FORT: A DeFi Development and Application System with Unlimited Liquidity
Zaugust

James Zhao

September 9, 2021

Introduction

Liquidity is the key to on-chain applications. To solve this problem, previously DeFi has tried the traditional idea of order book and AMM models. However, these models are not ideal solutions and cannot integrate all financial services into the same protocol and share the same liquidity, resulting in wasted resources and low performance. This paper proposes a new model: FORT protocol, which creates the concept of discounted computers and on-chain currency units, and systematically solves the liquidity and uniformity problem of all DeFi. It can be used for all financial product development and economic relationship lock-in of off-chain activities.

1 History of DeFi

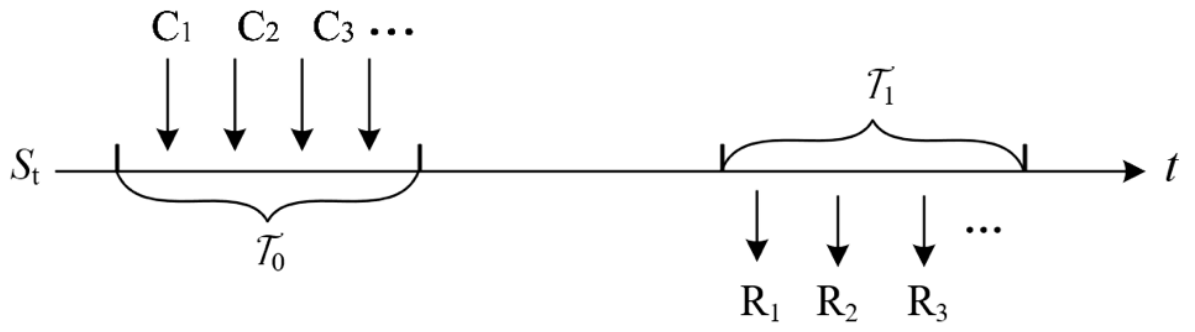
Decentralized finance, as known as DeFi, is the fastest growing application in the blockchain world and it has found a real demand. The history of DeFi can be traced back to the early days of orderbook-based on-chain transactions, as well as peer-to-peer lending. Some applications gained some attention around 2017, but due to the extremely high cost of on-chain brokering and the lack of decentralized oracle, this group of projects did not grow. Instead, Uniswap based on the AMM mechanism; Compound and MakerDao, lending protocols based on fund pools and asset prices, grew rapidly and led the wave of DeFi. Because this type of project better solves the fundamental contradiction of DeFi: lack of on-chain liquidity.

But no matter AMM or fund pool, its solution to liquidity problems is at the expense of the seller's flexibility. The seller needs to fix his own trading strategy and bear the fluctuations in the external market. Once the price is favorable to the seller, the buyer may choose to withdraw from the transaction. Once there is arbitrage, the buyers flock in. The seller has no choice but hope for subsidies from mining and commission or interest rate equalization. Although this asymmetric design temporarily alleviates the lack of on-chain liquidity, there are the following problems in the long run. Firstly, waste of resources due to the large amount of capital occupation. Numerous TVLs but only a small number of transactions are supported, and most TVLs are still rushing to liquidity mining; Secondly, the core variables, such as price and interest rate, are related to the size of the pool, which are easy to be arbitrage on the one hand, and it is difficult to carry out transactions and lending when the pool is not large enough on the other hand. Moreover, TVL of different products cannot be shared, resulting in the so-called portfolio lines being only formally combined, rather than liquidity sharing.

This kind of liquidity created at the expense of one party's choice is not a perfect idea under a decentralized architecture. Instead of allowing the seller to make asymmetric sacrifices, it is better to completely erase this asymmetry. The system only allows a completely symmetric buyer, and the seller is the system itself. This idea is in line with the spirit of the blockchain decentralized game. Everyone is in the same position to play the game with the algorithm, whether it is Bitcoin or Ethereum. Each participant only needs to pay the system tokens of the consideration to the system to get the financial products they want, and the income of the financial products will be settled by the system tokens. This idea frees us from the traditional model of financial transactions and allows the formation of a new financial paradigm, and ensures that DeFi does not just possess combinability, but is equivalent to a linear transformation in the same framework, with uniform programmability.

2 FORT Principle

Any financial service or financial product can be abstracted as the interchange of future revenue flow and current expenditure flow exchange, which is illustrated as follows



where

S_t is the price or interest rate information flow,

R_i is revenue flow,

$\mathcal{T}_0$ is the time domain of expenses, and

$\mathcal{T}_1$ is the time domain of revenue.

The principle of FORT is that, according to a given discounting algorithm, future revenue flows and current expenditure flows are equivalent (or slightly smaller, at this point in time for deflation), so each revenue flow corresponds to an expenditure flow, and both revenue flows and expenditure flows are uniformly settled with DCU. If we call the future benefit stream a financial product, we can call the corresponding expense stream the present value or cost of the product, which is uniformly called the cost here, and the process is as follows,

$$\sum_{\mathcal{T}_1} \mathbb{E}\left[e^{-rt_i} R_i | \mathcal{F}_0\right] \leq \sum_{\mathcal{T}_0} \mathbb{E}\left[e^{-rt_i} C_i | \mathcal{F}_0\right]$$

where

r is discount rate,

R_i is revenue flow,

C_i is expense flow,

$\mathcal{F}_0$ is information set at the time of transaction,

$\mathcal{T}_0$ is the time domain of expenses, and

$\mathcal{T}_1$ is the time domain of revenue.

Since a financial product can consist of a linear combination of basis return functions (BRF), each basis return function corresponds to a base discount function (BDF), such that the cost of the product is a linear combination of these basis discount functions. Thus, we can design the underlying return function and its discount function as a developable model: a discount computer - any financial product can be developed by this computer, and the underlying return function is like the instructions of the computer. The underlying discount function is like the price of these instructions or EVM-like gas, except that gas is paid in DCUs, and the instructions produce DCUs. the following is the calculation of the financial product and its cost.

$$\begin{aligned}
\mathbf{P} &= x_1 BRF_1 + x_2 BRF_2 + \cdots = \mathbf{BRF} \cdot \mathbf{X}^T \\
\mathbf{C}(\mathbf{P}) &= x_1 BDF_1 + x_2 BDF_2 + \cdots = \mathbf{BDF} \cdot \mathbf{X}^T
\end{aligned}$$

where

$\mathbf{P}$ is financial products; $\mathbf{C}(\mathbf{P})$ is costs

$\mathbf{X}$ is the $\mathbf{BRF}$ of P denotes the discounted function of $\mathbf{BDF}$

$\mathbb{E}\left[e^{-rt_i} BRF_i | \mathcal{F}_0\right] \leq \mathbb{E}\left[e^{-rt_i} BDF_i | \mathcal{F}_0\right]$

A variety of financial products can be developed based on discount computers, including options, perpetuals, leveraged transactions, swaps, common transactions, and lending, to name just a few, and almost any financial product can be produced.

DCU: Issuance and Settlement and Pricing

DCU stands for Decentralized Currency Unit. DCU is issued by FORT protocol and there is no upper limit, the initial DCU does not exceed 100 million pieces. In FORT protocol, DCU is the only currency unit, what you spend is DCU, what you get is also DCU.

For example, in the future, you will get 300 DCUs if you meet certain conditions, so you need to spend 50 DCUs now, and these 50 DCUs will be destroyed, and after 300000 blocks, you will get 300 DCUs, these DCUs are issued by the system.

As you can see, all DCU holders share the risks and benefits of DCU issuance or destruction and participate in the equilibrium of supply and demand in the secondary market for DCUs: the demand for DCUs is from those who buy financial products on the chain and those who invest in DCUs, and the supply is determined by the initial issuance and the DCUs settled by the FORT agreement, and the two reach a price equilibrium on the exchange. The advantage of sharing the same denomination unit is that we only need to continuously improve the liquidity of DCUs to solve all financial services without building too many tokens, whether trading, lending, derivatives can be solved by DCU denomination, payment, and settlement units.

According to $\mathbb{E}\left[e^{-rt_i}BRF|\mathcal{F}_0\right] \leq \mathbb{E}\left[e^{-rt_i}BDF|\mathcal{F}_0\right]$, total supply G_t fulfills $\mathbb{E}G_{t_2} \leq \mathbb{E}G_{t_1}$, $t_2 \geq t_1$, total demand D_t is determined by transaction demand, P_t is determined by equilibrium of (D_t, G_t). Considering the growth of demand and the long-term deflationary nature of supply, there is a logic for a sustained rise in P_t.

As the DCU is named, combined with the FORT contract, it is an on-chain universal currency with scenarios, something that BTC and ETH cannot achieve: BTC has no on-chain scenarios and has fixed issuance, ETH follows all scenarios as gas, but its issuance is according to a fixed algorithm, not incremental according to scenarios. DCU is guaranteed to clear in every scenario, which is in line with what many previous DCU guarantees clearing in each scenario, which matches the completely decentralized currency envisioned by many economists, and is a further development after BTC-ETH.

NEST Oracle

NEST oracle is the only truly decentralized oracle on the market today: given an off-chain price stream, how to design a decentralized game such that the game equilibrium can output a price stream with the smallest possible deviation from the off-chain price flow. nest oracle solves this problem with modules such as quotation mining, two-way options, validation cycles, price chains and beta factors.

NEST provides a price sequence that does not change the distribution of asset prices, but approaches a discrete sampling model, which is determined by the structure of the decentralized game, where the quote deviation and quotation density depend on the depth of the arbitrage market and the price of the NEST token. Overall, NEST provides an efficient decentralized oracle that maintains the fundamental traits of prices.

In the design of FORT, we tend to use highly efficient market prices, so the underlying chosen is the most liquid BTC and ETH, etc. The basic price model is the Geometric Brownian Motion, or the GBM model. Considering the deviation of prices and discrete time characteristics, we will correct the prices under GBM, this is the k-factor correction.

$$K = (0.00002T + 4\sigma) \cdot \gamma(\sigma)$$

where

σ is volatility in seconds

T is the time delay: $T =$ (height of successful packed block $-$ height of the block with the most recent valid NEST price) $\times$ timespan

$$\gamma = \begin{cases} 1 & \text{for} \quad \sigma \leq 0.0003 \\ 1.5 & \text{for} \quad 0.0003 < \sigma \leq 0.0005 \\ 2 & \text{for} \quad \sigma > 0.0005 \end{cases}$$

Time Domain

The time domain is denoted by $\mathcal{T}_i$ and is mainly divided into moments and intervals. A moment can be a definite moment or a random moment (e.g., a stoppage), and in the financial domain, intervals are often used to determine some mean or stoppage. Although the time on the blockchain is discrete, these discrete differences can be ignored for a longer period and compensated for in a shorter period based on the k-factor, and thus can be approximated by continuous time intervals.

Discounted Computers

We abstract all financial products (services) as a reciprocal of revenue flow and expense flow, and the revenue flow is represented by a linear combination of the underlying revenue function, then any financial product development only needs to determine the linear combination of the underlying revenue function to get its cost (present value) by the linear combination of the discounted function, such a linear combination is the same as we use computer programming, therefore we call this module figuratively as discounted computer, and it is shown as follows.

Any financial product corresponds to a piece of computer programming, so that the combinable lines of DeFi become program design and program calls in the same framework, reducing the difficulty of understanding and risk management.

Base Return Function and Discounted Function

The base return function (BRF) can be shown as a deterministic value (e.g., block 13678933 to get 1000dcu) or it can be a random variable after introducing the NEST oracle price. Here, we consider the basic types of deterministic values, random variables of NEST price oracle, and pure probability random variables, each of which consists of a polynomial function domain, exponential function, logarithmic function, absolute value function, maximum-minimum function, and definite integral operator, while the base discount function (BDF) contains a positive-terms distribution function as well as polynomial functions, exponential functions, logarithmic functions, etc. Here, we consider that the reality does not need so many return functions as well as the complexity of the calculation, we have chosen a relatively simple list of functions, which can be gradually improved.

As mentioned earlier, the basic return function is the basic instructions of the discount computer, a financial product is a program, the program is a combination of these instructions.

Discount Rate and Interest Rate Oracle

In principle, the discount rate responds to the risk-free return of the on-chain world, and we can choose a risk-free interest rate statistic of the chain into the POS yield of ETH or the interest rate provided by a decentralized interest rate oracle (a design as follows: given the number of DCU issues per year, anyone who locks DCUs can participate in the equalization of these issues) as the discount rate. However, this paradigm is considered in a traditional centralized world. In a decentralized world, in order to make the increase in DCU issues have deflationary properties and thus ensure a steady rise in DCUs, we can take the discount rate to a relatively small value, even zero.

Pricing Unit Transformation

If some kind of fiat or ETH is needed as the unit of measure, in FORT, it is sufficient to introduce the DCU/USDT or DCU/ETH price, which can be obtained by the NEST prediction machine. If the liquidity of DCU is large enough that the settlement of a single financial product has little impact on the price, the financial product with the introduced price is no different from the traditional financial product, and the pricing based on the risk-neutral measure can perfectly solve the calculation of the discount function, which can be used for hedging or asset portfolio management.

$$\mathbb{E}^{Q}\left[e^{-rt_i}BRF_i|\mathcal{F}_0\right] \leq \mathbb{E}^{Q}\left[e^{-rt_i}BDF_i|\mathcal{F}_0\right]$$

where $\mathbb{E}^Q$ denotes risk-neutral measure.

Financial Product Development

The financial product development here is the same as writing a smart contract, i.e., a vector with BRF as the base is found for the target's return, and that vector represents this financial product, while the product of that vector and the corresponding BDF base is the cost of the financial product, i.e., only that cost needs to be paid in the time domain $\mathcal{T}_0$, and that financial product is obtained. And that financial product will get the DCU incremented by the FORT contract in the time domain $\mathcal{T}_1$, and its quantity is the product of that vector and the BRF.

This process is just as mechanical as writing ordinary smart contract code. This makes it possible to implement all the financial products you want with the discounted computer programming of FORT, with uniformity, and without the developer having to operate the liquidity of the tokens, only the DCUs need to be sufficiently liquid.

Application Examples

FORT has a wide range of applications, covering almost all financial services and different trading structures (including peer-to-peer, multi-dollar, etc.). It is a landmark design in the history of blockchain development, as it covers almost all financial services and different transaction structures (including peer-to-peer, many-to-many, etc.), and can lock in various off-chain economic relationships.

Options and Options Coins

Options issuance is made very simple by simply entering an expiration date and an exercise price to obtain a call or put option whose cost is determined by a discounted function, although this formula is not a risk-neutral measure when DCU prices are not quoted, so care needs to be taken to understand the implications. If the DCU price is quoted it is the same as traditional options, except that the interaction becomes much simpler and there is not much brokering to consider.

A better model is to issue the options as a token, i.e., the same token for a given expiration date and strike price, regardless of when the issue starts. The advantage of this model is that it allows traditional derivatives exchanges to dispense with the issue and settlement issues: in order to meet the issue demand, they either need to do a lot of aggregation or find a market maker, and in order to settle, they often need margin management. And market makers also need to develop a hedging strategy, which is a huge financial auxiliary system, although traditional finance is happy to do so, but its cost is much higher than the FORT model, because with the latter issuance and settlement, exchanges only need to solve the problem of secondary market trading of derivatives.

Perpetual Contracts, Leveraged Trading and Leveraged Coins

Perpetual contracts or leveraged trading can also be very simple, as it is a dynamic settlement model and a basic revenue function. We can also develop perpetual or leveraged transactions into a model called a leveraged coin: dynamically changing the balance of its tokens based on price, which is also practiced in current algorithmic stablecoins.

Transactions, Price Coins and Stablecoins

A native asset is actually equal to a price coin $*$ DCU denomination unit, which is equivalent to splitting an asset into a dynamic price and a fixed settlement unit, except that this model can only be effectively implemented in a fully decentralized world: the traditional centralized world has the credit risk of cashing out. Thus, trading is equivalent to exchanging DCUs for various price coins, or settling out of DCUs with various price coins, or using the native asset to correspond to price coins 1 to 1 (which is actually slightly off, due to the bias of the price prediction machine). By analogy, a stablecoin pegged to a fiat currency such as the US dollar is a USDT price coin.

Exponential and Logarithmic Coins

A new paradigm is the index coin, where the percentage of price fluctuations feeds into the growth of returns in an exponential manner. Compared to leveraged coins, index coins have good properties: they never need to be closed, they grow faster, they can be transferred to each other, they can be freely stacked at the same address, etc. For example, when the price doubles, the index coin with e as the bottom can grow 7.4 times, and when the price doubles, the index coin grows 20 times.

Revenue Swaps

Various future revenue swaps are nothing but cost swaps, as they are all equivalent to discounted future revenue flow.

Borrowing and Lending

Borrowing is made simpler by pledging an asset recognized by the FORT contract to receive the corresponding DCU, repaying it to receive the pledged asset, and then being liquidated when the liquidation line is hit, where the core parameters are the liquidation line, pledge rate, interest rate, etc.

Insurance

Based on the tail characteristics of the event, a price insurance can be made to swap the tail loss with the premium.

Interest Rate Derivatives

Various interest rate derivatives are designed based on the base interest rate. It is possible to design various derivatives similar to those under its price information flow simply by having the information flow of the interest rate oracle.

Probability Coin

Design a token that gets a DCU at a given moment with a predetermined probability. For example, a 1-in-10 probability coin, each probability coin can have a 1-in-10 probability of getting 10 DCUs.

NFT Application

It is possible to lock any off-chain game or economic relationship designed by NFT based on DCU, i.e. all game assets can correspond to some probability coin or some derivative of the above. In this way no matter in which game, its game assets corresponding to NFT can be cashed in FORT, regardless of the existence of that game, thus building a consistent variable in the game world.

Multi-Way Trading

To design a transaction between two or more participants: A and B can make a contract based on FORT, each paying a certain amount of DCU at the current moment and receiving the corresponding DCU in the future, so that FORT can participate in the allocation between the two, thus realizing a multiplayer competition and game structure.

3 Summary

FORT offers a new paradigm: programming that understands a financial product as a basic discounted function. Its cost is the cost of invoking that function, much like EVM. The difference is that the economic relations of the discounted computer are internally generated. This new paradigm can cover almost all financial products (services) and can be bought at any time and settled with unlimited liquidity, where no market maker, no margin, no margin call, and no fear of being unable to settle are required. As long as the DCU liquidity is sufficient, it is extremely simple to restore the traditional financial market, and its function will be very powerful. Moreover, as the difficult issues of issuance and settlement are solved, traditional derivatives exchanges can focus on the secondary market, thus greatly reducing their costs. In addition, FORT can also become a fundamental consistency variable for building the popular metaverse, with the ability to traverse different games to lock in economic relationships, which has a very promising future.