Interest Protocol: Fractional Reserve Banking in Decentralized Finance[*]

Eddy Lee Getty Hill Face Shaver

April 2022

Abstract

Interest Protocol is the first fractional reserve banking protocol on the Ethereum blockchain that pays interest to all depositors. Interest Protocol issues a stablecoin, named USDi, that is both over-collateralized and highly scalable. Holders of USDi automatically earn yield without staking or sacrificing liquidity. Given market interest rates of 5% to 10%, Interest Protocol's lending operations are 5 to 7.6 times as capital efficient as those of lending protocols without fractional reserve.

Keywords: decentralized finance; fractional reserve banking; stablecoin; lending protocol; yield farming

[*]Emails

1 Introduction

Stablecoins serve an essential purpose in the cryptocurrency market. They are a store of value, allowing investors to protect their portfolios from market fluctuations. They also function as a medium of exchange, as many crypto assets trade against stablecoins, and payments such as salaries or grants are often made with stablecoins.

Meanwhile, lending and borrowing of stablecoins allow capital to flow to those who can make the most use of it. Borrowers are able to take leveraged positions or obtain liquidity from illiquid assets, and lenders receive yield from providing their capital.

Thus two major challenges for the cryptocurrency market is to develop a stablecoin that holds its peg to fiat while scaling flexibly in response to demand, and to design a mechanism that lends out the stablecoin in a capital-efficient way. Interest Protocol is a decentralized banking protocol on the Ethereum blockchain that aims to meet both challenges at once by applying *fractional reserve banking* to decentralized finance.

Fractional reserve banking consists of a number of protocol operations. Interest Protocol issues a stablecoin called USDi. USDi can be minted by depositing USDC into the protocol. USDi is a liquidity provider (LP) token that represents a claim on the USDC and the yield from the USDC. The protocol generates yield by making loans denominated in USDi. Finally, the yield is distributed to all holders of USDi.

These unique features allow Interest Protocol to provide the following benefits to its users.

- *USDi is stable.* The safest way to peg a stablecoin is to back it by assets. USDi is always over-collateralized by a combination of USDC reserves and collateral posted by borrowers.

- *USDi is scalable.* Anyone can mint USDi by either depositing USDC or posting collateral and borrowing USDi. At a given interest rate, users can borrow more USDi than most other stablecoins.

- *USDi holders earn yield without staking.* Most stablecoins only earn yield when they are staked. In contrast, all circulating USDi continuously and automatically accrues interest. Users can simply hold USDi in their wallet and see their balance grow. Investors providing ETH-USDi liquidity to automated market makers earn interest on the USDi portion of their liquidity.

- *Interest Protocol is capital efficient.* Compared to lending protocols without fractional reserve, at any given interest rate, Interest Protocol is able to generate more loans without incurring additional liquidity risk. This efficiency gain is shared by all users of the protocol. Borrowers can borrow more at lower rates, and each additional loan creates more USDi holders who earn yield. Holders of the protocol's governance tokens, called Interest Protocol Tokens (IPT), benefit from higher protocol revenue.

The rest of this paper describes the mechanisms through which Interest Protocol offers these benefits. Section 2 compares Interest Protocol to existing DeFi protocols that lend stablecoins. Section 3 introduces the USDi stablecoin, and Section 4 defines the accounting variables used by the protocol. Section 5 explains how the protocol makes loans and sets interest rates, and shows that the protocol's lending operation is more capital efficient than lenders that do not implement fractional reserve banking. Section 6 describes how protocol operations affect the accounting variables, and Section 7 explains the implementation of automatic yield distribution. Section 9 lays out the protocol's risk management mechanisim for liquidity and credit risk. Finally, Section 10 describes protocol governance and introduces the governance token, IPT.

2 Comparison to Existing DeFi Protocols

2.1 MakerDAO

The current iteration of MakerDAO is closely related to Interest Protocol because they issue a stablecoin, DAI, against a reserve that consists of fiat-backed stablecoins. There are three major differences. First, Interest Protocol automatically pays interest to all depositors, i.e. all Ethereum addresses that hold USDi. In contrast, DAI holders do not receive interest payments from MakerDAO unless they stake their DAI in the DAI Savings Rate contract. Second, Interest Protocol offers multi-collateral vaults, so that users can take out a single loan against a combination of multiple collateral assets. MakerDAO only offers single-collateral vaults. Third, Interest Protocol has a flexible interest rate curve that mitigates liquidity risk... THIS IS A BIG DIFFERENCE. MAKER'S CURVE IS A DOT.

Maybe a fourth difference is better governance?

2.2 Compound and Aave

Compound and Aave are multi-asset lending protocols where users can borrow assets against other assets. The major advantage of Interest Protocol over Compound or Aave is capital efficiency coming from fractional reserve. Issuing USDi against a reserve of USDC allows Interest Protocol to generate more loans without incurring additional liquidity risk. 5.3 provides a detailed description.

3 USDi

USDi is a stablecoin pegged to $1 that automatically earns interest. Anyone can mint USDi by depositing USDC into the protocol at a 1:1 ratio. The protocol maintains a reserve of USDC, and USDi can be redeemed for USDC in the reserve at a 1:1 ratio. Initially, USDC will be the unique asset held in the protocol reserve. Other stablecoins may be supported in the future.

USDi is also minted when users borrow from the protocol. Borrowers can post collateral and take out loans denominated in USDi. The protocol mints USDi to make loans and burns USDi when it is repaid.

3.1 Compounding Yield

The interest accrued to USDi loans, net of a protocol fee, is distributed pro rata to all USDi holders in the form of additional USDi. The protocol implements the distribution by scaling up the balance of USDi tokens in all addresses so that the value of each USDi remains pegged to $1.[1] This means that USDi is a *liquidity provider (LP) token that is liquid:* USDi earns yield generated from protocol operations, but remains liquid because it is pegged to $1.

As a result, USDi holders are able to *earn compounding yield without staking*. This improves gas efficiency because there is no need to switch back and forth between the LP token and the stablecoin. It is especially beneficial to retail investors who spend a higher percentage of their trades on gas fees. Investors no longer have to choose between liquidity and yield; USDi provides both.

3.2 Peg Mechanism

When USDi trades above $1, users can arbitrage by depositing USDC into the protocol, minting USDi, and selling USDi in the market. When USDi is priced below $1, users can buy USDi and redeem it for USDC against the protocol reserve.

It is theoretically possible that the protocol reserve is at times insufficient to immediately meet a large redemption demand. However, as described in Section 5, whenever

[1]Section 7 explains the scaling process in more detail.

the reserve ratio is close to 0, the borrowing rate becomes prohibitively high. This incentivizes borrowers to repay their loans and depositors to mint more USDi, both of which increase the reserve. The protocol calibrates the interest rate function to guarantee quick redemption in all market conditions.

3.3 Scalability

USDi can easily scale in response to higher demand due to two factors. First, USDi is minted from both depositing USDC and borrowing USDi. When the interest rate is low, those who wish to use USDi can borrow it at a low cost. Under higher rates, users can deposit USDC to mint USDi and benefit from the yield. Second, as explained in Section 5, the protocol's USDi lending operations are very capital efficient. This means that given the same interest rate, the protocol can generate more loans and thereby supply more stablecoins to the market compared to traditional lending protocols.

4 Accounting

Interest Protocol's operations are based on the following accounting identity:

$$\text{Reserve} + \text{Loan} + \text{Treasury} = \text{Deposit} + \text{Equity}$$

The variables on the left-hand side represent the protocol's assets, and those on the right-hand side are the protocol's liabilities. Each variable is defined as follows:

- **Reserve** is the pool of USDC that is available for redemption by holders of USDi. Whenever a user deposits USDC to mint USDi, the USDC is added to Reserve. USDC in Reserve can only be withdrawn by burning an equivalent amount of USDi.
- **Loan** is the balance of all outstanding USDi loans that have been made by the protocol.
- **Treasury** is the pool of assets held directly by the protocol. It will mostly consist of USDC and USDi, but can hold any other asset. Treasury is used to fund protocol operations, such as audits or grants.
- **Deposit** is the total supply of USDi.
- **Equity** equals the total value of the protocol's assets minus Deposit. Equity grows when the protocol earns fees from lending operations, and it serves as a safety buffer against credit risk.

Since USDi is minted if and only if USDC is deposited into Reserve or a borrower takes out a loan, Deposit is always equal to the sum of Reserve and Loan. Section 6 provides a detailed description of how the accounting variables change from protocol operations.

5 Lending USDi

Interest Protocol generates revenue from lending USDi. This section describes how the loans are made and interest rates are determined. Table 1 provides a list of variables that we introduce in this section. Borrow APR (r) and protocol fee rate (f) are annualized from per-block values in the smart contract.

Variable	Name in Smart Contract	Description
$\bar{v}$	`maxLTV`	Maximum LTV for a collateral asset
$\bar{l}$	`debtLimit`	Maximum amount a wallet can borrow
R	`ReserveBalance`	Amount of USDC available for redemption
D	`DepositBalance`	Circulating supply of USDi
s	`reserveRatio`	R/D
r		Borrow APR
L	`LoanBalance`	Total balance of all outstanding loans
f		Protocol fee rate (annual)
l	`loanBalance`	Balance of a wallet's outstanding loan
k	`liquidationPenalty`	Liquidation penalty

Table 1: Variables Related to USDi Loans

Loans are always denominated in USDi, have a variable interest rate, and never mature. Anyone can borrow USDi from the protocol by posting collateral. The protocol maintains a whitelist of assets that can be used as collateral, and each collateral asset has a maximum loan-to-value ratio (LTV). Borrowers call `post` to post assets in their wallet as collateral. They can then call `borrow` to borrow USDi up to their *debt limit*, which is calculated by multiplying the market value of each collateral asset by its maximum LTV, and summing across all assets. Formally, let collateral assets be indexed by $i \in I$, and let $\bar{v}_i$ denote the maximum LTV for asset i and p_i the price of asset i. Suppose a borrower has posted q_i of asset i as collateral for each $i \in I$. Then his debt limit is

$$\bar{l} := \sum_{i \in I} p_i q_i \bar{v}_i.$$

The function `borrow` is reverted if it results in the borrower's loan balance being higher than his debt limit.

5.1 Interest Rate

All USDi borrowers pay interest according to a single borrowing interest rate that varies over time. The interest paid by borrowers, net of a protocol fee, is distributed pro rata to all USDi *depositors*. Anyone who holds USDi is a depositor, regardless of how they obtained USDi - by depositing USDC, borrowing USDi, or trading in the market.

5.1.1 Borrow Rate

Let R denote the balance of USDC held in Reserve, and D the balance of Deposit, which equals the amount of all circulating USDi. We first define the *reserve ratio* s by

$$s = \frac{R}{D} \in [0, 1].$$

The reserve ratio is a measure of the liquidity that the protocol holds to meet immediate demand for redemption of USDi, relative to the total supply of USDi. If the reserve ratio is 10%, for every 10 USDi that exists in the market, there is 1 USDC that can be immediately redeemed. The higher the reserve ratio, the more likely that the protocol can immediately honor unexpected redemption demands.

Since making more loans from a given amount of Reserve lowers the reserve ratio, the protocol wants to encourage borrowing when the reserve ratio is high and discourage borrowing when the ratio is low. Thus the interest rate for borrowers is decreasing in the reserve ratio. Specifically, the annual borrowing interest rate, r_B, is a continuous, piecewise-linear function of s given by

$$r_B(s) = \begin{cases} -\frac{r_0 - r_1}{s_1} s + r_0 & \text{if } 0 \le s \le s_1 \\ \frac{r_1 - r_2}{s_2 - s_1} s + \frac{(r_1 - r_2) s_2}{s_2 - s_1} + r_2 & \text{if } s_1 < s \le s_2 \\ r_2 & \text{if } s_2 \le s \le 1, \end{cases}$$

where r_0 is the upper bound on the rate, r_2 is the lower bound on the rate, $s_1, s_2 \in (0, 1)$ are the thresholds at which the kinks occur, and r_1 is the rate when $s = s_1$. Function $r_B(s)$ is thus decreasing in the reserve ratio s. Figure 1 plots $r_B(s)$ for the initial parameter values that will be used at launch: $r_0 = 2$, $r_1 = 0.1$, $r_2 = 0.005$, $s_1 = 0.4$, and $s_2 = 0.6$.

The upper bound r_0 is intended to be high enough that it is prohibitively expensive for borrowers to borrow at rates close enough to r_0. As long as the borrow rate is strictly below r_0, the reserve ratio s remains strictly positive, and the protocol has USDC in Reserve to meet redemption demand.

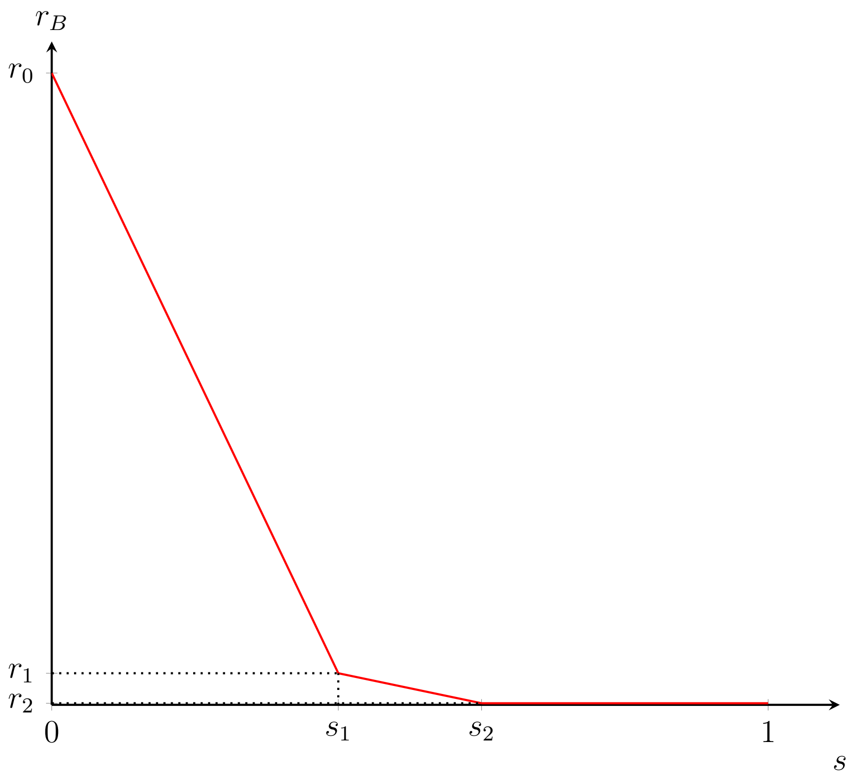

Figure 1: Reserve ratio (s) and Borrow APR (r_B)

5.1.2 Deposit Rate

Suppose that the current balance of outstanding loans is L, and that the current reserve ratio is s. Then the borrow APR is $r_B(s)$, and the annualized interest accrued to all outstanding loans is given by $r_B(s)L$. A percentage of this interest is withheld by the protocol as fees, and the rest is distributed pro rata to all depositors. If the current circulating supply of USDi is D, and the annual protocol fee rate is $f \in [0, 1]$, the current annual percentage rate for depositors, r_D, is therefore given as a function of s by

$$r_D(s) = \frac{r_B(s)L(1-f)}{D}.$$

The total supply of USDi equals the balance of all outstanding loans plus Reserve, that is, $D = R + L$. The deposit APR can thus be rewritten as follows:

$$r_D(s) = r_B(s)(1-s)(1-f).$$

The deposit rate, like the borrow rate, is decreasing in the reserve ratio. When the reserve ratio is low, the protocol encourages more deposits by setting a higher deposit rate. Function $r_D(s)$ is plotted along with $r_B(s)$ in Figure 2 for $f = 0.10$ and otherwise using the same initial parameter values as in Figure 1.

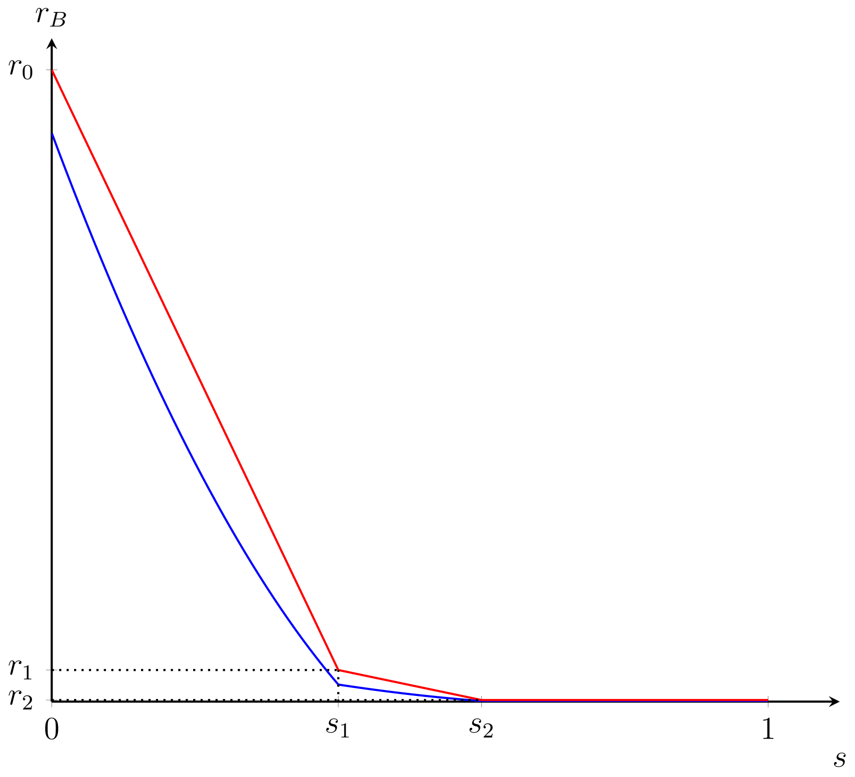

Figure 2: Reserve ratio (s) and Borrow APR (r_B)

5.2 Liquidation

Recall that a user can borrow USDi up to his debt limit $\bar{l}$. A borrower is *under-collateralized* if the balance of his outstanding loan, denoted l, is greater than his debt limit: $l > \bar{l}$. This can occur when l increases because of accrued interest or when $\bar{l}$ decreases due to a fall in the value of collateral assets.

Once a borrower becomes under-collateralized, anyone can liquidate the borrower's collateral by calling `liquidate`. The entity performing liquidation is called a liquidator. A liquidator purchases the borrower's collateral from the protocol by paying USDi or USDC to the protocol. If a collateral asset i has an oracle price of p_i, a liquidator purchases the asset at the price of $(1 - k_i)p_i$, where k_i is the liquidation penalty for asset i. The liquidation penalty serves as an incentive for liquidators to purchase the asset and an incentive for borrowers to keep their loan balance below debt limit. The liquidator can liquidate up to the amount that makes the borrower over-collateralized, plus a buffer.

Formally, suppose a borrower has posted q_i of collateral asset i for each $i \in I$, and that the price of asset i is p_i. If $l > \bar{l} = \sum_{i \in I} p_i q_i \bar{v}_i$, a liquidator who wishes to liquidate this borrower can purchase from the protocol an asset y_i at price $(1 - k_i)p_i$, subject to the following constraints:

1. The liquidator can only purchase up to the amount of collateral posted by the borrower: $y_i \in [0, q_i]$.

2. The liquidator can only purchase up to the amount that makes the borrower sufficiently over-collateralized, with buffer $e > 0$:

$$y_i \leq \bar{y} := \frac{(1+e)l - \bar{l}}{p_i((1+e)(1-k_i) - \bar{v}_i)}.$$

The upper bound $\bar{y}$ in the second constraint is the value of the minimization problem:

$$\begin{aligned} \bar{y} &= \min y \\ \text{subject to} \quad & \bar{l} - p_i y \bar{v}_i \geq (1+e)(l - p_i y(1-k_i)). \end{aligned}$$

Of course, the liquidator should only be able to purchase up to the amount of the borrower's outstanding loan: $(1 - k_i)p_i y_i \leq l$. However, it is easy to see that this is implied by the two constraints above.

In words, the liquidator is allowed to liquidate up to the minimum amount of asset i such that, if liquidated, would bring the borrower's collateralization ratio, $\bar{l}/l$, above $1 + e$. The bound $\bar{y}$ is increasing in e if $l \leq (1 - k)\bar{l}/\bar{v}$. If the borrower is sufficiently under-collateralized such that $l > (1 - k)\bar{l}/\bar{v}$, $\bar{y}$ decreases in e, which may seem counter-intuitive. However, whenever $\bar{y}$ is decreasing in e, $\bar{y}$ is higher than $l/(1 - k_i)p_i$, which is the upper bound from the third condition. Thus $\bar{y}$ is increasing in e whenever it is relevant.

As a result of liquidation, the borrower forfeits the collateral that was bought by the liquidator, and the borrower's loan balance is reduced by the amount of USDi paid by the liquidator. That is, the borrower now has $q_i - y_i$ of asset i posted as collateral, and his outstanding loan balance is $l - (1 - k_i)p_i y_i$.

5.3 Capital Efficiency

Most existing lending protocols directly lend out the stablecoins that lenders deposit, such as USDC or USDT. On the other hand, Interest Protocol introduces fractional reserve banking, so that the USDC deposited into the protocol is kept in Reserve, and loans are only made in USDi. This innovation allows Interest Protocol to be capital efficient.

In particular, compared to existing lending protocols, Interest Protocol can *make more loans from a given amount of USDC reserve while maintaining the same level of liquidity risk.*

Recall that the reserve ratio represents Interest Protocol's preparedness for meeting redemption demand, and hence is an inverse measure of liquidity risk. The reserve ratio corresponds to $1-$ (utilization rate) for lending protocols such as Compound (citation) or Aave (citation) because for every dollar of LP token (such as cUSDC or aUSDC), the protocols maintain $1-$ (utilization rate) USDC that is ready to meet redemption demand. Therefore, to argue that Interest Protocol is more capital efficient, we consider the maximum amount of loans Interest Protocol can make from a given amount of USDC in Reserve while keeping the reserve ratio above some level $\underline{s} \in (0,1)$, and compare this to the maximum amount of loans Compound or Aave can make from a given amount of USDC deposits while keeping the utilization rate below $1-\underline{s}$.

5.3.1 Capital Efficiency - Example

For example, suppose Interest Protocol has 100 USDC in Reserve and wishes to maintain a reserve ratio of at least 10%, i.e. $\underline{s} = 10\%$. Since the users who deposited USDC would have received equivalent amounts of USDi, there is 100 USDi circulating before any loans are made. If the protocol lends 900 USDi, the reserve ratio becomes

$$s = \frac{R}{D} = \frac{R}{R+L} = \frac{100}{100+900} = 10\%.$$

Thus the maximum amount of loans that Interest Protocol can make is 900 USDi.

Now, suppose Compound or Aave has 100 USDC in deposits and wishes to maintain a utilization rate of at most 90%, i.e. $1-\underline{s} = 90\%$. If the protocol lends out 90 USDC, then we have

$$\text{Utilization Ratio} = \frac{90}{100} = 90\%,$$

which means that Compound or Aave can lend out at most 90 USDC. We therefore see that Interest Protocol can make 10 times as much loans as Compound or Aave without taking on additional liquidity risk.

5.3.2 Capital Efficiency - Formal Argument

To make a formal argument, suppose Interest Protocol has a fixed amount of USDC reserves, R, and wishes to maintain the reserve ratio above $\underline{s}$:

$$s = \frac{R}{D} \geq \underline{s}.$$

Using our assumption that $D = R + L$, the above condition can be rewritten as

$$L \leq R\left(\frac{1-\underline{s}}{\underline{s}}\right). \tag{1}$$

Thus $R\left(\frac{1-\underline{s}}{\underline{s}}\right)$ is the maximum amount of loans that the protocol can generate while maintaining a reserve ratio of at least $\underline{s}$.

However, suppose Compound or Aave has R USDC in deposits and wishes to keep the utilization rate below $1-\underline{s}$. If the protocol lends out $\tilde{L}$ USDC, it must be that

$$\frac{\tilde{L}}{R} \leq 1 - \underline{s},$$

which can be rewritten as

$$\tilde{L} \leq R(1-\underline{s}). \tag{2}$$

Comparing (1) and (2), we see that Interest Protocol can make $\frac{1}{\underline{s}}$ times as much loans as Compound or Aave while maintaining the same level of liquidity risk. In other words, introducing fractional reserve banking allows Interest Protocol to increase capital efficiency by a multiple of $\frac{1}{\underline{s}}$.

Like many decentralized lending protocols, Interest Protocol does not impose a fixed value of $\underline{s}$ as a reserve requirement. Instead, the protocol sets the interest rate function $r_B(s)$ so that s remains high enough for practical levels of interest rate. Assuming that a single interest rate for stablecoins is set by the market, at a given point in time, the market interest rate pins down s, which in turn determines the efficiency gain from fractional reserve banking. For example, suppose the current borrowing rate for stablecoins is 5%. Under the initial parameters of $r_B(s)$, this implies that the reserve ratio is $s \approx 0.1968$, at which Interest Protocol is more than 5 times as efficient as lenders without fractional reserve. If the interest rate is 10%, the protocol is nearly 7.6 times as efficient as existing lenders. Notice that the efficiency gain becomes larger when the interest rate is higher; Interest Protocol is able to supply more loans precisely when the market demands them more.

Capital efficiency of the lending mechanism also means that USDi is highly scalable. Compared to lenders without fractional reserve, Interest Protocol is able to supply a larger quantity of loans at a given interest rate. Therefore, given the same level of demand for borrowing, USDi will have a larger circulating supply compared to stablecoins such as USDC or USDT that are not minted through lending.

6 Protocol Operations and Accounting Variables

Recall that the protocol uses the accounting identity, Reserve + Loan + Treasury = Deposit + Equity. Denoting Treasury balance and Equity balance by T and E, we have

$$R + L + T = D + E.$$

This section describes how the five accounting variables change as a result of common protocol operations.

Operation	Changes	Description
User borrows x USDi	$L \leftarrow L + x$ $D \leftarrow D + x$	Mint x USDi for user
User repays x USDi	$L \leftarrow L - x$ $D \leftarrow D - x$	Burn x USDi from user
User deposits x USDC	$D \leftarrow D + x$ $R \leftarrow R + x$	Mint x USDi for user
User withdraws x USDC	$D \leftarrow D - x$ $R \leftarrow R - x$	Burn x USDi for user
Protocol pays x USDi in grant	$E \leftarrow E - x$ $T \leftarrow T - x$	Transfer x USDi from T
Interest accrues for t blocks at rate r_{block}	See Section 7.2 for details	

Table 2: How Protocol Operations Change Accounting Variables

All user deposits of USDC go to Reserve and remain there until they are withdrawn. USDC in Reserve can be withdrawn only by burning an equal amount of USDi tokens, and there are no exceptions, including for the protocol itself. The protocol is only able to spend USDi or USDC that it holds in Treasury.

7 Implementing Automatic Yield

Interest Protocol automatically and continuously distributes yield to all USDi holders pro rata. To implement this, the protocol builds upon the contract `UFragments.sol`, which was developed by Ampleforth (**?**) and allows the balance of an ERC20 token in every address to scale automatically in proportion to a global index called `gonsPerFragment`. Interest Protocol adds minting and burning functionalities to the contract so that users can mint USDi by depositing USDC or taking out a loan.

Intuitively, fragments represent the USDi balance of an address, and gons represent the balance of the "internal token" that is actually used for transactions. The ratio between the two variables is the global index `gonsPerFragment`, denoted by i. When yield is distributed to USDi holders, the balance of gons in each address remains unchanged, but `gonsPerFragment` decreases so that each gon corresponds to more fragments. This allows the protocol to distribute yield in a gas-efficient way by updating a single global index.

7.1 Using `gonsPerFragment` to Scale USDi Balances

At time t, the USDi balance of an address is given by

$$\frac{\texttt{gonBalance}}{\texttt{gonsPerFragment}},$$

where `gonBalance` is the amount of gons in the address.

For example, suppose that an address with 0 USDi balance receives 100 USDi at block t. If $i_t = 0.5$, this means that the address has actually received $100 \times 0.5 = 50$ gons. If `gonsPerFragment` decreases to $i_{t'} = 0.25$ at block $t' > t$, then the USDi balance of the address at t' is scaled up to $50/0.25 = 200$. This represents the fact that all USDi holders have gained a yield of $(0.5 - 0.25)/0.25 = 100\%$ from block t to block t'.

7.2 Updating `gonsPerFragment` by calling `pay_interest()`

Any transaction that changes the Reserve or Deposit (and therefore the reserve ratio) updates `gonsPerFragment` and `interestFactor`. Suppose that we are in block t, and the last block before t in which Reserve or Deposit changed was block t_0. This means that the per-block borrowing rate must have remained constant after t_0, since the rate is determined by the reserve ratio. Denote this constant borrowing rate by r_{block}, and let R_τ, L_τ, T_τ, D_τ, g_τ, and i_τ denote the values of Reserve, Loan, Treasury, Deposit, `gonsPerFragment`, and `interestFactor` at block τ, respectively. Now, suppose that a protocol operation (explained in the Operations section) that changes either Reserve or

Deposit occurs in period t. This changes the variables in two steps:

1. Call `pay_interest()`. Letting f_u denote the per-update protocol fee rate, the function makes the following updates:

$$\begin{aligned}
i_t &= i_{t_0}(1 + r_{block})^{t-t_0} \\
L_t &= \text{totalBaseLiability} \times i_t \\
g_t &= g_{t_0}\frac{D_{t_0}}{D_{t_0} + (1-f_u)(L_t - L_{t_0})} \\
T_t &= T_{t_0} + f_u(L_t - L_{t_0}) \\
D_t &= D_{t_0} + (L_t - L_{t_0}).
\end{aligned}$$

2. Run the protocol operation. This will once again update R_t or D_t (and hence update the borrow rate).

Any function that changes the reserve ratio is first required to update `interest factor`, Loan, `gonsPerFragment`, Treasury, and Deposit by adding the interest that has accrued since the block at which the last such function was called. Then, the actual contents of the function are run, which updates the borrow rate.

8 USDi Liquidity

Here, liquidity means the ability to convert between USDi and other assets, such as ETH or BTC, without having a large impact on the market price.[2]

8.1 Importance of USDi Liquidity

This and rebasing is what enables gasless staking.

8.2 Bootstrapping Liquidity

LM

[2]Note that this definition of liquidity is related, but distinct from, definition of the protocol's liquidity, which is the ability to meet redemption demand.

9 Risk Management

To operate safely in all market environments, Interest Protocol manages two major economic risks - liquidity risk and credit risk. This section defines these concepts in the context of Interest Protocol and describes how the protocol manages the risks.

9.1 Liquidity Risk

Liquidity is defined as the protocol's ability to meet redemption demand at all times. The protocol has an obligation to USDi holders to redeem 1 USDi for 1 USDC on demand. On the other hand, the protocol allows borrowers to keep an outstanding loan balance as long as they wish to, provided that they maintain enough collateral to stay under debt limit. This causes a maturity mismatch, and it is possible at times for the protocol to face redemption demand in excess of USDC held in Reserve. The protocol uses two mechanisms to minimize this possibility.

First, Interest Protocol aims to maintain a sufficiently high reserve ratio at all times by using a variable interest rate. As explained in Sections 3.2 and 5.1, whenever the reserve ratio is low, a high interest rate incentivizes users to replenish the reserve by repaying their debt or depositing USDC. This ensures that any shortage in reserve is short-lived.

Second, as described in Section 8, the protocol aims to guarantee liquidity in various USDi markets. As long as USDi can be swapped for other assets with little slippage, users can hold USDi to earn interest until they wish to purchase other assets, at which point they can buy the assets by paying with USDi. Thus liquidity in USDi markets lowers the demand to redeem USDi for USDC.

9.2 Credit Risk - If no `writeOff()`, just talk about liquidation mechanism and careful onboarding of collateral?

Credit risk is the risk of borrowers not repaying their loans. Because Interest Protocol only makes over-collateralized loans, it is protected from credit risk under most circumstances. However, during market downturns, it is possible for the value of collateral assets to drop below the amount of outstanding loans before the assets can be liquidated. It is therefore necessary to have mechanisms to protect USDi holders against credit risk.

When a sufficiently large loan becomes uncollectible, the protocol writes off the loss by decreasing Loan, Treasury, Deposit, and Equity by the same amounts. This is implemented as follows.

Under extreme circumstances, it is possible that Equity does not cover all losses. This means that the protocol has more deposits than assets, i.e. $D > R + L + T$. In this case, the protocol initiates a bailout by auctioning off Interest Protocol Tokens.

10 Governance and Interest Protocol Token

While many operations of Interest Protocol happen automatically on-chain, the protocol inevitably requires governance actions for maintenances and upgrades. For example, the protocol automatically sets the interest rate as a function of reserve ratio, but this function may need to be updated from time to time in response to secular market trends. Similarly, while users can permissionlessly borrow USDi against registered collateral assets, the protocol may wish to register a new asset as collateral.

Interest Protocol's governance is controlled by a decentralized autonomous organization, which we call Interest Protocol DAO (IP DAO). IP DAO manages the smart contracts associated with the protocol by updating parameters or upgrading contracts.

Interest Protocol Token (IPT) is the governance token of the protocol. Whenever the DAO votes on a proposal, one IPT represents one vote. IPT will be distributed to the community through public sales and liquidity mining programs.

10.1 Actions Controlled by Governance

IP DAO can make the following parameter changes:

- Register new collateral assets
- Change LTV or liquidation incentive of existing collateral assets
- ...

The following contracts are upgradable by the DAO:

- USDi.sol
- Curve
- ...

10.2 IPT: Distributing Control of the Protocol

10.3 Governance Process

Propose, Review, Vote, Queue, Execute, Bravo/OZ

Initial msig

Msig for pausing

10.4 Recognized Delegates

Does it help?

11 Oracles

12 Voting with Collateral

When borrowing against collateral assets that are governance tokens (that follow certain standards?), users can delegate the voting power of their assets to an address of their choice. This allows Interest Protocol users to borrow against governance tokens while retaining voting power.

Does this belong in the governance section?

13 Disclaimer

DYOR