— MuesliSwap —
A decentralized token exchange on Cardano

The MuesliSwap Team

October 25, 2021

Abstract

While decentralized token exchanges have been enjoying great popularity on account-based blockchains (e.g. Uniswap on Ethereum), no platform of this kind exists on EUTXO-based blockchains (i.e. Cardano) yet. With MuesliSwap, we propose a novel approach for modelling and implementing such an exchange by revisiting the traditional order book model. We further extend the list of known issues with the initial naive design, to then provide a solution that solves all those previously open problems. Our exchange protocol draws from game theoretic concepts and distinguishes itself from alternatives through simplicity, modularity, self-regulation, and full decentralization, thus paving the way for MuesliSwap to become the first operational DEX on Cardano.

1 Introduction

Exchanges have been forming the heart of financial markets for centuries. Traditionally, these exchanges have been centralized (CEX), i.e. operated and controlled by a trusted central entity. By introducing a peer-to-peer electronic cash system, the launch of Bitcoin [8] in 2009 demonstrated how cryptographic methods can be leveraged to ensure trust between two parties directly and thus obviating the need for a trusted third party. The introduction of smart contracts on the Ethereum blockchain [5] in 2015 further advanced the vision of decentralized finance, a trustless financial ecosystem, by enabling arbitrary protocols to be executed on the blockchain. As an example application, Uniswap [4] implements a decentralized token exchange using so-called liquidity pools which enable automated pricing. While Ethereum smart contracts are implemented based on an account-based transaction model, other design choices have been proposed since then. In particular, the Cardano blockchain has recently launched its version of smart contracts based on the extended unspent transaction output (EUTXO) model. [6]

The purpose of this document is to discuss the design of decentralized token exchanges for EUTXO blockchains: conceptually at first, then concretely in terms of our implementation of MuesliSwap. In order to do so, we will, in a first step, point out the key differences between the account-based and the EUTXO model, and explain the resulting consequences on the design of exchange mechanisms. Instead of adapting working approaches from the account-based to the EUTXO model, we argue for a solution derived from first principles.

Therefore, we will revisit ideas from traditional centralized exchanges and show how the EUTXO model lends itself naturally to removing the third party and realizing the same ideas directly in a decentralized setting. In a second step, we will describe the technical implementation of our model for the MuesliSwap exchange on the Cardano blockchain.

1.1 The centralized order book model

What we will call *centralized order book model* is the classical exchange protocol used in traditional stock exchanges. Traders can submit their buy and sell orders to a central market maker whose job is to then match those orders following some matching strategy. Finally, pairs of matched buyers and sellers can execute their desired trades accordingly via the trusted central entity. We distinguish between market order and limit order:

1. A market order consists of a token type A to sell, a token type B to buy, and the amount n of desired B. A market order expresses the trader's intention to buy n many B at whatever price in A they are currently available on the market. Market orders are expected to be executed either immediately or not at all (in case of unavailability).

2. A limit order consists of a token type A to sell, a token type B to buy, the amount n of desired B, and the maximum price p in A the trader is willing to pay per B. A limit order remains

pending until the market provides n many B at price at most p, then the trade gets executed. While the order is pending, the trader has the option to cancel his order.

The important design choice that remains to be made is, what type of matching algorithm to use. Different strategies – such as the Price/Time and the Pro-rate algorithm – have been proposed and used in real-world exchanges.

1.2 Account-based vs. EUTXO model

For account-based blockchains, there are already a large number of successful token exchanges in operation, typically following the liquidity pool (LP) / automated market maker (AMM) idea first proposed by [Uniswap]. Naive approaches of applying the same concepts to EUTXO (Cardano) blockchains have been made and unsurprisingly failed. It is hence crucial to be aware of the very different design choices underlying account-based and EUTXO blockchains and the consequences this yields for designing exchange mechanisms.

Fundamentally, the goal of any blockchain is to achieve consensus over the state of certain continuously updated information (e.g. who owns what amount of a particular token) in a distributed, decentralized and trustless setting. Different approaches attempting to handle the inherent concurrency in such a distributed system make different trade-offs in their design choices. One such trade-off is the amount of shared state vs. parallelism.

Account-based blockchains maintain a global shared state across all participants. Essentially, after each step (i.e. block) this shared state consists of a database containing the balance of each participant's account. On state transitions (i.e. mining of a new block), arbitrary changes can be applied in the form of smart contracts resulting in an updated database of balances everyone agrees on. Ethereum, for example, calls these computations on globally shared state "the world computer". The rationale behind such a system design is to provide a level of abstraction that takes away the burden of dealing with concurrency from the user of the blockchain. This however comes at a cost, namely that the possibility of dealing with concurrency and turning it into parallelism in the case of applications that do not need to rely on global shared state is taken away from the user as well.

The EUTXO model on the other hand does not introduce such a layer of abstraction and instead fully exposes the user to the inherent underlying concurrency of the system. An EUTXO blockchain's state is not a sequence of globally agreed upon account balances but rather just the directed acyclic graph (DAG) where vertices are unspent transaction outputs and edges are transactions. By adding mined blocks of new transactions, this graph grows over time. The crucial point here is that each UTXO can only be spent once which prohibits shared state.

Depending on the intended application, arguments for both the account-based as well as the EUTXO model can be made. The purpose of this paper is not to evaluate blockchain designs. However, we emphasize that when developing decentralized mechanisms, it is of fundamental importance to be aware of the design trade-offs that have been made on the underlying blockchain model. When those

trade-offs are recognized and understood and applications not simply adapted from another model but individually rethought, then the advantages of each blockchain model can be leveraged.

As an example, consider token exchanges. As described in Section 1.1, traditional centralized exchanges are typically based on some variation of an order book model. Hence, when smart contracts were first introduced on account-based blockchains, this design pattern was adapted to the decentralized setting. However, since on account-based blockchains, inherent parallelism is subordinated to maintaining global shared state, this model is not suitable for powering high-volume order book exchanges. In contrast, the idea of automated market makers controlling prices via liquidity pools follows the account-based design trade-offs by putting shared state in the form of the balance of liquidity pools at the heart of the exchange – explaining the success e.g. of [Uniswap]. When smart contracts were recently introduced on the EUTXO-based Cardano blockchain, again, attempts have been made to directly adapt working solutions from account-based blockchains. It is clear design principles are misaligned: Every trader aiming to access shared state in form of a liquidity pool causes contention around an UTXO that can only be spent once. In this naive version, the result is an exchange supporting only one trade per block which in the case of Cardano means one trade every 20-40 seconds.

This development motivates a fresh start. In what follows, we return to the roots of trading – the centralized order book exchange – and derive a decentralized version of it, specifically for EUTXO blockchains.

2 The MuesliSwap decentralized order book model

This section presents the exchange model underlying MuesliSwap. We start out with a basic reformulation of the centralized order book model from Section 1.1 for the decentralized setting given by an EUTXO blockchain. Several issues with this somewhat naive translation have been raised in previous discussions. After extending this list with additional problems, we get to the core of this paper. Namely, we present two simple adjustments to the naive approach that eliminate all these issues. The basic observation we build on is borrowed from game theory and goes as follows: A decentralized system regulates itself whenever incentives are set in a way such that the strategy maximizing every individual's reward coincides with the strategy that coordinates individuals towards achieving the best common outcome for the whole group.

2.1 Basic model

- As in the centralized order book model, traders can make buy and sell orders. However, we only support limit orders (for reasons to become clear; and note that market order can be viewed as a special case of a limit order that is canceled after one block). There is of course no central entity. Instead, orders are made by calling a smart contract that locks the amount of tokens one is willing to sell in a UTXO together with a fee in ADA and with information on what

the trader wants to buy and how much. This UTXO then sits on the blockchain waiting to be matched with a trade "in the other direction" – the blockchain *is* the order book. Denote a limit order by (A, B, n, p, f) expressing the intention to buy n many B at a price of at most p amount of A per B (i.e. in total $n \cdot p$) with f ADA of locked fees.

- An external matchmaker is watching the blockchain for matching exchange partners. Say it finds order O_1, O_2 such that $O_1 = (A, B, n_1, p_1, f_1)$ and $O_2 = (B, A, n_2, p_2, f_2)$ where $p_1 \geq 1/p_2$. Then it checks if $f_1 + f_2$ suffice to cover the fee needed for executing the matching contract. If so, the pair is matched, and funds are exchanged accordingly. In particular, the smaller order is only partially fulfilled and is replaced by a new limit order representing the amount remaining unfulfilled. The matchmaker keeps the difference as a reward for matching.

2.2 Problems with basic model and our solutions

- **Problem 1:** *Power of the matchmaker.*

  The goal with our basic model from above was to translate the centralized order book into a decentralized version. Yet one might point out that the concept still evolves around a single central entity, namely the matchmaker, that initiates and coordinates all exchanges – have we then achieved anything at all? On the one hand, yes we have: To some degree, the central entity needs no longer be trusted by participants of an exchange. The role of providing security during the exchange has been shifted from the matchmaker to a smart contract which is trustworthy by design. On the other hand, we argue that there also lies a certain (limited, but not negligible) power in the task of coordinating an exchange. For example, a malicious matchmaker may provide a worse exchange experience to certain traders by not including them in matches as frequently as others. In a truly decentralized system, this power should not be in the hands of a single entity.[1]

  Note that the matchmaker's power in causing unfairness does not go beyond excluding traders from the exchange. In particular, observe that it is not possible for the matchmaker to favor certain traders by matching them with lower priced orders, since the buyer-seller price difference is awarded to the matchmaker and not the trader. However, any opportunity for unfairness to emerge remains as an issue.

- **Solution to Problem 1:** *Multiple matchmakers.*

  The remedy against having a single entity is obvious: have multiple entities. More precisely, everyone is allowed to take over the role of matchmaking. And since matchmaking is rewarded, there is a clear incentive to actually participate. Every potential match pays a reward and therefore, as long as enough orders are made in both directions of an exchange, some matchmaker will match a given order even (or especially) if other matchmakers have ignored it in an unfair way. The example of a group of traders being excluded from the market is no longer realistic.

This can be compared to mining in a proof-of-work-based blockchain: If there were only a single miner, this central entity would have control over whose transactions get included in the next block. Giving everyone the opportunity to mine and rewarding it accordingly decentralizes the process. As long as there is no malicious majority, the system behaves in a fair way.

Thus at first sight, distributing both the power as well as the computational cost that comes with matchmaking across many entities seems to be a convincing solution. This solution makes full use of the concurrency exposed by an EUTXO blockchain by leveraging it for both parallelism as well as decentralization of power.

However, new challenges arise, namely the following two. [2]

- **Problem 2:** *Contention around best prices.*

  Notice that since matchmakers keep the bid-ask difference as a reward, in order to maximize their individual profit they are incentivised to match the highest available bid with the lowest available ask; and this incentive is the same for all matchmakers. The result is contention around the pair of highest bid-ask difference. To understand the consequences of contention in an EUTXO-blockchain, recall from Section 1.2 that each UTXO can only be spent once and the definite result of the spending transaction is only available in the next block. Hence, in our case, only one matchmaker can succeed, and all others are doomed to fail.

  While this is certainly an issue that matchmakers should have in mind when designing their individual matching strategy, it does not directly hinder the overall exchange from making progress. This is because matchmakers are of course not limited to attempt making just a single match per block. On the contrary, they are incentivised to submit as many matching transactions as they can, given sufficient matching partners. Many transactions will then fail but no fee is charged for failing transactions. And since many attempts to match are being made, it is unlikely that a potential match remains undiscovered, i.e. the overall exchange does not suffer from slow-down due to contention.

- **Problem 3:** *The choice of batch sizes.*

  Regarding the choice of batch sizes though, the story is different. So far, we have implicitly been assuming that a matchmaker submits separate transactions for each matching attempt. However, from the perspective of a matchmaker, it makes sense to batch multiple matches into a single transaction in order to keep transaction fees low[1]. Here's the problem: If every matchmaker naively tries to minimize fees, everyone will put all their matches into a single transaction. Overlaps between the discovered matchings are then very likely, and all but very few transaction batches will fail. Since these few succeeding transactions are likely only covering

[1] Note that e.g. on Cardano the transaction fee is calculated as some constant amount plus some amount proportional to transaction size. Here, the matchmaker will try to save on the constant part by batching matches.

a small fraction of order pairs that would be contained in a maximum matching, the overall exchange is experiencing slow-down due to contention.

Hence it is unclear how to prevent matchmakers from following their individual incentive of maximizing the number of matching attempts per transaction towards maximizing the global exchange progress.[2]

- **Solution to Problems 2 & 3:**

  Solving problems 2 and 3 seems hard. How does one coordinate a group of matchmakers – each primarily interested in maximizing their own reward – to commonly create matchings that are large, disjoint, and fair? More generally, how does one coordinate a group without relying on shared state and without any central entity? It is far from clear if there exists a solution that enforces all the properties named above in the form of EUTXO-based smart contracts.

  Instead of trying to engineer a convoluted set of restrictions that becomes difficult to implement and verify, we take a step back and look at some basics from the field of game theory. The point we are trying to make is best illustrated by a story (that in reality may have never happened) surrounding John Nash himself. Imagine John visiting a bar with his colleagues, when a group of attractive women enters the room – or, analogously, a set of outstanding orders with matchmakers waiting to match, when a group of potentially matching, oppositely directed orders enters the market:

  > *"If we all go for the blonde, we block each other and not a single one of us is going to get her. So then we go for her friends, but they will all give us the cold shoulder because nobody likes to be second choice. But what if no one goes to the blonde? We don't get in each other's way and we don't insult the other girls. That's the only way we win."*

  Previously, we have only looked at a matchmaker's decisions from a narrow self-centered perspective. If matchmakers are aware of the competitive game they are playing – which they better be, if they want to have a chance of winning – then the optimal matchmaking strategy changes. Put in the words of Mr. Nash:

  > If they all go for the best price (Problem 2) / the largest possible batch of matchings (Problem 4), then they block each other and not a single one of them is going to get a reward. So if they try worse priced orders / smaller batch sizes only afterwards, the EUTXOs might have been spent already, because there is no such thing as second choice for EUTXOs. But what if no one goes for the best price / largest batch size? They don't get in each other's way and the EUTXOs of sub-optimal prices will not already be spent. That's the only way they win.

[2] One trivial solution would of course be to only allow one matching per transaction. But as we will show, this is unnecessarily limiting and wasteful.

One may criticize that the best price always remaining unmatched is not a desired outcome. However, this is not a concern here since unlike in Nash's example, matchmakers still have a second chance to submit transactions aiming to match the best price, and because there is a high incentive to do so, this will happen. Apart from that, in real world scenarios, there is usually a risky player betting on everyone else playing it safe and directly targeting "the blonde".

These observations lead us to the conclusion that lies at the core of MuesliSwap's exchange model:

> A decentralized order book in which traders can submit their orders freely, and a group of matchmakers can initiate matches freely, regulates itself without the need for additional restrictions, because incentives are set in such a way that whenever the individual's reward is greatest, the overall exchange also makes the most progress.

We end up with a an exchange protocol that is surprisingly simple due to self-regulation, yet robust, fully-decentralized and modular (see Section 2.3).

2.3 The protocol's actors

In the introduced protocol three main actors can be identified: the traders, the matchmakers and the front-end operators. The matchmaker and the front-end operator are required to provide the necessary infrastructure for traders to perform trades. The two main infrastructure actors have unique roles. To encourage independent operators to participate in the protocol each actor has its own unique economic incentive.

2.3.1 Front-end provider

The task of the front-end provider is to provide traders with the necessary infrastructure to submit limit orders to the smart contract. Through a trading front-end, the trader should be able to indicate his trading intent. This trading intent is then used to construct a smart contract interaction with which the trader can submit his order to the blockchain. When a trader is submitting his order, a small fee will be paid to the front-end provider. This small fee can be seen as an incentive to provide a trading front-end to the MuesliSwap protocol.

2.3.2 Matchmaker

The task of the external matchmaker is to match orders submitted to the smart contract. The exact details of the matching are described in the protocol specifications. For each successful match the matchmaker submits to the blockchain, a reward is paid to him. This reward can be seen as an incentive to offer order matching.

2.3.3 Modularity via independent operators

In an ideal setting, there will be multiple front-end providers and multiple independent matchmakers. No matter on which front-end the orders have been submitted they will all be submitted to the same smart contract. That means the matchmakers will match all placed orders independently of the used front-end providers. This makes the MuesliSwap order book model a truly decentralized exchange model and helps to solve the liquidity problem across all operators.

2.4 Technical Remarks

At this point, we have established the fundamental principles supporting our exchange. There are some technical remarks that we will briefly address.

2.4.1 Handling large orders.

Certain contention problems might still arise when orders are of very different sizes. If, for example, a large order is matched with a very small order, the corresponding EUTXO is spent and this matching blocks other potentially preferable matches that would have been able to satisfy a greater fraction of the large order.

As a countermeasure, when large orders are submitted, we simply split them into multiple smaller ones, i.e. the order appears as a set of EUTXOs that can be matched with independently.

2.4.2 Collecting rewards

Reading Subsection 2.3, one quickly sees that economic incentives are the key driver behind deployment and hence availability of trading infrastructure. However, the rewards for each action in the system may be too small to be sent over the Cardano blockchain, since usually a minimum amount of around 1 ADA is required. [3] To circumvent this issue, the actors can supply their action with some sufficient amount of ADA that will be bundled with the reward and sent back to the actor during the transaction.

3 MuesliSwap's components

We are planning to operate both of the described infrastructure actors. This is to ensure that token trading will be possible as soon as the MuesliSwap protocol has been released. The two components can be viewed as independent actors.

3.1 MuesliSwap Front-end

We plan to operate a front-end through which traders can submit orders to the blockchain. Our front end will fulfil the tasks of the "front-end provider" described in the protocol. However, we will not only provide basic functionalities for placing an order but also offer additional features.

3.1.1 Market Monitoring

Traders will be provided with a live indication of the current market price and historical data. The front end will have charts showing the current price developments and volume for all tradeable order pairs. Moreover, a live view of all orders submitted to the order book will be shown. That means it will not only show orders placed through our front-end but any order submitted to the MuesliSwap protocol.

3.1.2 Market Order

As explained in the protocol specifications the MuesliSwap protocol will only support limit orders. To still provide traders with the chance of submitting something similar to market orders the MuesliSwap front-end will make limit suggestions. Through limit suggestions traders get a suggested price for their limit orders. The suggested price will have a high chance of direct order matching which is very similar to performing a market order.

3.2 MuesliSwap Matchmaker

The MuesliSwap Matchmaker is an off-chain component operated by us. The MuesliSwap Matchmaker will be acting as a "matchmaker" in the protocol. The main task of the off-chain component is to find a matching through all of the submitted orders. Finding this optimal matching is similar to the task faced by centralized exchanges. That makes it possible to use existing algorithms from research for constructing the MuesliSwap matchmaker. The most known algorithm types from research are the Price/Time algorithms and the Pro-Rata algorithms. [7]

As the choice of the matching algorithm is an important part of the trading mechanism the MuesliSwap team plans to open-source the matching algorithm used. Moreover, we are planning to operate on a fair-matching basis. That means we will set up a publicly accessible monitoring tool reporting about the matching algorithm inputs and results. This makes it possible for any market participant to verify that the MuesliSwap Matchmaker is operated with the open-sourced algorithm. This is of crucial importance at the beginning where not many matchmakers are participating in the protocol.

In addition to this, to bring the game-theoretic insights from Section 2 to real use, it is planned to not only provide one matching algorithm but several. These come with different flavors especially regarding preference of best prices and batch sizes, optimally parameterized. By deploying multiple algorithms that follow different strategies, there should be as little congestion as possible while maintaining a high degree of parallelization and low waiting times for traders.

References

[1] Concurrency in the eutxo model is not a problem but a challenge . https://liberlion.medium.com/concurrency-in-the-eutxo-model-is-not-a-problem-but-a-challenge-db4b395a8eda . Accessed: 2021-10-23.

[2] Concurrency, state, & cardano . https://sundaeswap.finance/posts/concurrency-state-cardano . Accessed: 2021-10-23.

[3] Minimum ada value requirement . https://docs.cardano.org/native-tokens/minimum-ada-value-requirement . Accessed: 2021-10-24.

[4] H. Adams, N. Zinsmeister, and D. Robinson. Uniswap v2 core. 2020.

[5] V. Buterin. Ethereum white paper: A next generation smart contract & decentralized application platform. 2013.

[6] M. M. Chakravarty, J. Chapman, K. MacKenzie, O. Melkonian, M. P. Jones, and P. Wadler. The extended utxo model. In *International Conference on Financial Cryptography and Data Security*, pages 525–539. Springer, 2020.

[7] K. Janecek and M. Kabrhel. Matching algorithms of international exchanges. *url: https://pdfs. semanticscholar. org/6d92/0528fc5a3a25cb7a627b93ae3e7d5789bde8. pdf*, 2007.

[8] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *Decentralized Business Review*, page 21260, 2008.