the discount: $\frac{150}{120} = 1.25$ thus $\frac{1}{1.25} = 0.8 = (1-\gamma)^2$. Thus the exchange rate that renders the trade equitable for both parties is $\sqrt{0.8} \cdot 3.75 \approx 3.3541$ token `A` per token `B`.

Bob gives 4 token `B` and receives 13.4164 token `A`, more than the 12 he was expecting for those 4 tokens. Alice receives 4 token `B` as intended but gives only 13.4164 token `A` in exchange, less than the 15 she was willing to give for those 4 tokens. Note, a portion of this margin will go towards paying fees to incentivize miners (and wallets). (See section 8.1).

7.2.3 Fill Tracking & Cancellation

A user can partially or fully cancel an order by sending a special transaction to the LPSC, containing the details about the order and the amounts to cancel. The LPSC takes that into account, stores the amounts to cancel, and emits an `OrderCancelled` event to the network. The LPSC keeps track of filled and cancelled amounts by storing their values using the order's hash as an identifier. This data is publicly accessible and `OrderCancelled` / `OrderFilled` events are emitted when it changes. Tracking these values is critical for the LPSC during the order-ring settlement step.

LPSC also supports cancelling all orders for any trading pair with the `OrdersCancelled` event and cancelling all orders for an address with the `AllOrdersCancelled` event.

7.2.4 Order Scaling

Orders are scaled according to the history of filled and cancelled amounts and the current balance of the senders' accounts. The process finds the order with the smallest amount to be filled according to the above characteristics and uses it as a reference for scaling all transactions in the order-ring.

Finding the lowest value order can help to figure out the fill volume for each order. For instance, if the i-th order is the lowest value order, then the number of tokens sold from each order $\hat{s}$ and number of tokens purchased $\hat{b}$ from each order can be calculated as:

$$\hat{s^i} = \overline{s}_i,\ \hat{b^i} = \hat{s^i}/\hat{r^i},\ ;$$
$$\hat{s^{i\oplus 1}} = \hat{b^i},\ \hat{b^{i\oplus 1}} = \hat{s^{i\oplus 1}}/\hat{r^{i\oplus 1}};$$
$$\hat{s^{i\oplus 2}} = \hat{b^{i\oplus 1}},\ \hat{b^{i\oplus 2}} = \hat{s^{i\oplus 2}}/\hat{r^{i\oplus 2}};$$
$$...$$

where $\overline{s}_i$ is the balance left after orders are partially filled.

During implementation we can safely assume any order in the order-ring to have the lowest value, then iterate through the order-ring at most twice to calculate each order's fill volume.

Example: If the smallest amount to be filled compared to the original order is 5%, all the transactions in the order-ring are scaled down to 5%. Once the transactions are completed, the order that was considered to have the smallest amount remaining to be filled should be completely filled.

7.3 Ring Settlement

If the order-ring fulfills all the previous checks, the order-ring can be closed, and transactions can be made. This means that all n orders form a closed order-ring, connected as in figure 4:

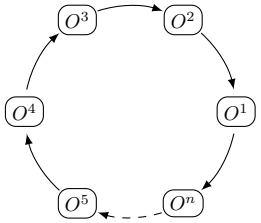

Figure 4: Ring Settlement

To make the transactions, the LPSC uses the `TokenTransferDelegate` smart contract. The introduction of such a delegate makes upgrading the protocol smart contract easier as all orders only need to authorize this delegate instead of different versions of the protocol.

For each order in the order-ring, a payment of `tokenS` is made to the next or the previous order depending on the implementation. Then the ring-miner's fee is paid depending on the fee model chosen by the ring-miner. Finally, once all the transactions are made, a `RingMined` event is emitted.

7.3.1 Emitted Events

The protocol emits events that allow relays, order browsers, and other actors to receive order book updates as efficiently as possible. The emitted events are:

- **OrderCancelled**: A specific order has been cancelled.
- **OrdersCancelled**: All orders of a trading pair from an owning address have been cancelled.
- **AllOrdersCancelled**: All orders of all trading pairs from an owning address have been cancelled.
- **RingMined**: An order-ring has been settled successfully. This event contains data related to each inner-ring token transfer.

8 LRx Token

LRx is our generalized token notation. LRC is the Loopring token on Ethereum, LRQ on Qtum, and LRN on NEO, etc. Other LRx types will be introduced in the future as Loopring is deployed on other public blockchains.

8.1 Fee Model

When a user creates an order, they specify an amount of LRx to be paid to the ring-miner as a fee, in conjunction with a percentage of the margin (`marginSplitPercentage`) made on the order that the ring-miner can claim. This is called the margin split. The decision of which one to choose (fee or margin split) is left to the ring-miner.

A representation of the margin split:

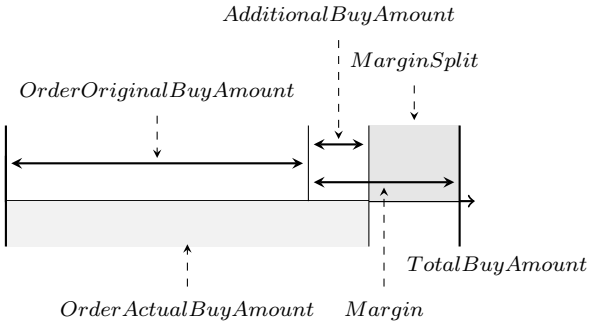

Figure 5: A 60% Margin Split

If the margin on the order-ring is too small, a ring-miner will choose the LRx fee. If, on the contrary, the margin is substantial enough for the resulting margin split to be worth much more than the LRx fee, a ring-miner will choose the margin split. There is another proviso, however: when the ring-miner chooses the margin split, they must pay the user (order creator) a fee, which is equal to the LRx the user would have paid to the ring-miner as a fee. This increases the threshold of where the ring-miner will choose the margin split to twice the LRx fee of the order, increasing the propensity of the LRx fee choice. This allows ring-miners to receive a constant income on low margin order-rings for the tradeoff of receiving less income on higher margin order-rings. Our fee model is based on the expectation that as the market grows and matures, there will be fewer high margin order-rings, thus necessitating fixed LRx fees as incentive.

We end up with the following graph:

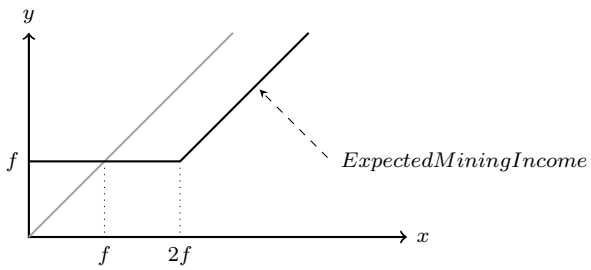

Figure 6: Loopring's Fee Model

where f is the LRx fee, x is the margin split, y is the mining income. $y = max(f, x - f)$ as indicated by the solid line; if the LRx fee for the order is 0, the equation is $y = max(0, x - 0)$ that simplifies to $y = x$ as indicated by the gray line.

The consequences are:

1. If the margin split is 0, ring-miners will choose the flat LRx fee and are still incentivized.
2. If the LRx fee is 0, the gray line results and the income is based on a general linear model.
3. When the margin split income is greater than 2x(LRx fee), ring-miners choose the margin split and pay LRx to the user.

It should be noted that if the LRx fee is non-zero, no matter which option the ring-miner chooses, there will always be a transfer of LRx between the ring-miner and the order's sender. Either the ring-miner earns the LRx fee, or pays the LRx fee back to the sender to take the margin split.

Ring-miners will share a certain percentage of fees with wallets. When a user places an order through a wallet and gets filled, the wallet is rewarded with a portion of the fees or margin split. Although this is modular, and unique business models or implementations are possible, our inclination is for wallets to receive approximately 20%-25% of earned fees. Wallets represent a primary target for Loopring protocol integration as they have the user base, but little or no source of income.

8.2 Decentralized Governance

At its root, the Loopring protocol is a social protocol in the sense that it relies on coordination amongst members to operate effectively towards a goal. This is not dissimilar to cryptoeconomic protocols at large, and indeed, its usefulness is largely protected by the same mechanisms of coordination problems [20], grim trigger equilibrium, and bounded rationality. To this end, LRx tokens are not only used to pay fees, but also to align the financial incentives of the various network participants. Such alignment is necessary for broad adoption of any protocol, but is particularly acute for exchange protocols, given that success rests largely on improving liquidity in a robust decentralized ecosystem.

LRx tokens will be used to effectuate protocol updates through decentralized governance. Smart contract updates will, in part, be governed by token holders to ensure continuity and safety, and to attenuate the risks of siphoned liquidity through incompatibility. Given that smart contracts cannot be altered once deployed, there is a risk that dApps or end users continue to interact with deprecated versions and preclude themselves from updated contracts. Upgradeability is crucial to the protocol's success as it must adapt to market demands and the underlying blockchains.

Decentralized governance by LRx stakeholders will allow for protocol smart contract updates without disrupting dApps or end users, or relying too heavily on smart contract abstraction. LRx tokens have a fixed supply, and in the case of LRC, certain percentages are frozen from the Loopring Foundation, and allocated to community-purposed funds [21].

However, LRx token owners are not the only stakeholders to consider in steering the protocol's direction: relays/ring-miners, wallets, developers, and others are an integral part of the ecosystem and their voice must be heard. In fact, given that these agents need not hold any LRx to perform their respective roles (since traditional makers/takers and market-makers are nonexistent, initial token reserves are not mandatory) we must allow alternative methods for their interests to be respected. Furthermore, "simple" token-based voting, both on-chain and off, is an imperfect salve for disagreement, as low voter turnout and token ownership concentration pose risks. Thus, the goal is to implement a governance model that is built in layers, and rests on a shared knowledge that some set of decision-making processes is the norm. This can be facilitated by coordination institutions that offer signals from a diverse set of participants, and, perhaps, from pre-established protocol focal points. As this comes to fruition, the Loopring Foundation will inevitably evolve from protocol developers into protocol stewards.

9 Fraud and Attack Protections

9.1 Front-running Prevention

In decentralized exchanges, front-running is when someone tries to copy another node's trade solution, and have it mined before the original transaction that is in the pending transaction pool (mempool). This can be achieved by specifying a higher transaction fee (gas price). The major scheme of front-running in Loopring (and any protocol for order-matching) are order-filch: when a front-runner steals one or more orders from a pending order-ring settlement transaction; and, specific to Loopring: when a front-runner steals the entire order-ring from a pending transaction.

When a submitRing transaction is not confirmed and is still in the pending transaction pool, anyone can easily spot such a transaction and replace `minerAddress` with their own address (the `filcherAddress`), then they can re-sign the payload with `filcherAddress` to replace the order-ring's signature. The filcher can set a higher gas price and submit a new transaction hoping block-miners will pick his new transaction into the next block instead of the original submitRing transaction.

Previous solutions to this problem had important downsides: requiring more transactions and thus costing ring-miners more gas; and taking at least twice the blocks to settle an order-ring. Our new solution, Dual Authoring[22], involves the mechanism of setting up two levels of authorization for orders - one for settlement, and one for ring-mining.

Dual Authoring process:

1. For each order, the wallet software will generate a random public-key/private-key pair, and put the key pair into the order's JSON snippet. (An alternative is to use the address derived from the public-key instead of the public-key itself to reduce byte size. We use `authAddr` to represent such an address, and `authKey` to represent `authAddr`'s matching private-key).

2. Compute the order's hash with all fields in the order except `r`, `v`, `s`, and `authKey`), and sign the hash using the `owner`'s private-key (not `authKey`).

3. The wallet will send the order together with the `authKey` to relays for ring-mining. Ring-miners will verify that `authKey` and `authAddr` are correctly paired and the order's signature is valid with respect to `owner` address.

4. When an order-ring is identified, the ring-miner will use each order's `authKey` to sign the ring's hash, `minerAddress`, and all the mining parameters. If an order-ring contains n orders, there will be n signatures by the n `authKey`s. We call these signatures the `authSignature`s. The ring-miner may also need to sign the ring's hash together with all mining parameters using `minerAddress`'s private-key.

5. The ring-miner calls the submitRing function with all the parameters, as well as all the extra `authSignature`s. Notice that `authKey`s are NOT part of the on-chain transaction and thus remain unknown to parties other than the ring-miner itself.

6. The Loopring Protocol will now verify each `authSignature` against the corresponding `authAddr` of each order, and reject the order-ring if any `authSignature` is missing or invalid.

The result is that now:

- The order's signature (by the private-key of the `owner` address) guarantees the order cannot be modified, including the `authAddr`.

- The ring-miner's signature (by the private-key of the `minerAddress`), if supplied, guarantees nobody can use his identity to mine an order-ring.

- The `authSignature`s guarantees the entire order-ring cannot be modified, including `minerAddress`, and no orders can be stolen.

Dual Authoring prevents ring-filch and order-filch while still ensuring the settlement of order-rings can be done in one single transaction. In addition, Dual Authoring opens doors for relays to share orders in two ways: non-matchable sharing and matchable sharing. By default,

Loopring operates an OTC model and only supports limit-price orders, meaning that orders' timestamps are ignored. This implies that front-running a trade has no impact on the actual price of that trade, but does impact whether it gets executed or not.

10 Other Attacks

10.1 Sybil or DOS Attack

Malicious users – acting as themselves or forged identities – could send a large amount of small orders to attack Loopring nodes. However, since we allow nodes to reject orders based on their own criteria – which they may hide or reveal – most of these orders will be rejected for not yielding satisfying profit when matched. By empowering relays to dictate how they manage orders, we do not see a massive tiny order attack as a threat.

10.2 Insufficient Balance

Malicious users could sign and spread orders whose order value is non-zero but whose address actually has zero balance. Nodes could monitor and notice that some orders actual balance is zero, update these order states accordingly and then discard them. Nodes must spend time to update the status of an order, but can also choose to minimize the effort by, for example, blacklisting addresses and dropping related orders.

11 Summary

The Loopring protocol sets out to be a foundational layer for decentralized exchange. In so doing, it has profound repurcussions in how people exchange assets and value. Money, as an intermediate commodity, facilitates or replaces barter exchange and solves the double coincidence of wants problem [23], whereby two counterparties must desire each other's distinct good or service. Similarly, Loopring protocol aims to dispense of our dependencies on coincidence of wants in trading pairs, by using ring matching to more easily consummate trades. This is meaningful for how society and markets exchange tokens, traditional assets, and beyond. Indeed, just as decentralized cryptocurrencies pose threat to a nation's control over money, a combinatorial protocol that can match traders (consumers/producers) at scale, is a theoretical threat to the concept of money itself.

Protocol benefits include:

- Off-chain order management and on-chain settlement means no sacrifice in performance for security.
- Greater liquidity due to ring-mining and order sharing.
- Dual Authoring solves the pernicious problem of front running faced by all DEXs and their users today.
- Free, public smart contracts enable any dApp to build or interact with the protocol.
- Standardization among operators allows for network effects and an improved end user experience.
- Network maintained with flexibility in running order books and communicating.
- Reduced barriers to entry means lower costs for nodes joining the network and end users.
- Anonymous trading directly from user wallets.

12 Acknowledgements

We would like to express our gratitude to our mentors, advisers and to the many people in the community that have been so welcoming and generous with their knowledge. In particular, we would like to thank Shuo Bai (from ChinaLedger); Professor Haibin Kan; Alex Cheng, Hongfei Da; Yin Cao; Xiaochuan Wu; Zhen Wang, Wei Yu, Nian Duan, Jun Xiao, Jiang Qian, Jiangxu Xiang, Yipeng Guo, Dahai Li, Kelvin Long, Huaxia Xia, Jun Ma, and Encephalo Path for reviewing and providing feedback on this project.

Appendices

Appendix A New Fee Model in Loopring 2.0

One of the most exciting developments of 2.0 has been reimagining the protocol fee model, the role of our native token, LRC, within it, and how we can ensure token holders capture the value of the Loopring network.

A.1 Why Were Open to Change

Utility tokens that act as a medium of exchange within a particular ecosystem have been popular in terms of projects issuing them, but, more recently, less popular in terms of expected value capture and actual usefulness.

Beyond the questionable value accretion for these types of utility tokens, the imposition of a specific token as the means of payment creates significant friction. dApps, including DEXs, are currently UX un-friendly enough without requiring users pay with an esoteric and otherwise non-demanded token. Abstracting away token transfer behind the scenes helps, but is still only a half-solution.

Open financial infrastructure is in its infancy, and what was considered best practice last year will certainly change. Projects, protocols, and builders must remain adaptive and open to tweak their token dynamics, lest they stubbornly cling to suboptimal models.

A.2 Objectives

We knew we wanted to:

- Make fee payment more flexible: remove the friction of **requiring** a specific fee-payment token.
- Ensure the LRC token is coveted and provides real utility to the network.
- Remain rent-free, with all benefits remaining in-protocol, accruing to every participant.

We realize that while incentives are the most powerful force to allow rational actors to organize themselves, mechanisms which optimize for the desired network behavours are valuable.

A.3 Introducing the LRC Burn Rate

With this in mind, we have introduced a fee element, **LRC burn rate** (or *burn rate*). **Wallets and ring-miners who earn fees by fulfilling roles on the Loopring protocol will have a portion of their fees converted into LRC (if not LRC already) and burned**.

In the example below, the user pays for their trading fees using BNB. It does not matter if the trade actually involves BNB or not. Lets assume the protocol-defined LRC burn rate for the BNB token is 20%, and the fee for the transaction is 100BNB. At settlement, the trader pays 100BNB and the wallet and relay earn 100*(120%)=80BNB, **with the 20BNB being sent to a smart contract which buys and burns LRC.**

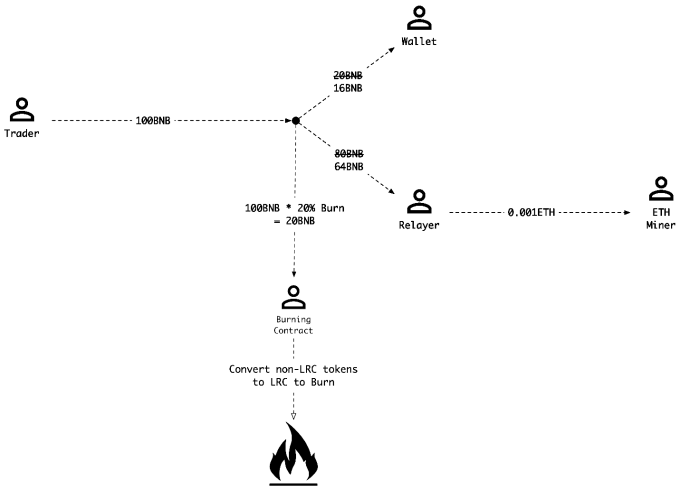

Figure 7: Fee model in Loopring protocol 2.0

A.4 Burn Rate Settings

If the above fee were to be paid in LRC instead of BNB, for example, the burn rate would be lower, such as 5%, allowing the wallet and ring-miner to keep more of their earned fees. The burn rate associated with different ERC20 tokens will be configurable by LRC token holders governing the protocol. For now, we propose the following:

Tiers	Tokens	Burn Rate for Normal Orders	Burn Rate for P2P Orders	LRC Burning Cost of Buying Down Rate to a Lower Tier for 2 Years
Tier1	LRC	5%	0.5%	NA
Tier2		20%	2%	0.5%
Tier3	WETH	40%	3%	0.5%
Tier4	Any ERC20 tokens	60%	6%	0.5%

Figure 8: Default settings, but will be configurable.

Now, a user can pay fees in any token, but there will be preferable treatment, and thus lower cost, to pay fees in LRC.

The preferable treatment is for the fee-earning participants: wallets and relays; the fee to the trader is the same. However, our reasonable assumption is that wallets and relays will pass-upwards the fee (dis)savings, and calculate different fees for payment in different tokens.

A.5 Other Projects Burning LRC

We want other protocols and projects to be able to take advantage of the fee model. As such, it is possible for **a third-party project to burn a certain amount of LRC to knock down the burn rates applied to their token when users pay fees in it.** The exact burn rate schedule is TBD, and again, will ultimately be configurable.

If you want to lower the rate of a certain token, such as GTO, to the same third tier burn rate as WETH for 24 months, you may need to burn 1% of the total LRC supply (currently equivalent to $550,000 USD); if you want to lower the burn rate to the first tierLRCs burn rate for 24 months, it may be necessary to burn 1.5% of the total LRC. **As more LRC are burned, the total LRC supply will decrease, and the number of LRC that need to be burned to switch tiers will decrease accordingly.**

Lowering the burn rate is particularly appealing for other decentralized exchanges that use their own platform payment tokens:

1. The cost of burning LRC to obtain a lower rate is much less than the cost of developing a new DEX/protocol, or bootstrapping liquidity.

2. This mechanism actually gives the respective DEX token increased utility and value.

3. After the third-party DEX platform token is enabled for a lower burn rate, using that token to pay fees is no different to the user than using LRC, which should enhance the competitiveness of the platform.

A.6 P2P vs Normal Orders

In 2.0, we distinguish between **normal orders**as seen in all the examples aboveand P2P orders.

Definition A.1 (P2P Order) *If the orders* `tokenSFeePercentageor` *or* `tokenBFeePercentage` *is greater than zero, the order is a P2P order. Other fee parameters in the P2P order will be ignored by the protocol.*

Normal Order Parameters:

- `address feeToken`: optional and defaults to LRC token address if not specified. This is the type of ERC20 token to be paid to the miner as matching fees.
- `unit feeAmount`: optional and defaults to 0. This is the amount of feeToken to be paid. This number should be calculated by the wallet software off-chain.
- `uint16 feePercentage`: optional and defaults to 0. This is the percentage of tokenB (token that user is buying) to be paid as matching fees if the order does not have enough feeToken to pay. This parameter should also be supplied by the wallet.
- `uint16 waiveFeePercentage`: optional and defaults to 0. If this number is positive, the miner will waive a waiveFeePercentage% percentage of the fees before the burn; if this number is negative, the miner will reward-waiveFeePercentage%of its total fee after the burn to the orderthe reward is also subject to burn.

P2P Order Parameters:

- `uint16 tokenSFeePercentage`: optional and defaults to 0. The percentage of tokenS to charge by the wallet.
- `uint16 tokenBFeePercentage`: optional and defaults to 0. The percentage of tokenB to charge by the wallet.

A normal order is what most of us are used to: we submit an order to a DEX and leave it to them to match and settle it on the Ethereum blockchain. On the backend, a ring-miner is the one finding the match and submitting it, and that is why you pay them a fee in the first place.

P2P orders do not pay matching fees, because they do not want to be matched by a third party (ring-miner), but want to match it themselves with a counter-party. Thus, **the fees for P2P orders are paid only to the wallet**.

However, the wallet here is a broader concept, including any tools that can help you generate orders. As a fee, the wallet can obtain a percentage of the purchase token (`tokenB`) and the sell token (`tokenS`) from a P2P order. This percentage is set by the wallet, approved by the user, and is not provided by the protocol itself.

Lets assume that both fees set by a wallet are 0%. In the example below, two P2P orders are swapped between 100DAI and 400GTO:

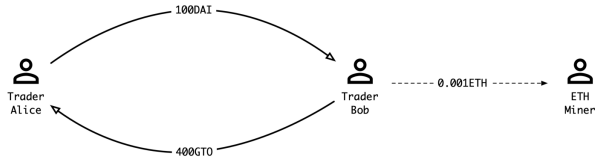

Figure 9: Two orders of zero-fee P2P orders

However, if the wallets corresponding to the two P2P orders are set to charge different `tokenSFeePercentage` or `tokenBFeePercentage`, the actual fee and transaction calculation is more complicated:

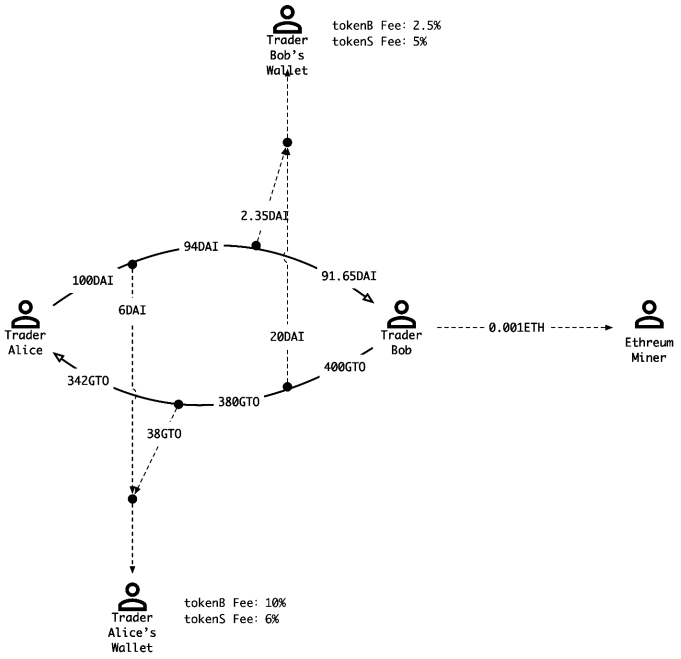

Figure 10: Two orders consisting of P2P orders with commissions

There are many instances in the real world that will make use of P2P orders and do not require third-party matching:

1. You create an order for 100 AnyTokens for 1 WETH on your Loopring-enabled wallet, and show the QR code to your friend. She scans it from her wallet, signs it, and submits it directly to the Ethereum blockchain in one tap, resulting in a trustless token swap. Your friend was the ring-miner here, because she submitted the transaction.
2. Youre launching your ICO and plan to distribute it by submitting sell orders for your NewToken on a Loopring-built DEX. You send this order out in emails to your subscribers, or put the QR code printed on the back of your teams t-shirts. Any interested buyer can fill these orders with a Loopring-enabled wallet, and act as ring-miner by submitting it as a transaction to the Ethereum blockchain.

In the ICO example, the wallet/ICO platform can charge the project fees for this service in a few different ways:

1. Use NewTokens as income. For example, for every 100 NewTokens sold, 5 NewTokens are charged from the project side as a handling fee (the project party distributes a total of 105 tokens, the buyer buys 100 tokens); or every 100 NewTokens sold, 5 are charged from the buyer (the project party distributes a total of 100 tokens and the buyer receives 95 tokens).

2. Use ETH as income. For example, if there are 100 ETH received from ICO buyers, 7 ETH will be charged from the project side (the project party will get 93 ETH); or the 7 ETH will be charged from the user (the user actually needs to pay 107 ETH).

3. Any combination of the above four different methods.

Without the new P2P model in 2.0, these scenarios cannot be achieved.

A.7 P2P Fee Burn Rate

For P2P orders, the protocol specifies different burn rates. Assuming that the burn rates for DAI and GTO are 10% and 20%, respectively, the transaction calculations need to be adjusted as follows:

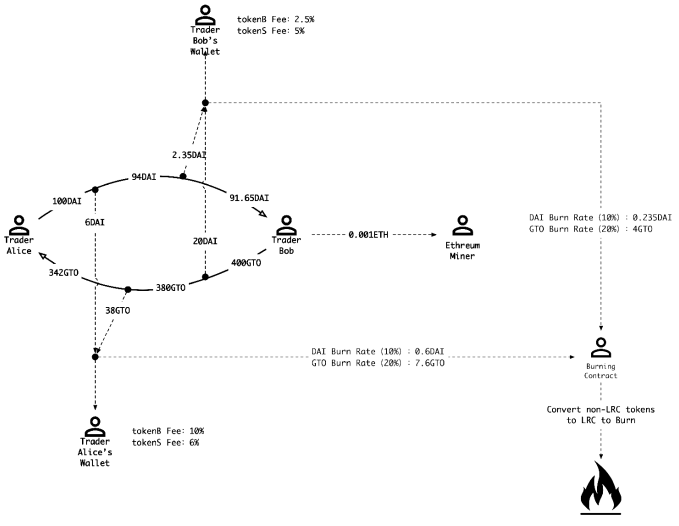

Figure 11: P2P fee burn rates. See the table in the 'Burn Rate Setting' section above for default P2P burn rates.

Burn Contract Specifications With the new fee model, a small piece of every transaction will be directed to this smart contract and ultimately reduce LRC supply. As such, more Loopring network activity results in more burn, and in LRC holders owning a larger share of the token supply. So now, we have an economy where LRC holders:

1. **Pay the lowest fees for their transactions.**

2. **Benefit from all network-wide activity.**

You may have a few questions:

(Q1) ***Who burns the LRC tokens?***

Given that the Loopring Foundation is a non-profit and ultimately wants to take a backseat as the protocol flourishes, the burning smart contract (aka holding smart contract) will be coded to allow any user (willing to pay the small gas cost to trigger the function) to burn the tokens.

(Q2) ***What happens to the non-LRC (WETH, BNB, etc) that are sent to the burning smart contract?***

Like the burn rates, this will ultimately be configurable. LRC holders may be inclined to send the tokens to some external address to fund a community project, but for now, they will be used to **buy and burn more LRC**.

The means by which non-LRC tokens will be sold for LRC is quite an interesting topic, and highlights the spirit of open financial infrastructure. We will allow anyone (willing to pay the gas cost) to trigger functionality in the smart contract that queries other DEXsspecifically Dutch auction modelsfor any tokens price vs LRC, and submits the tokens to be sold on the open market. This allows the burning contract to trustlessly sell our non-LRC tokens at discrete times, all at once.

The purchased LRC will then be burned. This has the effect of:

- **Increasing LRC demand.**

- **Reducing LRC supply.**

A.8 LRC Lockup for Fee Rebates

Loopring 2.0 also allows LRC token holders to lock up LRC in a smart contract in exchange for a percentage of fees that would have otherwise been burnedeffectively, a **rebate**. This would apply for each and every order during the lockup period.

For example, an order has a total fee of 10 AnyToken, and 4 AnyToken (40%) is to be burned. Assuming the fee payer locked up 1 million LRC and has a 50% **rebate rate**, he/she will ended up paying only 8 AnyToken, 2 of which will be burned. **The rebate comes from the fee component that is to be burned, not the wallet or relay income**. Please also note that a wallet/relay can also lockup LRC for rebates, but that doesnt apply here.

A rebate rate of 100% ensures all orders are **burn-free**. The Fee-Rebate feature will be very helpful for high frequency traders.

A.9 Incentivize Market Makers

Market making is not a protocol level consideration, but a business decision made by a ring-miner on their own. It is up to them to **incentivize market makers by waiving fees** for orders, so we introduce the parameter `waiveFeePercentage`.

If this parameter is positive, say, 50%, then if the original fee paid to the ring-miner is 80BNB, it now becomes 40BNB, before the burn. The waived fee is applied to the ring-miners part of the fee, but not the wallets.

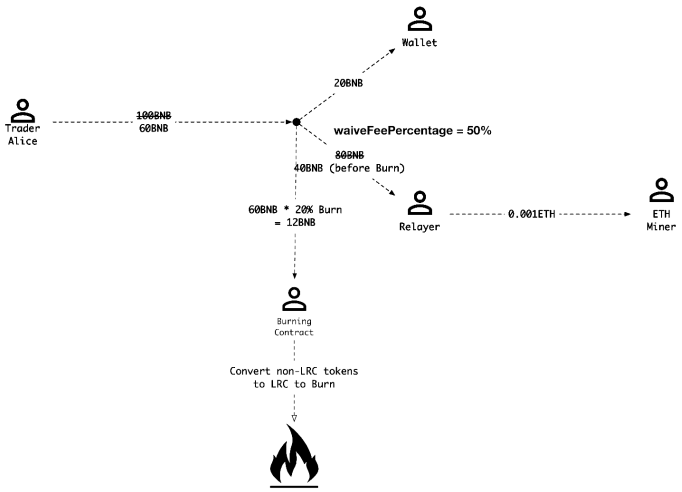

Figure 12: Waiving 50% of fee for Market Maker Alice

This `waiveFeePercentage` parameter can also be negative. If it were -50%, the order does not pay any fees to the relayer, but can also get 50% of the fee other orders pay in the ring. For example, in the below diagram, the ring-miner will pay Alice half of the 40 LRC fee paid by Bob. Alices rebates are also subject to the corresponding burn.

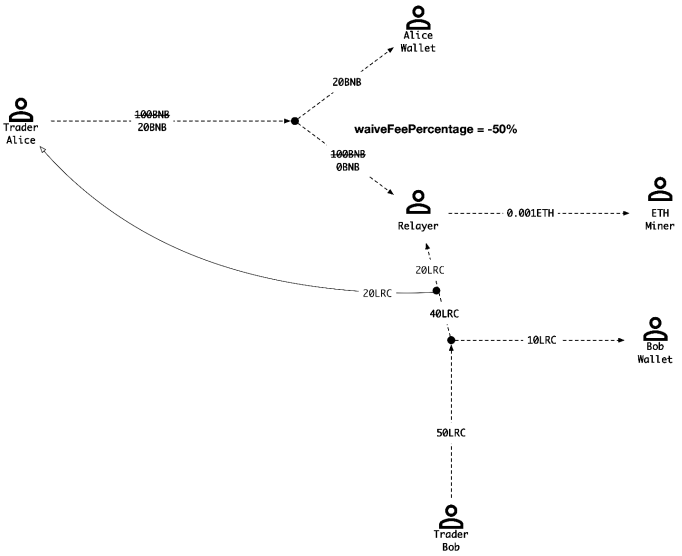

Figure 13: Giving a -50% fee rebate to Market Maker Alice

With this functionality, the relay can choose to exempt some or all of the fees of a market makers order, and even pay part of the fees earned on other orders. Market makers will be incentivized to provide liquidity to generous and prosperous ring-miners.

A.10 Decentralized Governance

Please note that in addition to its utility, weve always planned on giving LRC holders governance rights. While this is easy to say, and just about every project can throw that in as a catchall, well-defined rights are truly important to us as the Loopring Foundation will cede control. However, in isolation, we find a pure-governance token to be lacking, and are excited to reveal our 2.0 model which supports further utility and value capture.

With that said, there is now more to govern within the protocol itself. **Important future governance issues of the Loopring protocol will mainly relate to the various burn rate parameters, and which ERC20s should be placed in certain tiers.**

Decentralized governance is incredibly difficult and nuanced, and faces risks like plutocracies and voter apathy. We are currently exploring this aspect and hope to provide an upgraded version in the first half of 2019, allowing community parameter proposal and voting.

References

[1] Vitalik Buterin. Ethereum: a next generation smart contract and decentralized application platform (2013). *URL {http://ethereum. org/ethereum. html}*, 2017.

[2] Gavin Wood. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum Project Yellow Paper*, 151, 2014.

[3] Patrick Dai, Neil Mahi, Jordan Earls, and Alex Norta. Smart-contract value-transfer protocols on a distributed mobile application platform. *URL: https://qtum. org/uploads/files/ cf6d69348ca50dd985b60425ccf282f3. pdf*, 2017.

[4] Viktor Atterlönn. A distributed ledger for gamification of pro-bono time, 2018.

[5] Hernando de Soto. *The Mystery Of Capital*. Basic Books, 2000.

[6] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008.

[7] Fabian Schuh and Daniel Larimer. Bitshares 2.0: Financial smart contract platform, 2015.

[8] Bancor protocol. *URL https://bancor.network/*, 2017.

[9] Yaron Velner Loi Luu. Kybernetwork: A trustless decentralized exchange and payment service. `https://kr.kyber.network/assets/KyberNetworkWhitepaper.pdf`, Accessed: 2018-03-05.

[10] Fortune. How to steal $500 million in cryptocurrency. `http://fortune.com/2018/01/31/coincheck-hack-how`, Accessed: 2018-03-30.

[11] Robert McMillan. The inside story of mt. gox, bitcoins 460 dollar million disaster. 2014.

[12] Sylvain Ribes. Chasing fake volume: a crypto-plague. https://medium.com/@sylvainartplayribes/chasing-fake-volume-a-crypto-plague-ea1a3c1e0b5e, Accessed: 2018-03-10.

[13] Rossella Agliardi and Ramazan Genay. Hedging through a limit order book with varying liquidity. 2014.

[14] Will Warren and Amir Bandeali. 0x: An open protocol for decentralized exchange on the ethereum blockchain, 2017.

[15] Iddo Bentov and Lorenz Breidenbach. The cost of decentralization. http://hackingdistributed.com/2017/08/13/cost-of-decent/, Accessed: 2018-03-05.

[16] Daniel Wang. Coinport's implemenation of udom. https://github.com/dong77/backcore/blob/master/coinex/coinex-backend/src/main/scala/com/coinport/coinex/markets/MarketManager.scala, Accessed: 2018-03-05.

[17] Supersymmetry. Remarks on loopring. https://docs.loopring.org/pdf/supersimmetry-loopring-remark.pdf, Accessed: 2018-03-05.

[18] Fabian Vogelsteller. Erc: Token standard. *URL https://github.com /ethereum /EIPs /issues /20*, 2015.

[19] Chris Dannen. *Introducing Ethereum and Solidity*. Springer, 2017.

[20] Vitalik Buterin. Notes on blockchain governance. https://vitalik.ca/general/2017/12/17/voting.html, Accessed: 2018-03-05.

[21] Loopring Foundation. Lrc token documents. https://docs.loopring.org/English/token/, Accessed: 2018-03-05.

[22] Daniel Wang. Dual authoring looprings solution to front-running. *URL https://medium.com/loopring-protocol/dual-authoring-looprings-solution-to-front-running-d0fc9c348ef1*, 2018.

[23] Nick Szabo. Menger on money: right and wrong. http://unenumerated.blogspot.ca/2006/06/menger-on-money-right-and-wrong.html, Accessed: 2018-03-05.