Using an AMM to Balance a Web3 Game Economy

By Jojo Meylaerts Castle of Blackwater Published on 09/01/2023

Introduction

Over the past year, we've been wrapping our heads around the application of liquidity pools in web3 game economics. Now, after months of modelling, brainstorming and discussing our hypotheses with contacts, we finally feel confident to share this idea with the larger community, in the hopes of gathering additional insights and feedback.

The goal of this essay is to demonstrate how a liquidity pool can not only act as a thermometer for a web3 game tokenised economy, but also as an automated balancing protocol, and entirely new business model for web3 game studios.

Table of Contents

Chapter 1: Understanding the Castle of Blackwater Economy

Before we can attempt to explain the Automated Market Maker application in our game, we have to first introduce the other economical components of our project; Castle of Blackwater.

Castle of Blackwater is a 2D social deduction game, that can most easily be imagined as a blend between Among Us and Town of Salem. There's much more to it, of course, but this basic explanation of the concept will suffice for now. One thing that's important to know is that each playable character in our game has a set of unique skills (passive & active), giving them a certain playstyle that differentiates between characters. This allows us to create The Bidding System, which will end up being one of our main token sinks.

1.1 The Bidding System

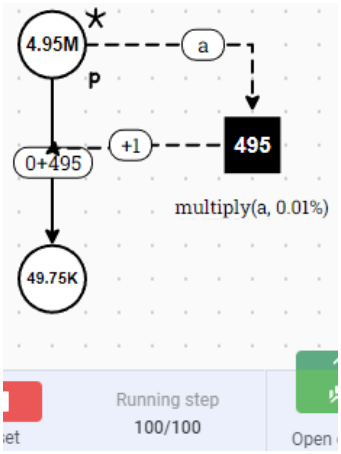
In response to a long-existing problem in web2 gaming, we created the Bidding System. If you've ever played a team-based game with different roles/characters, you'll know that the majority of the player base always has a preference for the more "fun" characters. Thereby, you'll often have to queue longer if you want to play the most popular characters, and finding good supportive players can sometimes be a struggle. To solve this, we introduced a bidding system, where players can set their loadout of favourite characters and 'bid' on which they would like to play next game. The higher you bid, the more chance you have of playing your favourite character (at a fee). If you lose your bid, it won't cost you anything, and you'll have more currency available to bid higher next round should you wish to.

With this system, we advocate for "pay-for-fun" rather than "pay-to-win". We also believe that this system creates a unique purpose for scholars (which we call "Value Extractors"). Value Extractors will play the least popular roles in exchange for a more consistent pay-out, and Value Injectors can spend more tokens to ensure they play their favourite characters. This system forms the main token sink for our utility token, but is not required for the AMM economy to function. A different game could have different tokens sinks, which are crucial, to allow players to increase their gaming experience at a cost of tokens.

1.2 Percentage-based Payouts

Without going into too much detail, as I've previously covered this concept, the %-based pay-outs protocol creates a direct correlation between value being spent by players and the earning potential per game.

Simply put, if the majority of the player base consists of value extractors, the protocol will increasingly lower game session pay-outs. As value extractors are predominantly money-motivated, they will eventually leave in search of new opportunities. Once more value starts being injected than extracted, the protocol will reverse and start increasing pay-outs once more.



When modelled out on machinations, we can see how the protocol slowly starts reducing pay-outs as the treasury starts decreasing.

(This image shows that after 100 games of the Treasury paying out %-based rewards, the rewards per game have shrunk from 500 to 495.)

1.3 Ranked / Ticket System

Another Important aspect of controlling inflation is gatekeeping the amount of value extraction that can occur. If you create a game that has an earning mechanic that is operational 24/7, you can bet you'll attract hard-core value extractors (and bots) that will come to grind your economy 24/7 for an attractive pay-out. Instead, we believe it's better to restrict the amount of earning that can take place, and allow players to increase their earning potential over time as they become better at the game.

To realise this, we did a few things. First, we separated "Casual" from "Ranked" play. Casual is our first game-mode that is based on a web2 game economy. This mode will launch first, and will be used to simulate our economic theory for our future web3 economy. The web2 game mode will only attract value injection, as extraction is simply not possible. This will allow us to balance and perfect the gameplay loop with a real player base and create an initial base of injectors and gamers enjoying the game. In the next phase, we'll introduce a "Ranked" game-mode, for which NFT's will be required to play, bidding will cost $COBS (a crypto token) and rewards will be paid out in $COBS also.

Anyone that owns an NFT can play unlimited games of ranked, however, to activate the earning mechanic they will have to use a 'ticket'. The number of tickets you own is tied to your 'rank', and as you increase your rank over time, you'll unlock more tickets to increase your earning. This way, the most competitive players will have the highest earning opportunities. Do note, that players can play ranked without the earning mechanic, simply to enjoy a higher level of play, and they can also still spend $COBS on bidding even though they're not earning.

1.4 Dual-token Economy

We've always been proponents of the dual-token economy. Mainly due to the fact that a web3 game economy has so many different stakeholders that need to be accounted for, that we found it extremely difficult to align the interests and incentives of each of these in a single token model.

A web3 game economy has a speculative side; there's investors, team token allocations, potential staking rewards, etc… Anyone interested in the speculative side wants to hold a token that focuses on value accrual. They want to see the token price rise, and preferably, the supply deflate and become scarcer. Understanding this aspect of web3 game economy design has proven to be difficult for web2 game studio's breaking into the space, as their digital assets have never really been speculated on to this degree. But merely understanding this desire for speculation will only get you so far, and embracing this speculative side can actually unlock new business models and exciting new revenue streams for web3 game studios.

The other side of the coin (pun intended), lies on the 'Medium of Exchange' (MoE) side. Pretty much every game has some kind of currency that is intended to be spent or transacted with. A currency that is highly liquid, has high volume, preferably some guarantee of price stability, and of course utility. As a game economy designer, our priority is not to create a MoE token that people buy and hold as a speculative asset… On the contrary, we're designing it for the players to spend in exchange for game assets, unlocks and priorities! The best way to incentivise spending is through controlled and moderate inflation, which is contrary to what we expect from a speculative asset.

We believe a dual-token economy is the best way to separate the different incentives and ensure each asset can focus on its purpose.

1.4.1 “Value-accrual” token

Our value-accrual token is called $COBE. It is the token that we sell to investors, that we hold as a team, give to advisors, etc… Around half of the supply will be reserved for ‘ecosystem’ rewards, such as tournaments, staking, marketing, etc…

Our main goal for this token will be for the value to represent the growth of our success. We aim to do this by:

- Using cliffs and vesting to ensure a slow & steady circulating supply growth
- Using company profit to buyback excess supply and either burn it (deflation) or add it as staking rewards (redistribution)
- Provide a staking pool with rewards to minimise token dumping, and instead incentivise holding
- Selling premium/speculative in-game assets (like LAND) for $COBE, to create buy pressure and utility

By having this separation of tokens, we encourage all speculators to speculate on this asset. As such, we could pre-sell some of its supply to investors to raise start-up capital.

1.4.2 “Utility” token

Because we’ve been able to cater to the speculators through our other token, we’ve given ourselves a lot of freedom and flexibility to control the supply of our utility token. By no longer having to worry about investor, advisor or team allocations, we can maintain complete control of the utility token supply, which will be crucial in our attempts to balance it.

Our utility token will be called $COBS, and before launch we have control over 100% of it’s token supply. We’ll cap the total supply at 500,000,000, but we never expect the full supply to enter circulation at any time, so you may also consider it uncapped to some degree.

For simplicity’s sake, let’s imagine the tokenomics to look something like this:

Tokenomics	$COBS (Spend token)	%
COB Reserve	485,000,000	97%
Liquidity Pool (IDO)	10,000,000	2%
Treasury	5,000,000	1%

Seems very basic, right? But remember that the priority here is to have as much control over the supply as possible.

1.5 Supply VS Demand

The reason it is designed this way is because supply is much easier to control than demand. Furthermore, it’s much easier to increase supply than decrease it. This is a key factor to remember moving forward.

The demand side is much harder to control. Sure, we can do marketing and forecasting, but demand growth is not as guaranteed and predictable as supply growth can be. Trying to model based on pre-defined demand growth predictions can quickly lead to disappointment and loss of control.

The conclusion we drew from these realisations is that demand should be a leading indicator, and supply should be the controlling stabilizer. Create many data points that help you track demand in real-time, and then use your economic supply levers to match it. The goal is to slowly introduce more supply as demand grows, but also be able to contract supply if demand reduces temporarily.

Chapter 2: The LP Web3 Game Economy

2.1 What is an AMM

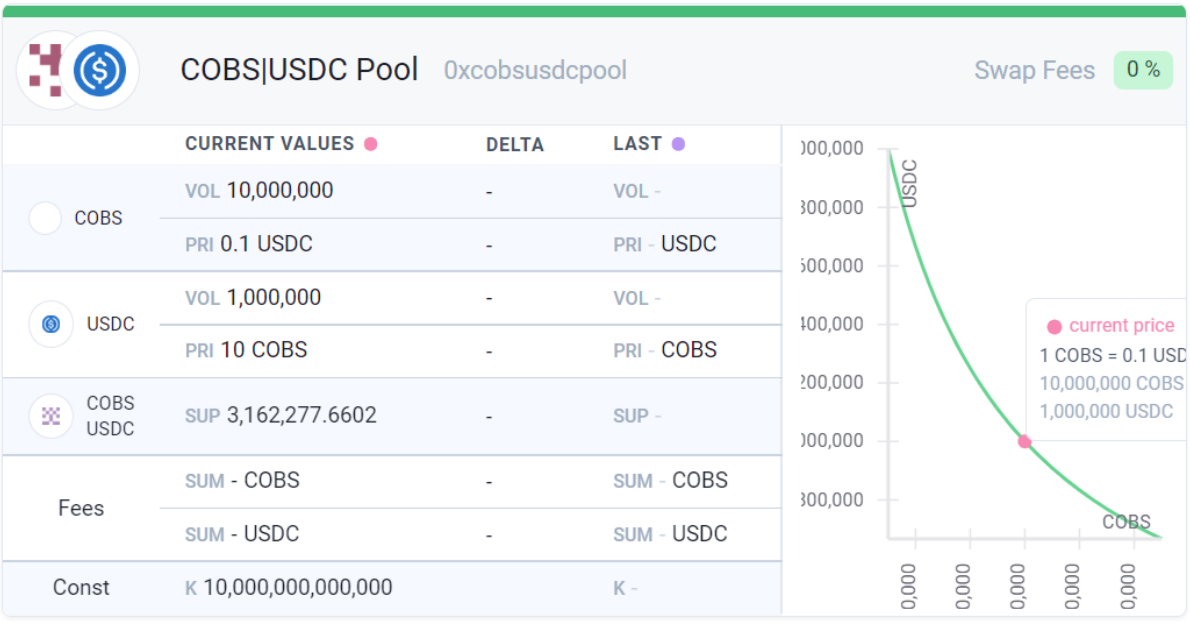
AMM stands for Automated Market Maker, but is sometimes also referred to as a Liquidity Pool. First ideated in 2017 by Vitalik Buterin, the concept really took hold when implemented by Uniswap in the world's first decentralised exchange.

For an ELI5 explanation of AMM's, check out this video:
https://www.youtube.com/watch?v=1PbZMudPP5E

What makes AMM's so interesting is that they mathematically enforce a balance between two assets, based on supply and demand, in relation to each other. This self-balancing system can work really well in helping us find the balance between value injectors and value extractors.

2.2 Setting up the AMM

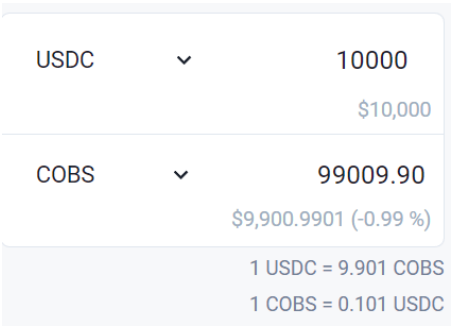
Let's simulate setting up an AMM for our utility token ($COBS), based on the above mentioned tokenomics. We'll take 2% of our token supply (10M tokens), add $1M worth of stablecoin (USDC), and create a liquidity pool (LP) based on the conversion that 1 $COBS = 0.10 $USDC (or 10$COBS = 1 $USDC). We'll use a simulator (https://amm-playground.on.fleek.co/) to support this experiment.



After depositing these funds into the pool, we've set the initial conversion price. Anyone that owns $COBS would be able to trade them with this pool in exchange for $USDC, and on the flipside, anyone that owns $USDC can trade it for $COBS accordingly. For the purposes of this demonstration, we've set the swap fees at 0%. However, in a real-life scenario we would likely have a small swap fee percentage as an extra revenue generator.

Because initially we control the entire supply of $COBS, there are no $COBS in circulation for anyone to withdraw our $USDC. Thus, for now, only people with $USDC can swap it for $COBS.

If at this stage anyone swaps $USDC for $COBS, the conversion price will recalibrate to match the supply change. Imagine someone swapping 10,000 $USDC, they will receive 9,9009.9 $COBS. The reason they don't receive 10K $COBS is due to "price impact", as the trading pair does not have a huge amount of liquidity, which means this single transaction will impact the conversion price by 0.99%.



2.3 AMM: an economic thermometer

An important thing to take away from this, is that as long as the LP conversion rate is equal to 10 $COBS = 1 $USDC (10:1), then the economy is in equilibrium, meaning there is an equal amount of value injected into the system as value that has been extracted.

Therefore, the higher the conversion rate goes, the more value has been injected into the LP ($COBS bought) than has been extracted ($COBS sold). In this way, we can now use the LP conversion rate as a thermometer, to give us a real-time number of value inflow vs outflow, and consequently, we can change our economic policy to respond to this. To simplify this concept using examples:

LP conversion = 10:1 (economic equilibrium) - Stable
LP conversion = 10:0.7 (value extraction) – Bad
LP conversion = 10:1.3 (value injection) – Good

2.4 Token flow

There are three ways to receive $COBS tokens; you can earn them by playing, you can 'buy' them from the LP, or someone else can gift/trade them to you.

Once you have $COBS token, there are four things you can do with them; you can hold them in your wallet, you can spend them in the game, you can gift/trade them with another player or you can 'sell' them to the LP.

Let's remove gifting/trading and holding from the equation for a second, as these events don't have any effect on the balance between value injection or extraction, and focus instead on the remaining options;

You earn or buy tokens, and then eventually you spend or sell them.

Simply earning tokens does not make one a value extractor, as those tokens can still be spent to remain in the economy. Similarly, merely buying tokens does not make one a value injector, as those bought tokens could be sold again the next day.

It is the combination of actions that determines the impact of the token flow.

As such:

	Earn	**Buy**
Spend	No impact	Value Injection
Sell	Value extraction	No impact

It's important to differentiate these token flows, and the impact they have on the economy, when using the LP as a thermometer. Looking at this table, we now understand that we always need a double confirmation before we can define economic policy, so as not to react to singular events that could result in a 'No Impact' outcome.

An example to demonstrate why that is, goes as follows:

Imagine an individual swaps $100K worth of $USDC for $COBS tokens. Due to price impact, they would receive only 909,090 $COBS. This single transaction would also change the LP conversion from 10:1 to 10:1.121 (1 $COBS = 0.121 $USDC).

Using the LP thermometer rule described above, one would now assume that because the LP conversion > 10:1, there must be more value injection that extraction. In theory, this would be correct, as there is indeed more USDC injected in the economy than has been extracted. However, until those $COBS are actually spent in the economy sinks, we don't yet consider this a net positive outcome. That player could turn around and swap back all of their $COBS at any time, so as long as the player doesn't SPEND them, we don't consider the transaction as 'revenue'.

Once $COBS tokens start being directed into the token sinks, and back into our company wallets, that's the point where we look at the LP conversion thermometer to decide what we do with those collected $COBS.

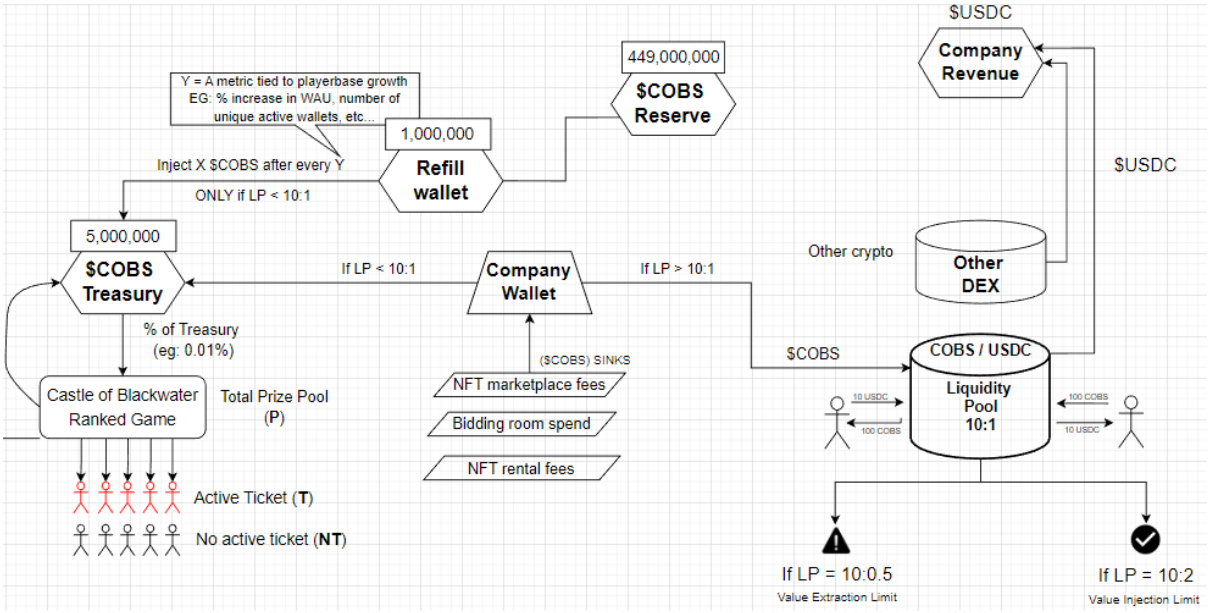
If at that point the LP conversion is still in our favour, we direct those $COBS back into the LP in exchange for $USDC, which brings the conversion back closer towards 10:1, and the $USDC we gain can be treated as revenue.

On the other hand, when the LP conversion is below 10:1, any tokens that we collect from token sinks are directed back into the rewards treasury, as these tokens were likely earnt from there in the first place, and not bought by value injectors.

Using this methodology, we can distinguish to what degree token inflow comes from rewards earnt by players (no impact) or from real value injection (tokens bought and spent). We're only converting token inflow to actual revenue when the economy is running at a value injection surplus.

2.5 Tying it all together

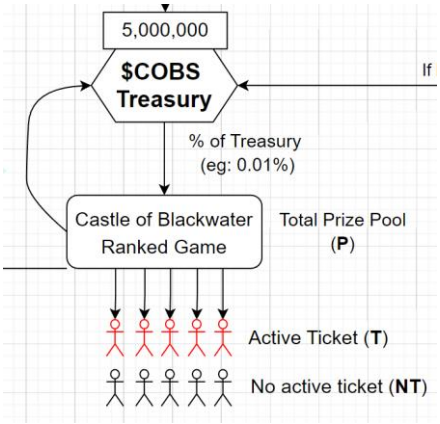
Here's a visual to demonstrate the whole $COBS AMM balancer model in action:



If you've read and understood everything so far, you should have a pretty good understanding of how this system is starting to operate.

To summarise the model, we'll quickly run through the individual parts one more time, and add some more context to aspects that haven't yet been explained.

2.5.1 Minimise Value Extraction



This section demonstrates the initial treasury that will be filled up with 5M $COBS tokens. The %-based pay-out protocol and the ticket system minimise value extraction.

The first 'ranked' game played will generate 0.01% of 5M $COBS tokens (500). Depending on the game outcome, these will be allocated at the end of the game to all participating players. Players that have an Active Ticket will receive their tokens, those that don't will instead receive an off-chain currency alternative to be used in the casual game mode. Any $COBS tokens that do not get paid out are redirected to the Treasury.

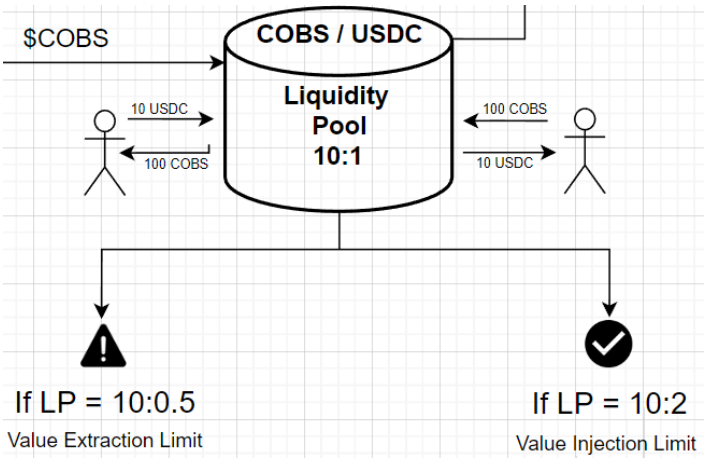
A very rough initial calculation suggested that around 2,000 games need to be played before the Treasury pays out 1M tokens according to the current set-up.

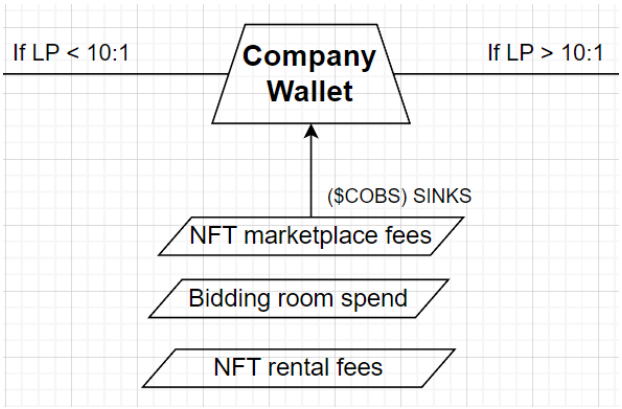
2.5.2 AMM Balancer

This section showcases the AMM Liquidity Pool, which we use as a thermometer to measure the level of value injection vs value extraction.

This pool will be loaded with 10M $COBS tokens and 1M $USDC, to set a baseline conversion of 10:1.

Below the pool, you'll see two boundaries that we can set, which create a maximum range of acceptable volatility that we'll allow our utility token to sustain. Tighter limits mean less price volatility and reduced risk, at the expense of lower potential market making revenues.



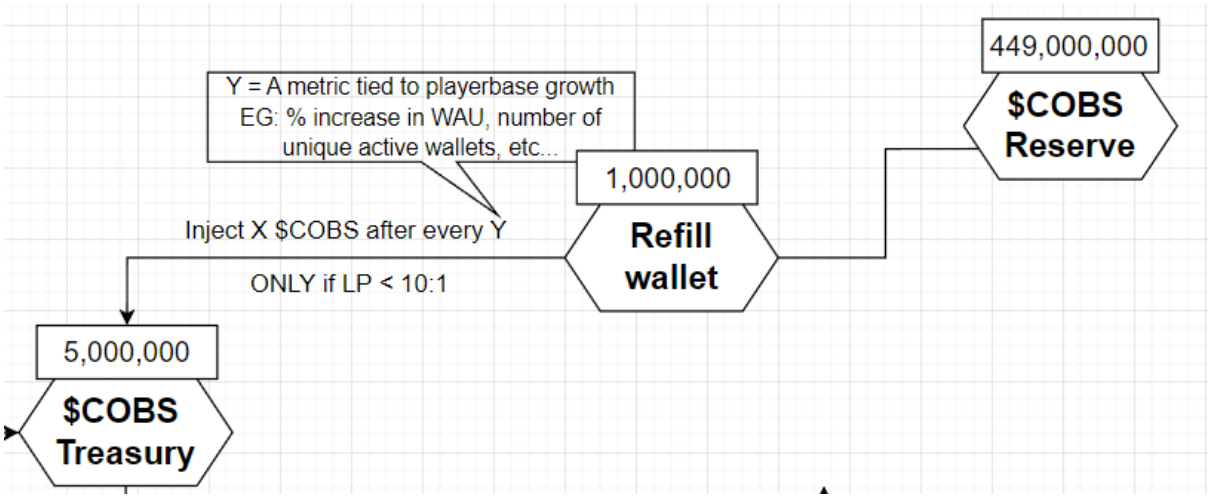


2.5.3 Double confirmation

When $COBS tokens are spent in our token sinks, they find their way into the "Company Wallet". When the LP thermometer gives us the green light, the $COBS tokens are directed into the LP and exchanged for $USDC. This will happen as long as the LP reads higher than 10:1.

As soon as the LP conversion goes below 10:1, the $COBS tokens are instead redirected back to the Treasury wallet.

2.5.4 Reserve Refill



The reserve holds the vast majority of $COBS tokens. It serves to provide extra supply to control deflationary pressures, as well as sporadically refill the treasury to account for player growth.

As the player-base grows, the number of players holding a small amount of idle $COBS in their wallets will increase. To account for this, we will occasionally have to inject more tokens into the treasury, as long as the LP thermometer shows positive value injection and we can establish that the tokens are being held in small amounts by new players, rather than single whales trying to exploit the system.

The exact method of $COBS injection from the Reserve to the Treasury to account for player growth is an area that needs more research and testing, however it should be a metric that can be automated without being easily exploitable.

Ideally, the refilling of the Treasury should be an automated process as much as possible, to reduce the risk of human error and foul-play interfering with the fair distribution of tokens. However, to safe-guard against any potential extreme exploitation of an automated system, we will use a refill wallet that is manually refilled from the Reserve, so that in an event of an exploit the potential losses can be mitigated.

2.6 The launch

Prior to the 'ranked' game mode launching, and the token earning mechanic ever being functional, we would have reached a lot of important milestones.

First, we will have launched a web2 version of the game, based on a more 'casual' game mode, that involves off-chain currency and character assets. We will grow an initial player base that enjoys the game, and ensure the gameplay loop is balanced and functional.

Only when we feel confident that we have achieved moderate success as a web2 game experience, will we feel confident in taking the next step in creating a web3 version of this experience. The goal of the web3 'ranked' game mode, will be to offer a more competitive scene, where players can really put their skills to the test, and take part in a more decentralised ecosystem.

Once we launch the Liquidity Pool and the ranked game-mode, we can imagine different scenarios to occur, with different outcomes that need to be accounted for. The following scenarios are estimations based on possible token flows. Note that they include some very rough estimations regarding percentage distributions, and that they should merely give a general idea of possible outcomes.

A quick python script simulation of the %-based pay-outs protocol, calculates that it will take 2232 game sessions to generate over 1,000,000 $COBS tokens.

```
treasury = 5000000
percentage = 0.0001
payouts = []

for i in range(2232):
    payout = (treasury - sum(payouts)) * percentage
    payouts.append(payout)

print("Sum of payouts:", sum(payouts))
```

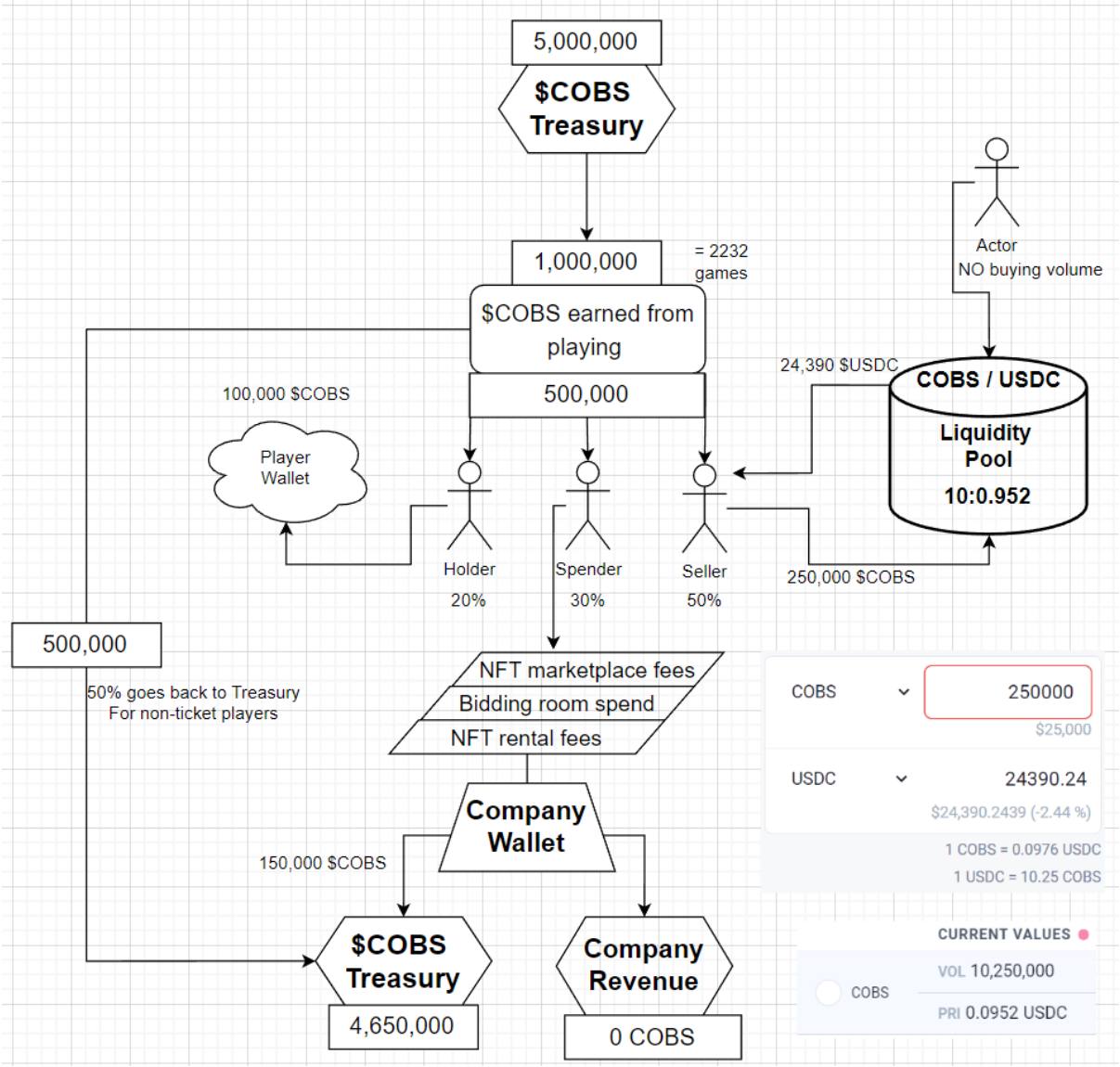
```
Sum of payouts: 1000270.4285775223
```

Therefore, in the following scenario simulations, each scenario will be based on roughly 2,232 games being played and a simulated 1M $COBS tokens being paid out. We'll also assume that in every game about 50% of players have an active ticket to unlock the earning, and the other 50% don't, meaning that half of the token pay-outs will automatically be redirected back to the Treasury.

Chapter 3: Scenario Simulations

3.1 Only Value Extraction

Before we calculate the maximum possible risk, let's first examine a more realistic but still pretty gloomy scenario:



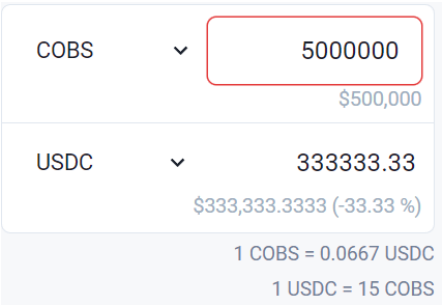
Here we simulate what would happen if 50% of all tokens earnt are then extracted from the economy by swapping them for USDC in the LP, whilst simultaneously no USDC is injected into the LP. We can see how this extraction removes just under 25K of USDC liquidity from the pool, at an average price of 0.0976 USDC, which brings the subsequent LP conversion down to 10:0.952.

As the LP conversion is lower than 10:1, thereby we see more extraction than injection, any $COBS tokens spent on token sinks will be directed back to the Treasury.

In this scenario, the company makes no revenue and 'loses' 25K USDC from the pool. The Treasury gets reduced, without being refilled, and game pay-outs are lowered from 500 to 465.

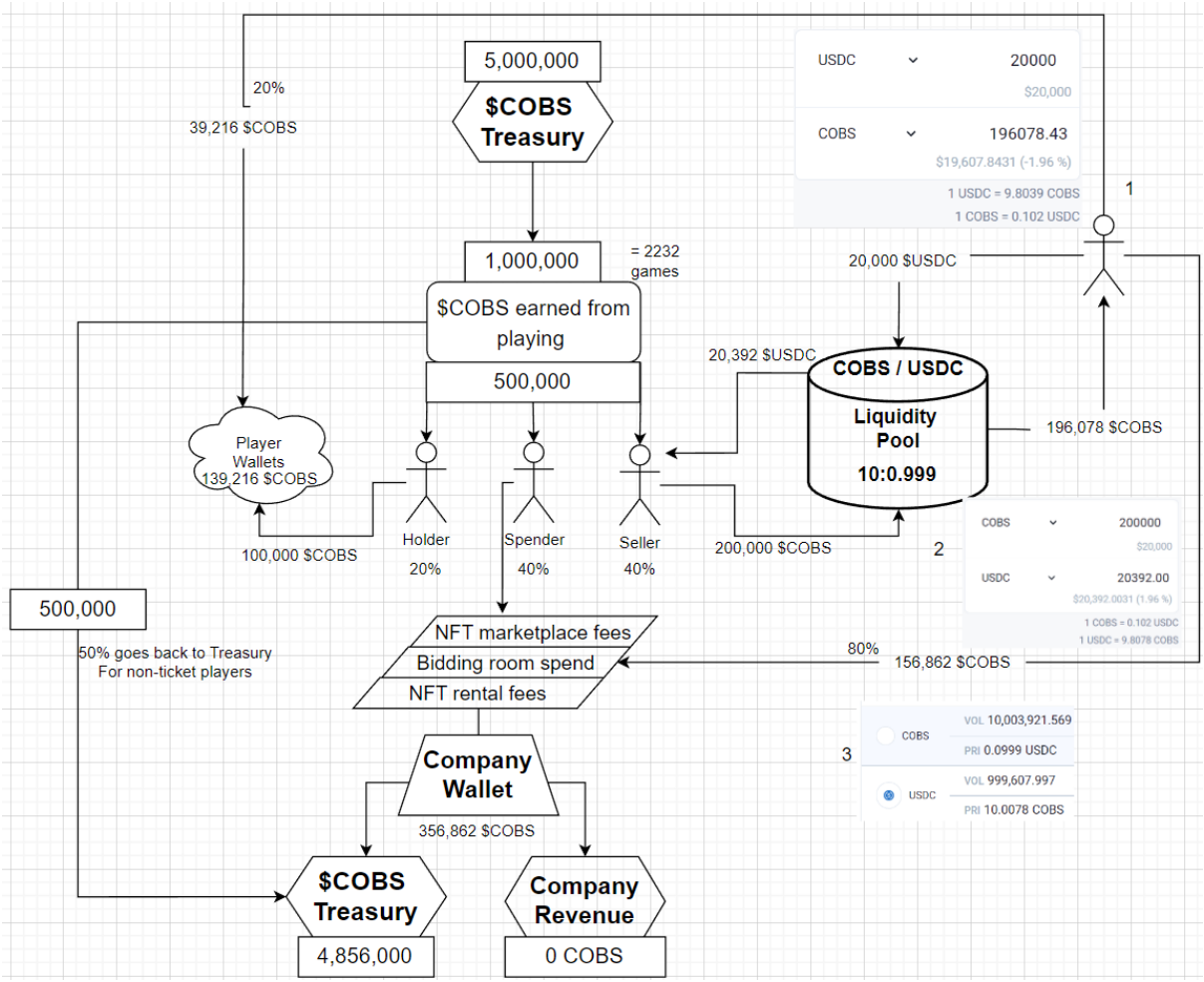
At the end of this simulation, the Treasury is missing 350,000 $COBS from its original state. 100,000 of these are held in wallets, and 250,000 of them are oversupplied to the liquidity pool. This means that game rewards will remain lower until the excess tokens are bought back from the LP and spent in the tokens sinks once more.

Should this trend persist, we can simulate a worst-case scenario. Although hypothetically impossible (due to the exponentially decreasing nature of the %-based pay-out protocol), let's imagine the entire Treasury get drained of its 5M tokens and all of those get 'sold' into the LP. Using our LP simulator, we can calculate that the maximum total loss in this case would be 333,333 USDC. After this, there is practically no more circulating supply of $COBS available.



3.2 Value extraction = Value injection

Here we see a prediction of a situation where value injection is equal to value extraction. Due to our limiting systems of value extraction, we'll run the calculations on the assumption that value injection occurs before extraction.



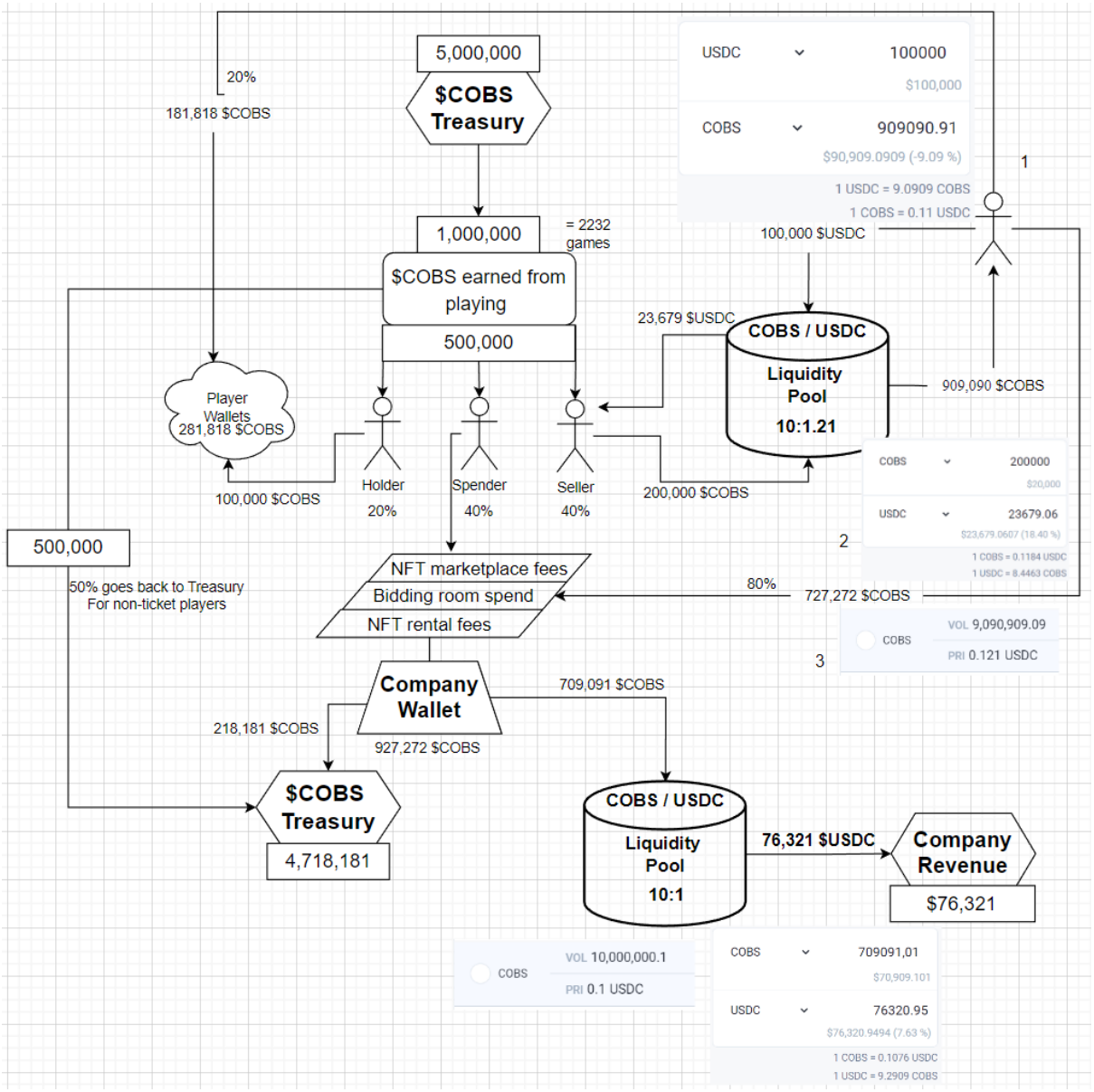
The simulation shows that in this scenario the LP conversion thermometer does a good job at showing that there is a balance between injection and extraction. As the LP conversion is at (or just below) 10:1, the funds coming in from the token sinks are not redirected to the company revenue.

The result of this simulation is that the LP loses about 400 $USDC of value, mainly due to the order of the conversion that took place. The company makes no revenue from this scenario, and any potential losses are negligible.

The Treasury has reduced slightly, as a portion of tokens are being held idle in wallets.
Should these tokens be spent, they will be redirected to the Treasury (refilling it to 5Mil). Should they be sold, game rewards will remain lower until more value is injected and equilibrium is restored once more.

3.3 Value Injection > Value Extraction

At this rate, the model really starts to show its effectiveness. At first, we simulate that 100,000 $USDC is swapped into the LP for $COBS. Assuming this is done by real players, we'll imagine 80% of those $COBS find their way into the token sinks, whereas 20% are held wallets.

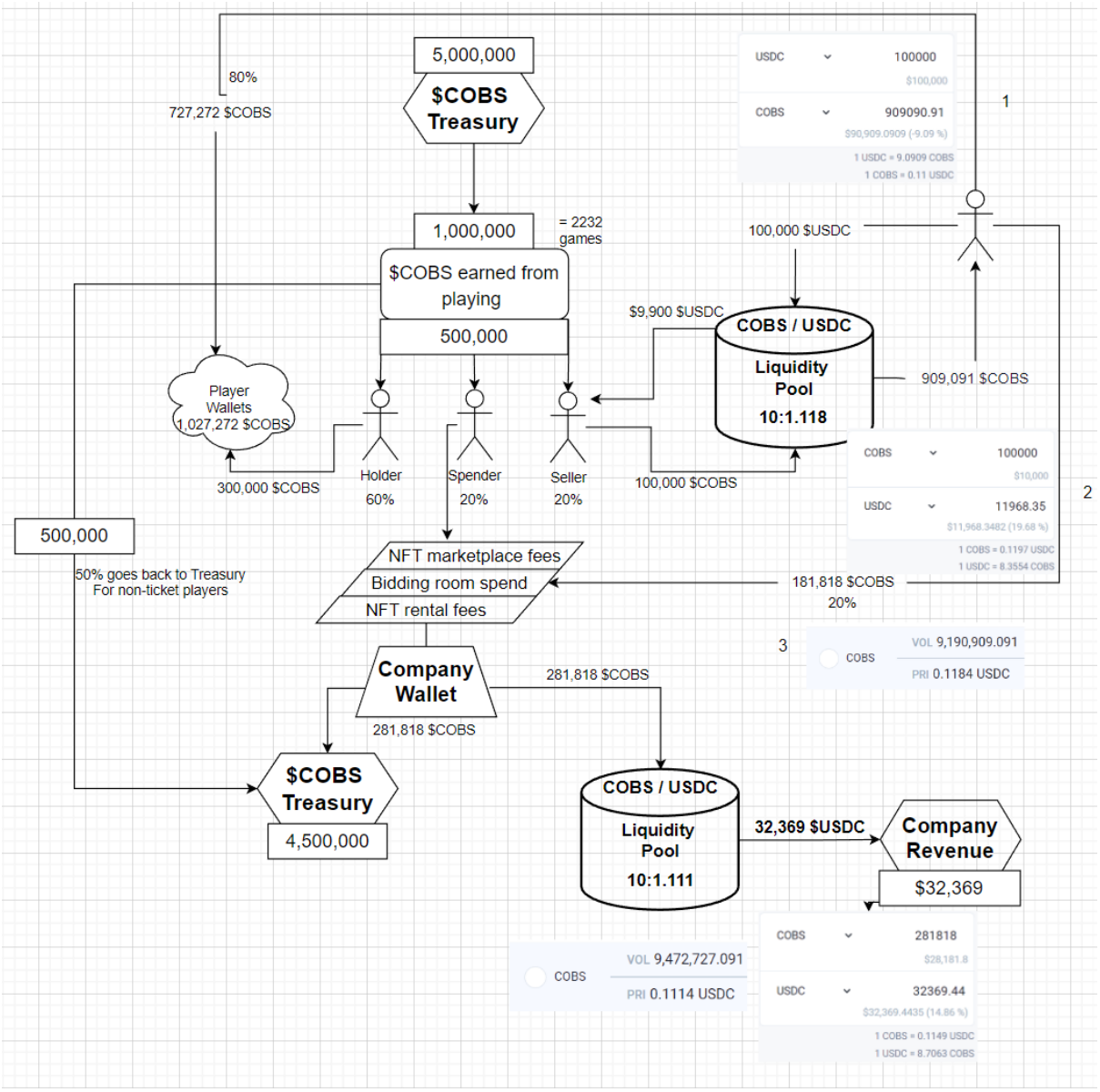


Through this simulation, we see that 100,000 $USDC is injected into the LP, and 23,679 $USDC is extracted. This leaves a surplus of 76,321 $USDC. As most of the tokens are being spent on the token sinks, we see them be collected back in the company wallet. At that point, we look at the LP conversion to see it positive, and are able to use the $COBS we have collected to bring the LP back to its equilibrium (10:1) and claim some revenue. Whatever $COBS are left over will be directed back into the Treasury.

The difference between the starting Treasury balance and the final one, can once again be found by the tokens being held idle in player wallets. If those are spent, the Treasury will return to 5M, if they are sold, game rewards will remain lower until more value is injected and equilibrium is restored.

3.4 Asset Speculator Scenario

The following scenario demonstrates how the model is built to protect again speculators. Speculators, in this context, refers to individuals purchasing the token not to spend on the game, but to speculate on it's potential future increase in value. As it is our goal to create stability and reduce volatility for our MoE token, speculators provide no tangible value to achieve this. Therefore, the system actively disincentivises them from doing so.



By the end of this simulation, idle player wallets are holding over 1M $COBS tokens. A lot more value has been injected into the LP, which is shown by the LP conversion thermometer (10:1.118), yet the company wallet doesn't reflect this yet. This means that the majority of those tokens are sitting idle in a wallet, and we can't predict if they will be sold again or if they will get spent.

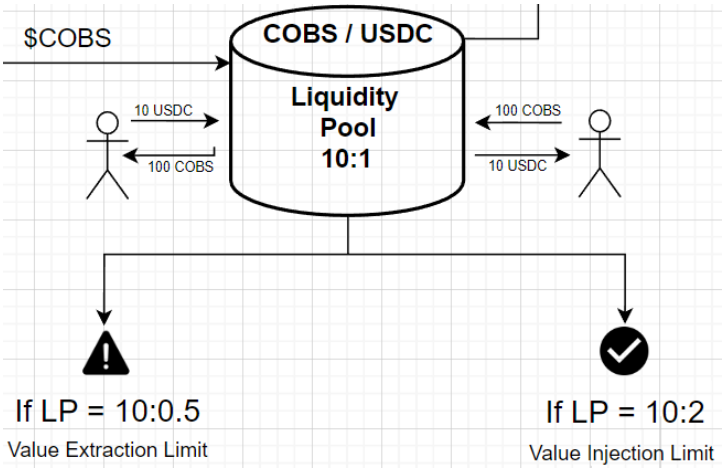
To counteract this, we run the system as intended, and because the LP conversion is positive, we swap all token sink proceeds for USDC and treat it as revenue. This leaves the Treasury sitting at a reduced 4.5M, meaning the speculators have caused the game rewards to decrease.

As we're constantly using token sink proceeds to bring the LP conversion price back down to equilibrium, there isn't much profitability in buying and holding the MoE token in the hopes it will appreciate in value.

3.5 Protective Pricing Pegs

In order to protect against extreme price manipulation / speculation, we would create an upper and lower price boundary, to create a maximum range of volatility for our MoE token.

Looking at the previously suggested boundaries, we would place protections to ensure that the $COBS token price should never go lower than half, or higher than double its initial value.



Starting from 10:1, for the lower end this would mean a value extraction surplus of around 4,150,000 $COBS tokens.

For the higher boundary, starting from equilibrium, this would be a value injection surplus of 415,000 $USDC

It is highly unlikely that we ever hit the lower boundary before we first hit the higher one, simply because this would require more than 4M $COBS tokens to be earnt and sold. Just accounting for the %-based pay-outs protocol, this would require over 16,000 ranked games to be played, where the whole lobby has an active earning ticket. At this point, a single game would only pay-out 100 $COBS per game.

```
treasury = 5000000
percentage = 0.0001
payouts = []

for i in range(16094):
    payout = (treasury - sum(payouts)) * percentage
    payouts.append(payout)

print("Sum of payouts:", sum(payouts))
```

```
Sum of payouts: 4000042.562025184
```

Now take into account the ticket system, which limits the amount of earning a single player can do based on their rank, and further reduces the value drain even more.

Ideally, for the protective pricing pegs to work as intended, we would first want to see value injection to the point where the upper boundary is hit. When the LP conversion goes above 10:2, and large portions of $COBS tokens are held in player wallets, we can start to increase the circulating supply by swapping some of the $COBS tokens we keep stored in the Reserve. We would only swap enough $COBS to keep the LP conversion from going higher than 10:2, and all the tokens we would be swapping would be sold at a premium. The USDC gained from these swaps would be stored in a separate wallet, “Strategic Cash Reserves”.

Considering the size of the $COBS Reserve, we could hypothetically keep this up indefinitely, however at some point the demand should slow down again until the free market supply/demand can once again stabilise below 10:2. By taking this action we have increased the circulating $COBS supply beyond what our initial model was balanced for. Therefore, it will be much easier to get down to the lower boundary again should there suddenly be a lot of sell pressure.

However, as long as we hold on to our “Strategic Cash Reserves”, that we raised by selling excess $COBS supply at a premium price, we should in theory always have sufficient USDC reserves to buy back the entire circulating $COBS supply at a discount (10:0.5). In essence, by enforcing these boundaries, we become a market maker of our own economy.

Conclusion

The ideas explained in this paper are the result of over a year of research, modelling, iteration and feedbacking. We could not have come to these conclusions without the constructive criticism from so many of our web3 frens, and we are very grateful to have access to such an incredibly diverse and supportive network that have the patience to dive deep into these complex subjects with us.

Although the model is far from finished, we've reached a stage where we feel it necessary to publicly share our progress within our communities, with the hope that others may add to and learn from these early ideas we present today.

Although Automated Market Makers have only recently been invented, we've already seen them create radical change in the finance markets by kickstarting the rise of DeFi. We believe that there are many more applications of this new technology, and are determined to showcase how their naturally balancing mechanisms can be useful in the inherently unstable web3 game economics we have thus far seen in "play to earn" gaming.

We welcome any and all questions, comments and feedback, as we aim to perfect the model and have it ready for testing in a real-world environment. We're also eager to explore any possibilities to improve the rather static and simple simulations we have run in this research phase, and hope to have a more dynamic and real-time simulation model before we launch.

And finally, we appreciate you taking the time to learn more about our vision on web3 game economics, and would like to promote all other web3 game builders to share their models with us too.

Areas for Further Research

The following areas have been determined for further research, to be conducted before the model should be launched publicly:

- More refined/dynamic modelling with real-time simulations (eg: Machinations)
- Which time-cycle would be most appropriate for the balancing processes to occur in this economy? (real-time/daily/weekly/seasonal/etc..)
- How would be the impact of launching the LP a week before launching the game mode?
    - Forced value injection? Over-speculation?
- What would be the impact of having some other forms of initial supply distribution?
    - Some advisory/team/marketing token allocations
- What would be the impact of the token also being launched on other DEX's or CEX's aside from just the LP?
- Conduct market research among guilds/scholars to find out at what level of daily earnings they migrate to other web3 games
- Gather data from live web3 games about what percentages of their player base are categorised as token spenders, holders or sellers
- How would the model hold up under extreme price manipulation events / exploits