Algorithmic (Crypto) Finance

Prof. David Tran, PhD
duc.tran@umb.edu
University of Massachusetts

1

Decentralized Finance (DeFi)

- **DeFi** = financial services on the blockchain
  - Lending, Borrowing, Saving, Earning
- **MakerDAO:** the first DeFi app (2017) – Ethereum-based protocol
  - Allow anyone to take out a loan without relying on a centralized entity
  - Borrow cryptocurrency (**DAI stablecoin**) by collateralizing digital assets (**ETH**)
- **Compound Finance** (2018): a decentralized marketplace serving **borrowers** of collateralized loans and **lenders** who earn interests from borrowers
- **Uniswap** (2018): a decentralized **exchange** to swap between ETH tokens
- Many others: Curve Finance, Balancer, etc.

2

Outline

- Theme: how to provide Financial Services in a decentralized manner?
- Topics
  - Bonding Curve:
    - How to **set an asset's price** automatically with an algorithm
  - Automated Market Making
    - How **exchange assets** automatically with an algorithm without an order book?
  - Algorithmic Stablecoin
    - How to **create a stablecoin** that is backed automatically with an algorithm
- We have only 1 hour, so I cannot talk much into detail. I will provide **basic knowledge** and **potential ideas** to investigate

3

Bonding Curve

Buy/sell a token; Token price depends on supply. How do you set the token price automatically to reflect this so we never run out of liquidity?

4

Continuous Token Model

USE CASE

- An ecosystem to make capital gain (e.g., lending pool, mutual fund)
- Investors **buy tokens** for **membership stake** (hoping to capitalize from fund performance)

- Tokens are newly **minted** upon a membership purchase, at a price dynamically set
- **Inflation** control: the more tokens minted, the more expensive

What is the token price at current time?

5

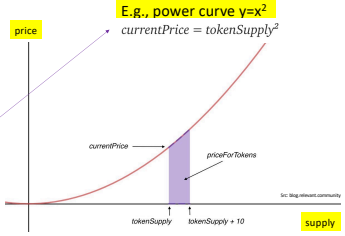
Bonding curve

- Token price is set by math based on a curve, called "**bonding curve**"; e.g.,

currentPrice = tokenSupply²

- **tokenSupply** = #tokens minted
  - A token is minted when somebody buys
  - A token is burned when somebody redeems it

E.g., power curve y=x²



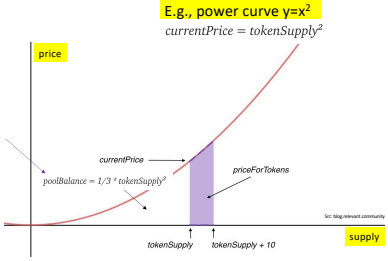
2018: Originally used in Bancor Protocol

6

Pool balance

- The cash amount tokens minted so far
  - integral of curve function ($x^3/3$)
- Suppose someone is buying 10 tokens
- New *tokenSupply +10*
- New *pool balance = (tokenSupply+10)^3/3*
- The total cost will be

E.g., power curve $y=x^2$



$$priceForTokens = 1/3 * (tokenSupply + 10)^3 - poolBalance$$

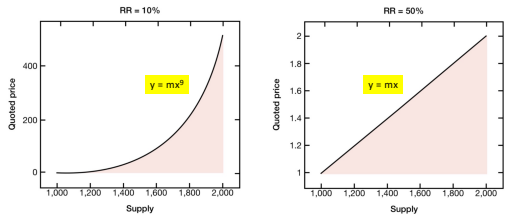
7

Power-function Bonding Curve: Reserve Ratio

- Bonding curve $y = x^p$ i.e., $currentPrice = tokenSupply^p$
  → $poolBalance = 1/(p+1) * tokenSupply^{p+1}$
- **Market cap**: $currentPrice * tokenSupply = tokenSupply^{p+1}$
- **Reserve ratio:** always a constant **1/(p+1)** → if we maintain this ratio we will never run out of liquidity. Every sell/buy will be fullfilled
  $reserveRatio = poolBalance / marketCap = 1/(p+1)$

8

Higher reserve ratio → less price sensitivity



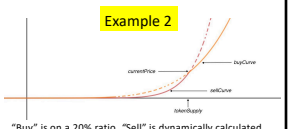
9

Buy/Sell Curves & Dynamic Reserve Ratio

- A smart contract to run a **liquidity pool** automatically (similar to AMM)
- Design separate **buy curve** and **sell curve** with **dynamic reserve ratio** to make profit or avoid pump-n-dump

Example 1

Example 2



"Buy" is on a 20% ratio. "Sell" is dynamically calculated from actual reserve ratio but always keeps 10+%

Src: Bancor Protocol: Relaunching Ethereum's Liquidity Network
Src: blog.relevant.community

10

Automated Market Maker (AMM)

Swap tokens: How can we automatically swap tokens on an exchange without order book, yet never running out of liquidity?

11

Currency Exchange

- How it works:
  - **Order book** = bids (from buyers) + asks (from sellers)
  - **Trade** executed upon a **price match** between a bid and an ask
  - That price becomes the asset's **market price**
- If few offers → **trading** is **not 24/7** and highly **volatile**
- Hence, need **market makers!**
  - **Market maker**: provide liquidity to the order book, ensuring sufficient offers to fulfill trades instantly and seamlessly
  - Usually, market makers = **large banks** or financial institutions

12

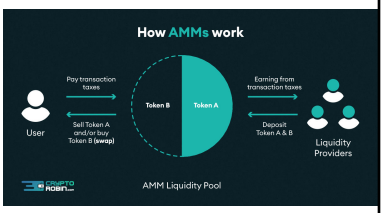
Market Makers

- Traditional exchanges:
  - **Trading is not always 24/7** and subject to volatility
  - Traders **can see** the order book and manipulate prices
  - **Market makers** are **centralized** → have to **trust** the owner of the exchange
- Automated market makers (AMMs): trading **24/7** without permission and **automatically** by using a liquidity pool managed by an algorithm, never running out of liquidity

13

Automated Market Makers (AMM)

- AMM works **beautifully** for DEX
  - Lack of order book, not enough people to match trades
- Create a **liquidity pool =** a pot of tokens
  - incentivize people to lend tokens to the pool (hence called **liquidity provider**)
- Traders **trade** with this pool
- **Token prices** are set by **math** based on the pool's current reserves



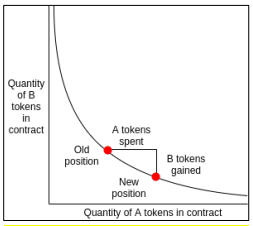
14

Constant-Function AMM

- Assume **n**=2 assets: A and B
- **Pool** = x tokens of A, y tokens of B
- The reserves in the pool **at any time** must satisfy a condition to reflect that the price of A versus B should increase if A is rarer than B in the pool
- Most often used: **F(x, y) = constant**
- E.g, **F(x, y) = x*y = c** (used in Uniswap)
- To sell A to get B, the **price** 1 A = **p** B is chosen such that

  (x + m) * (y – m * p) = c

(here, m is the number of token A sold)

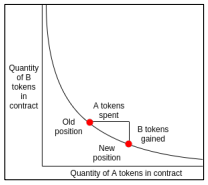


In theory, any monotonic decreasing curve can be used

15

Constant-Function AMM: **Example**

- Consider pair (ETH, USDC), 1 **ETH = 2000 $USDC**
- **Constant Product Model**: **x * y** = **c**
- To begin, pool = **same** $amount of ETH = $amount of USDC; e.g., x=**5** ETH and y=**10,000** USDC
  - i.e., $10,000 of ETH and $10,000 of USDC, respectively
- So **c** = 5 * 10,000 = 50,000
- Trader wants to buy 3 ETH for USDC: **price** 1 ETH = **p** USDC. What is **p**?
- We need to keep (x-3)(y+3*p) = c = 50,000
- (5-3)(10,000+3p)=50,000
- So, price **p = 5000:** 1 ETH = 5000 USDC



16

What if the market price is very different?

- **Arbitrageurs** will see the opportunity to profit
  - buy cheap on a market exchange to sell high on the Pool
  - buy cheap on the Pool to sell high on a market exchange
- Each such activity will make the pool price to **converge** to market price

Let's say, hypothetically, when you enter the liquidity pool, ETH is worth $100 USD:

- The price of 1 ETH = 100 DAI
- The pool consists of 10 ETH and 1000 DAI

Then, the price of ETH doubles to $200:

- The price of 1 ETH = 200 DAI
- Arbitrageurs from outside of Uniswap come in and buy all the ETH in your pool until the price reaches 200 DAI and matches external exchanges

17

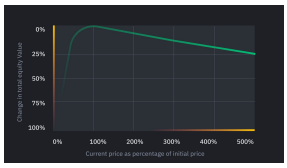
Disadvantage: Slippage

- Selling **m** token A to get **m*p** token B
- **p**: price of this trade (1 A = **p** B)
- **(x + m) * (y – m * p) = k**
- **Therefore, the higher m → the higher p**
- **Slippage** = slippage in token price from the time you want to make an order to the time the order actually takes place

18

Disadvantage: Impermanent Loss

- LP receives **less $value** at withdrawal than at the time of deposit, **compared** to just holding the assets **outside** the pool
- The **wider** the price change, the **higher** the loss
- **Stablecoin** pool: small loss because price does not change much



Impermanent loss for a given change in price ratio k

$$\mathrm{IL}(k) = \frac{2\sqrt{k}}{1+k} - 1$$

19

Impermanent Loss Formula

- Initial liquidity pool (x of asset X and y of asset Y): x*y=L²
- Initial price (at time of providing liquidity): 1 X = P Y
- **Initial holding value** $V_0 = y \cdot 1 + x \cdot P = 2L\sqrt{P}$
- Price of X at time of **withdrawing**: P' = P*k
- Value of holding if **withdrawn** $V_1 = 2L\sqrt{P'} = 2L\sqrt{Pk}$
- Current value if **holding outside** the pool $V_{\text{held}} = y + xP' = L\sqrt{P}(1+k)$
- **Impermanent loss** $\mathrm{IL}(k) = \frac{V_1 - V_{\text{held}}}{V_{\text{held}}} = \frac{L\sqrt{P}(2\sqrt{k} - 1 - k)}{L\sqrt{P}(1+k)} = \frac{2\sqrt{k}}{1+k} - 1$

$$y = L\sqrt{P}$$
$$x = \frac{L}{\sqrt{P}}$$

20

Example

- Alice deposits 1 ETH and 100 DAI in a liquidity pool. In this particular automated market maker (AMM), the deposited token pair needs to be of equivalent value. This means that the price of ETH is 100 DAI at the time of deposit. This also means that the dollar value of Alice's deposit is 200 USD at the time of deposit.
- In addition, there's a total of 10 ETH and 1,000 DAI in the pool – funded by other LPs just like Alice. So, Alice has a 10% share of the pool, and the total liquidity is 10,000.
- Let's say that the public price of ETH increases to 400 DAI. While this is happening, arbitrage traders will add DAI to the pool and remove ETH from it until the ratio reflects the current price
- If ETH is now 400 DAI, the ratio between how much ETH and how much DAI is in the pool has changed. There is now 5 ETH and 2,000 DAI in the pool, thanks to the work of arbitrage traders.
- Alice decides to withdraw her funds. As we know from earlier, she's entitled to a 10% share of the pool. As a result, she can withdraw 0.5 ETH and 200 DAI, totaling 400 USD
- But wait, what would have happened if she simply holds her 1 ETH and 100 DAI? The combined dollar value of these holdings would be 500 USD now.
- Alice would have been better off by HODLing rather than depositing into the liquidity pool. This is what we call impermanent loss.

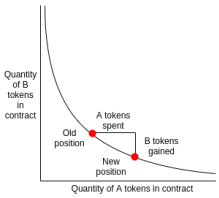
21

Disadvantage: Front-Runner Attack

- Current state (10 A, 10 B) : thus, c = 10*10=100
- Trader: want to spend 1 A, he would get 0.909091 B
- New state (11, 9.090909) (product = c = 100)

Front-running attack: a miner can do the following:

- Miner: spend 1 A: (11, 9.090909), get 0.909091 B
- Trader: spend 1 A: (12, 8.333333), get 0.757576 B
- Miner: spend 0.757576 B: (11, 9.090909), get 1 A
- Miner: earn 0.151515 B for free, at the loss of the trader



22

Vitalik Buterin's Front-Runner Solution

- **2 pools**: pool (10, 10) for spending A and another pool (10, 10) for spending B
- Starting state: ((10, 10), (10, 10))
- Miner: spend 1 A: ((11, 9.090909), (10, 10)), get 0.909091 B
- Trader: spend 1 A: ((12, 8.333333), (10, 10)); get 0.757576 B
- Miner: spend 1.111111 B: ((12, 8.333333), (9, 11.111111)), get 1 A
- Miner: lose 1.111111 - 0.909091 = 0.202020 B
- Trader: still lose 0.151515 B, but the miner *loses more*
- if the purchases were both infinitesimal in size, this is a 1:1 griefing attack,
- The larger the purchase, the larger relative loss the attacker gets

23

Capital Efficiency

- **Capital Efficiency** is the relationship between how much liquidity you provide to the pool and how much you get in return
- More capital efficiency if
  - Less **gas fees** to perform a transaction
  - Less **slippage** when swapping tokens
  - **Fewer transactions** to achieve what you want
- **Concentrated Liquidity**: Uniswap V3 significantly improves Capital Efficiency by allowing LPs to concentrate liquidity on a specific small price range of their choice

24

Algorithmic Stablecoin

I don't trust centralized-backed stablecoin. Can we create a stablecoin managed automatically by an algorithm that is transparent and never runs out of liquidity?

Prof. David (Duc) Tran | duc.tran@umb.edu

25

Stablecoin

- Pegged to the value of some fiat asset (for example, USD, Gold)
- 1-1 vs. under-collateralized vs. over-collateralized?
  - 1-1 Pegged: USDT, USDC
  - Under-collateralized: CDBC (Central Bank Digital Currency), FRAX, Ampleforth
  - Over-collateralized: DAI, Angle
- Centralized vs Decentralized (Algorithmic)?
  - Centralized: USDT, USDC, CDBC
  - Decentralized: DAI, Terra, Frax, Fei, Angle, yUSD, mUSD, Gyroscope

26

Pegged Stablecoin: Disadvantages

- Pegged 1:1 to a fiat asset, for example, gold or USD
  - Pay (collateralize) 1 USD to buy a 1 USDC: this token is minted new and given to the buyer, and the 1 USD saved in the reserve
  - Sell 1 USDC to get 1 USD: this 1 USD is removed from the reserve and returned to the seller, and the 1 USDC token will be burned
  - Tokens are traded freely in the market → hence price changing → arbitrage opportunity → stabilize price back to 1:1 peg
- Drawback
  - The stablecoin company may not be transparent and auditable, and may block a blacklist of stablecoin holders (for whatever reasons)
  - Regulatory risks due to competition with its official currency
  - The collateralized assets must be deposited to a bank which may have a negative interest rate (e.g., some European banks for Euro deposits)
  - The type of collateralized asset is limited: must be "real" assets, cannot be "imaginary" assets (for example, temperature in NYC)
  - Difficult to scale to many stablecoins so that they can be used in a trading exchange (e.g, FOREX)

27

Algorithmic Stablecoin

- Decentralized, automated by smart contracts
- Open source, transparent, auditable
- On-chain: hence, can only accept cryptocurrencies as collateral
  - E.g., ETH, BTC, or even stablecoins (USDC, USDT)

28

DAI Stablecoin

- **Over-collateralized decentralized** stablecoin, created by MakerDAO
- **DAI** = the stablecoin, pegged to the USD
- Idea: Stability relies on collateralization. DAI buyer must collateralize cryptocurrency (ETH, WBTC)
- **Collateral ratio** must be **150+%**
  - Buy DAI: deposit ETH, then the protocol mints new DAI, at most 100/150 the $value of this ETH collateral
  - Return DAI: to get the ETH collateral back, the DAI holder must pay the borrowed DAI amount plus some stability fee (fixed-rate interest)
- When collateral ratio becomes < 150% (due to ETH price decrease):
  - The protocol will liquidate some collaterals via auction sale to always keep collateral value higher than the DAI value total in circulation
- Maintain stability
  - When DAI price **< $1** in the market: **increase stability fee** to discourage minting new DAI and encourage repaying of debt
  - When DAI price **> $1**: **decrease the stability fee** to encourage borrowing more

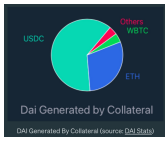
29

DAI Disadvantages

- 150% ratio is too high → capital **inefficiency**
- Stability fee is not automatically computed: depend on active governance
- If DAI value keeps increasing, cannot reduce Stability Fee to below zero → cannot keep 1:1 pegging to USD
- So, how to stabilize?
  - When DAI < $1: DAI borrowers who previously bought DAI at higher price will see the opportunity to buy new DAI at the lower price to repay the debt → increase BUY demand → DAI price up
  - When DAI > $1: DAI holders will sell DAI → increase SELL demand → bring DAI price down when they will repay the debt
- The DAI borrowers have to actively watch the market → complex, not for everybody
- Non-DAI people cannot participate in arbitrage opportunity → limit stability
- The DAI users are usually those who want to make profit from volatility. Those who seek asset stability do not benefit from this stablecoin

30

DAI Price (source: CoinMarketCap)



More stable recently because DAI becomes more popular (more borrowers). Why? MakerDAO introduced a small tweak in the system and USDC can be used to back DAI. As of July 2021, 60% of DAI has been minted using USDC. Using USDC, the collateral ratio just needs be 100% (not 150% as in the case of backing with ETH)

31

Algorithmic Stablecoin

- **Goal**: **decentralized** and **capital efficient**
- **Rebasing** model: Automatically **expand** token supply upon decreasing stablecoin price and **shrink** it otherwise
  - E.g., Yam, Ampleforth
  - Drawback: hard to achieve stability in practice during crisis where everyone loses faith in the token and wants to get rid of it
- **Seigniorage share** model:
  - E.g., Basis Cash, Empty Set Dollar, TerraUSD
  - **People crowdfund the reserve** for the stablecoin issuance
    - Stablecoins are sold by the protocol against this reserve: Sale proceeds go back to the reserve
  - Need a secondary **"reserve" token** (volatile, e.g., governance token)
  - To interact with the stablecoin, must be holders of reserve token (buy from exchanges)
    - Expecting a dividend earning or token appreciation
  - When stablecoin supply increases (decreases) → reserve token price increases (decreases)
    - Stablecoin can be burnt against reserve tokens (newly issued or auctioned)

32

Terra stablecoin

- **UST**: the stablecoin (1-1 pegged to USD)
- **Luna**: the reserve token
- When UST > $1: swap Luna for UST by the protocol at price UST = $1
  - For the same Luna amount, one gets more UST by swapping than buying UST directly in the market
  - These UST are newly minted → increase UST supply in circulation → bring UST price down
- When UST < $1: swap UST for Luna by the protocol at price UST = $1
  - The UST that the protocol gets from this swap will be burnt → bring UST price up
- Luna price increases/decreases with UST price

33

Frax stablecoin

- **Drawback** of using a volatile reserve token
  - It is a new asset → **difficult to bootstrap** -> require community support
  - If the system is in **crisis**, the value of this token can decrease, thus not being able to back the stable coin
- **Frax = a hybrid approach**: using both volatile token (FRX) and some stable reserve (USDC)
  - Less risky than using the volatile token only to back the stablecoin
- Regardless, could still be **under-collateralized** due to the volatility of the reserve token, **especially when people lose faith** (bank-run)

34

They still cannot keep it stable

Empty Set Dollar

Terra UST

35

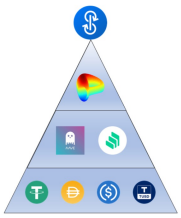
FEI stablecoin

- **No collateralization** needed
- Maintain a **FEI/ETH liquidity pool on Uniswap**
  - Liquidity providers (LPs) earn rewards for contributing to the liquidity pool
- To maintain stability: make **exit fee** (stablecoin redeem) **quadratically expensive**
- When FEI < $1: the fee is high (could go up to 50%), preventing LP from selling FEI
- This however leads to a serious liquidity problem
- In April 2021, FEI **crashed 50%** from its $1 peg
- Not so successful, FEI has now shifted back to a FRAX like model

36

Meta-stablecoin

- Meta-stablecoin = backed by a basket of other stablecoins
  - E.g., Gyroscope, yUSD, mUSD: backed by USDT, Dai, USDC, TUSD, sUSD
- Diversifies the risks of the individual stablecoins
- Drawback
  - **Composability risk**: a crisis with one backing token will affect the meta-stablecoin
  - **Market cap limit**: cannot grow above the marketcap of the backing stablecoins. These stablecoins couldn't be used to boost alone the entire DeFi ecosystem



yUSD stablecoin is backed by positions in the yCRV Curve AMM pool, composed of USDT, Dai, USDC, and TUSD

37

Derivative-Backed Stablecoin

- Main challenge for algorithmic stablecoin is to absorb volatility risk of crypto collaterals
- **Derivatives can help a protocol hedge against collateral volatility**
  - **Sell derivatives contracts** like perpetual futures to people who want to **gamble-profit** from possible future price increase. Get rewarded for sharing the volatility risk on behalf of the protocol
  - **If** collateral price decreases: long traders' losses are sent to the reserve
  - **Else**: the protocol pays the hedgers by the reserve's price-increase gain
- 2017: Veriabl (but never launched on mainnet)
- 2021: Angle (ongoing)
- Others: Pika Protocol, UXD, Lien Finance

38

Thank you

Prof. David Tran, PhD
duc.tran@umb.edu
University of Massachusetts

39