DEGIS Whitepaper

November 2021 @ v1.0.5

Abstract

The insurance industry is enormous and plays an essential role in the financial market. However, there is cumbersome paperwork and claim procedures in the traditional way, which is unfriendly for users. However, with the increasing development of blockchain technology, things can become more efficient and secure under smart contract settings. A combination of high- advanced tech and time-honored insurance products gives birth to DEGIS, our integrated decentralized insurance platform.

1 Introduction

Insurance, as the fundamental part of the financial market, has a considerable market volume. Related products such as delay insurance, accident insurance have been shown as effective in hedging various risks. Nevertheless, in the traditional market, companies need to hire sales associates to investigate personal information about customers before offering insurance products. Also, there are rather complex paperwork and cumbersome claim procedures, limiting insurance application scenarios.

With DeFi infrastructure, smart contracts will play sales associates' roles based on their immutable and forced automatic execution characteristics. We are thinking if there will be a platform where anyone can put their money to behave like an insurance provider to help cover others' risks without complicated verification and settlement procedures.

In this paper, we present DEGIS v1, a novel Integrated Decentralized Insurance platform that entitles both liquidity providers and insureds to more flexible risk covering options with lower costs and higher returns. Technical advantages of blockchain and the big data analysis model benefit each participant engaged in DEGIS. Besides, DEGIS offers several significant features:

- Multiple coverages of on-chain and off-chain products.
- Concentraded pool to aggregate capital and liquidity.
- Secondary market to promote circulation.
- NFT attributes and lottery mechanism.

2 How DEGIS Works

We set a public insurance pool here on DEGIS, where liquidity providers can enter it acting as insurance providers. Each insurance would be activated by the platform and backed by the liquidity pool. This is very convenient for users with different needs and also helps concentrate capital and liquidity. Through big data analysis, we get rather precise prediction on expected payoff, which supports us to offer low and competitive price in insurance market. On the other hand, the insurance status is automatically updated through decentralized oracles, i.e., trustworthy third-party off-chain data service providers. The payoff will be cleared immediately once the insurance is due. Comparing to the salesmen, smart contracts can reduce the manual cost and dramatically optimizes the user experience.

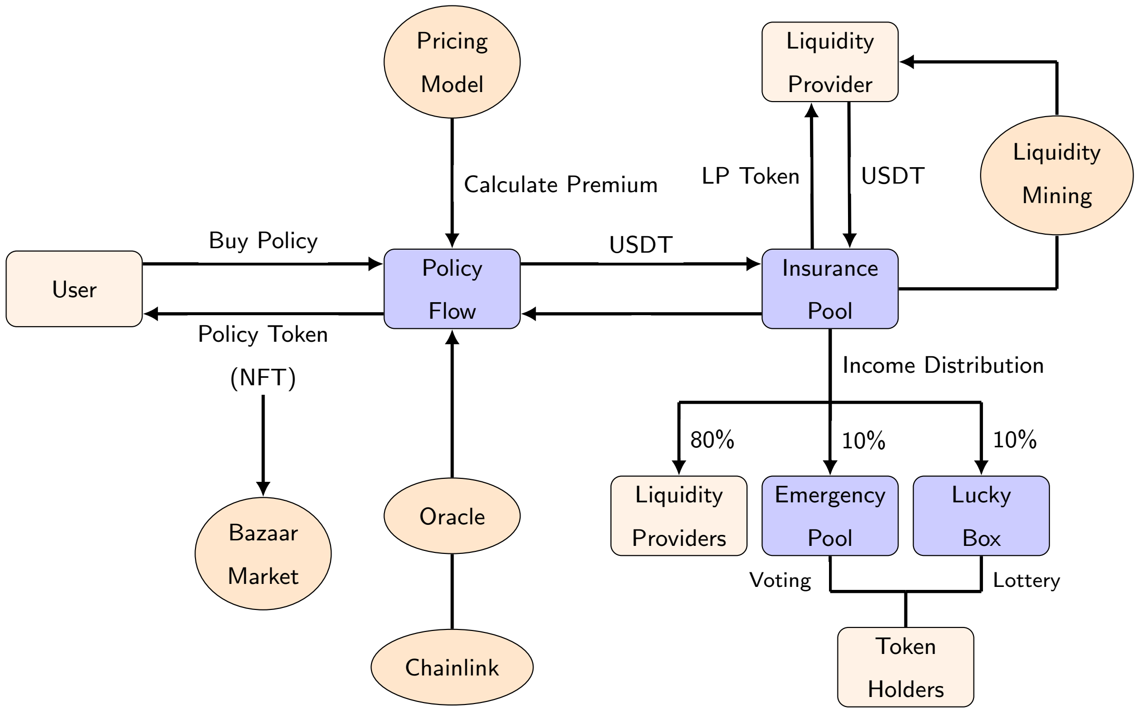

Figure 2.1: Workflow of the DEGIS platform

2.1 Supply an insurance

2.1.1 Why supply on DEGIS

- Easy to enter, easy to earn.
- Permissionless to enter and exit.
- Stable premium income based on statistics.
- Additional token incentives.

2.1.2 How to supply

As an insurance provider, or to say, liquidity provider on DEGIS, you can get quick, upfront, and stable premiums from buyers through staking your money into the insurance pool, then publishing and selling insurance policies by the platform itself. There is no minimum entry requirement, and you are free to exit at any time to withdraw all your liquidity. Profit/Loss will be settled at the time of withdrawal, which is proportional to the staking time and amount.

2.1.3 Risks and Benefits

Pooled liquidity is designed to reduce risks for LPs, big data analysis and advanced machine learning models help ensure stable income. In the next section we provide some numerical simulation results for reference.

2.2 Buy an insurance

2.2.1 Why buy on DEGIS

- Fair, transparent and automated.
- No central controller and disputing about policies.
- Simple operation and instant settlement.
- Rational price and significant payoff.

2.2.2 How to buy

Find our products(both on-chain and off-chain products are available on DEGIS from the website (degis.io), choose the type of insurance product you'd like, and finally confirm with just one simple click! The payoff will be directly sent to your wallet if insurance conditions get met in the future. No other actions are required.

2.2.3 Risks and Benefits

Users should be well aware of the risks of insurance products before buying them, and we would offer a relatively low price and high payoff to attract insureds in need.

3 Concentrated Liquidity

Different kinds of insurance products will be backed by one same liquidity pool on DEGIS. Liquidity providers enter this pool to provide insurance services and earn premiums.

3.1 Flight Delay Insurance

3.1.1 Payout

Unlike traditional insurance, which sets a fixed payout, DEGIS provides users with a continuous and increasing payout when delay time does exceed the threshold. Currently, the threshold is fixed at 30 (minutes), while users may set their preferred one in the short future. Detailed payout policy is stated in the following, where *threshold* as 30, t is the delay time in minutes and currency unit in USD.

$$Payout = \begin{cases} 0 & t < threshold \\ \frac{1}{480} \cdot t^2 & threshold \leq t < 240 \\ 120 & t \geq 240 \end{cases} \tag{3.1}$$

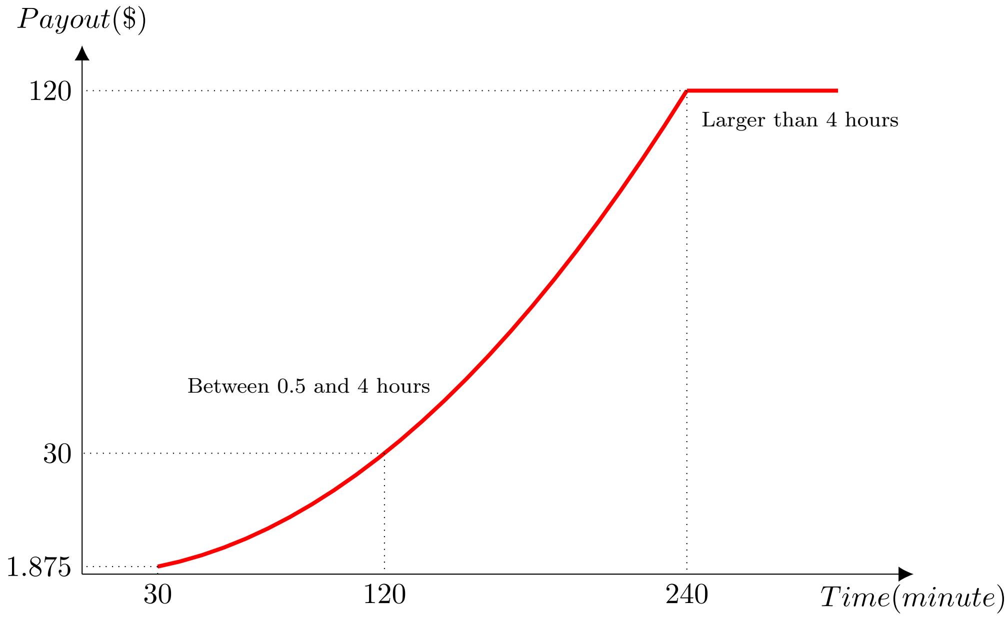

Figure 3.1: Function graph of payout amount and delay time

3.1.2 Pricing

DEGIS uses self-developed machine learning models to predict the expected payout of each flight, based on departure time, weather, airlines, airports, distances, associated flights, and many other features. Basic price will be the maximum between the predicted payout and 4.2 dollars. Which means, different flights may have different basic prices, but the majority of them will be 4.2 dollars.

We first divide the delay time into several intervals, leading to an ordered classification problem, and train gradient boosted regression trees with a squared loss penalty [1] $H'(\mathbf{x}_i) = \sum_{t=1}^{T} h_t(\mathbf{x}_i)$, where each function $h_t(\cdot)$ is a limited-depth CART tree. We then apply the mapping $\mathbf{x}_i \to \phi(\mathbf{x}_i)$ to all inputs, where $\phi(\mathbf{x}_i) = [h_1(\mathbf{x}_i), \ldots, h_T(\mathbf{x}_i)]^\top$, and refer these CART trees as *weak learners*. To solve our final prediction problem, we learn the weighted-vector β by minimizing a convex empirical risk function $\ell\left(\phi(\mathbf{x}_i)^\top \boldsymbol{\beta}, y_i\right)$ with l_1 regularization, $|\boldsymbol{\beta}|$. In addition, we incorporate a cost term $c(\boldsymbol{\beta})$, which we derive in the following statement, to restrict test-time cost. The combined test-time cost loss function is

$$\mathcal{L}(\boldsymbol{\beta}) = \underbrace{\sum_i \ell\left(\phi(\mathbf{x}_i)^\Gamma \boldsymbol{p}, y_i\right) + \rho\,|\beta|}_{\text{regularized risk}} + \underbrace{\lambda\; c(\beta)}_{\text{test-cost}}$$

There are two factors that contribute to the test-time cost of each classifier: the weak learner evaluation cost of all active $h_t(\cdot)$ (with $|\beta_t| > 0$) and the feature extraction cost for all features used in these weak learners. Define an auxiliary matrix $\mathbf{F} \in \{0,1\}^{d\times T}$ with $F_{\alpha t} = 1$ if and only if the weak learner h_t uses feature f_α. Let $e_t > 0$ be the cost to evaluate a $h_t(\cdot)$, and c_α be the cost to extract feature f_α. With this notation, we can formulate the total test-time cost for an instance precisely as

$$c(\boldsymbol{\beta}) = \underbrace{\sum_t e_t \|\beta_t\|_0}_{\text{evaluation cost}} + \underbrace{\sum_\alpha c_\alpha \left\| \sum_t |F_{\alpha t}\beta_t| \right\|_0}_{\text{feature extraction cost}}$$

To take insurance risks into account, we define $\tau = \mathbb{E}\left(\frac{L}{e}\right)$ where L is the loss and e the period for which the insurance is valid(exposure). Assume that the size of the claims is independent of the claim frequency and orders' arrival (denoted as $A_t, a\ Poission\ Process$), we make follwoing expansion:

$$\tau = \mathbb{E}\left(\frac{L}{e} \mid A_t\right) = \mathbb{E}\left(\frac{L}{N} \mid N > 0\ , A_t\right) \times \mathbb{E}\left(\frac{N}{e} \mid A_t\right) := \mathbb{E}\left(S\right) \times \mathbb{E}\left(F\right)$$

where N is the number of claims, S the severity, or size of the claim and F the claim frequency(conditioned on agent behaviors).

Combination of all these to obtain our actual optimization problem (3.2)

$$\operatorname*{argmin}_{\beta} \quad \underbrace{\sum_t \mathbb{E}\left(S|A_t,\beta\right) \times \mathbb{E}(F|A_t,\beta) + \rho\ |\beta_{\mathbf{t}}|_0}_{\text{regularized risk}} + \underbrace{\sum_t e_t\,|\beta_t|_0 + \sum_\alpha c_\alpha \sum_t |F_{\alpha t}\beta_t|_0}_{\text{test-cost}} \tag{3.2}$$

3.1.3 Prohibit Inside Trading

A total of 500 insurances are available for sale on each flight episode, and in order to combat insider traders and malicious attackers, also protect the interests of ordinary consumers. Meanwhile, we have designed a price floating system with three rules:

- For a certain flight, the price increases 2% for every insurance purchased(except for the first 20).
- For a certain flight on a certain day, the price increases 2% for every insurance purchased(except the first 5).
- For flights departing within 24 hours, the price increases 5% for every insurance purchased.

The three rules work together, and product this modifier to basic price results to its final price. Such float-pricing system is designed to limit the operating space of inside traders. All kindly users are not expected to be affected, thus some exemptions(from all three rules) are provided accordingly:

- Users who upload their identity information and ticket vouchers can be exempted.
- Users holding 1,000 + DEGIS tokens for 3+ months are exempted.

3.2 Token Price Insurance

The crypto market has a large variation, and investors may suffer a huge loss within a short period. To help investors hedge the extra risk, DEGIS issues a new kind of insurances upon token price fluctuation. There are two kinds of token price insurance.

- *cover price decreasing*
- *cover price increasing*

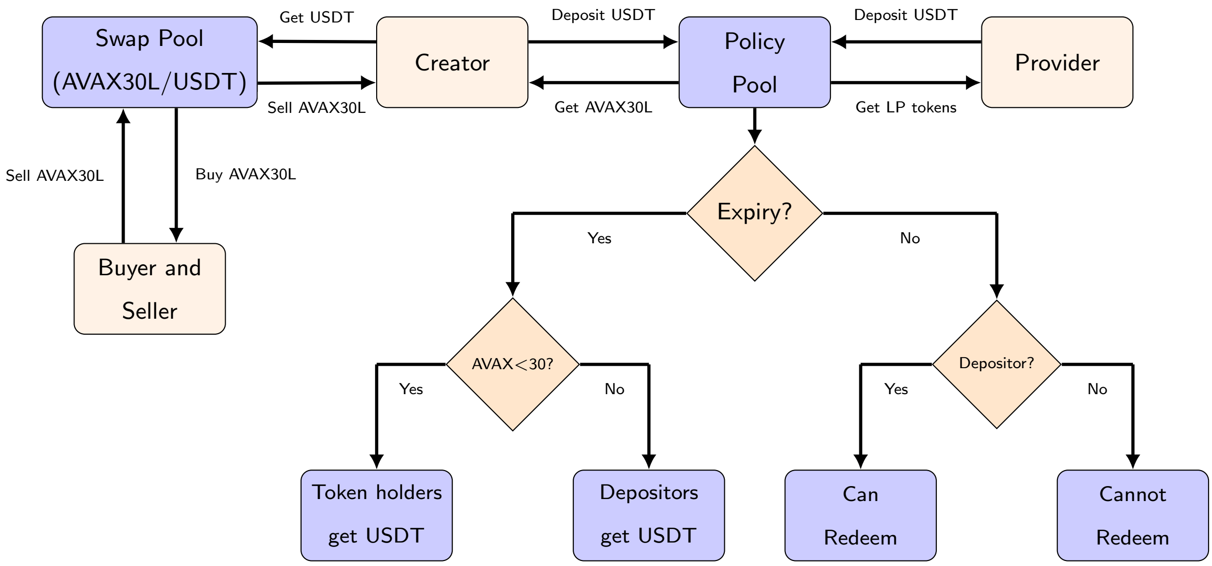

Figure 3.2: Workflow of the insurance token system

3.2.1 Insurance tokens

To mint the insurance token for a certain event e.g. AVAX goes below \$30, one needs to make a full mortgage, which means using one USDT to mint one insurance token. The USDT are staked in the policy pool. The minted insurance tokens are ERC-20 tokens and can be exchanged and transferred.

The staked USDT in the policy pool are controlled by smart contracts. If the certain event happens at the expiry date,then the staked USDT can be redeemed by the insurance token holders. Since the only way to mint insurance tokens is to stake the same amount of USDT, holding each insurance token will surely get 1 USDT if the certain event happens. If the certain event does not happen at the expiry date, the staked USDT will be sent back to depositors automatically.

Before the expiry date, depositors have right to redeem their USDT by burning the insurance tokens which is called striking. The maximum total amount of striking is the total amount of insurance tokens that the depositor have ever minted.

There is a set of naming methods for insurance tokens. For example, AVAX30L2201 means that this is an insurance token for AVAX and can get payout if it is lower than \$ 30 at 1st

January 2022. One kind of insurance tokens can star to be minted two months before the expiry date.

3.2.2 Participants

Depositors deposit USDT into the policy pool and can sell these tokens in our swap pool to gain profits. If the certain event does not happen at expiry date, these profits will be the final earnings. If the event does happen, they will lose their originally staked money. They can also repurchase the insurance tokens after selling, and redeem their staked USDT.

Liquidity providers directly put their USDT in the mining pool. Smart contracts will automatically mint insurance tokens for liquidity providers according to the current market ratio in the swap pool and then stake the minted insurance tokens as well as the remained USDT into the swap pool to earn liquidity rewards. When liquidity providers want to withdraw, smart contracts will help them to redeem the minted insurance tokens, and they can get all deserved assets in USDT. If the deserved amount of insurance tokens are larger than the originally minted amount, the extra amount will be sold automatically, since the liquidity provider can not use the extra amount of insurance tokens to redeem USDT. If the deserved amount of insurance tokens are smaller than the originally minted amount, liquidity providers can use the right of redeeming to arbitrage themselves.

Hedgers can use USDT to buy insurance tokens in the swap pool which can be regarded as insurances. They can get the corresponding amount of payout if the certain event happens.

3.2.3 Swap

The swap pool is an AMM pool like Uniswap V2. When someone wants to swap tokens in the pool, the product of the amount of tokens on two sides remains unchanged [2]. The transaction fee will be charged when the trader putting the tokens in the one side of the swap pool, which is 2 %. Transaction fee will be used to reward liquidity providers.

We will be the first one to provide liquidity for both sides based on our pricing model, then the pool will star running and the price will be decided by the market.

4 Simulation

To verify the effectiveness of our pricing methods, we designed a backtesting system, an agent-based one, for simulating the actions and interactions of autonomous agents to understand the behavior of a system and what governs its outcomes. Three different types of agents are considered here:

- Random Trader: enter this insurance pool randomly to buy products at given probability.
- Rational Trader: ask for insurances based on his own prediction model.
- Attacker: entitled some certain ability to know the future information, and would buy if current price falls below the expected payout.

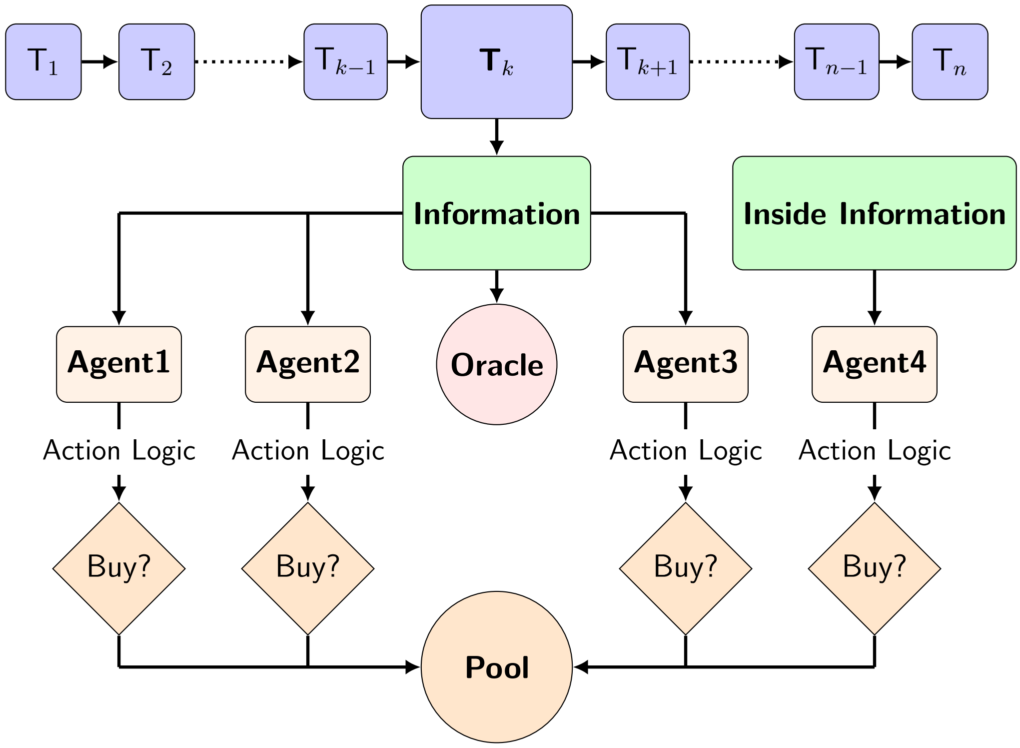

Figure 4.1: Schematic diagram of the back test system. The back test system uses historical data in time series to simulate transactions. All normal agents and the oracle get only the published information, e.g. airline timetable, weather. Inside traders know the exact delay time. Agents will make decisions depending on the information they get and the price that the oracle predicted. If they decide to buy, the order will be put in the pool.

Let us take the flight delay insurance as an example to explain further. Under this specific market settings, agents can be described as:

- Random trader: buy with a *base probability* chosen from [0.4, 0.8], and actual buying probability would be adjusted according to insurance settings.

$$Actual\ Probability \ = Base\ Probability \times \frac{Days\ to\ Depart}{7} \times \frac{1}{Insurance\ Price}$$

  i.e. the closer to the departure date and the more expensive ticket will lower the possibility of purchase, which seems to be reasonable.

- Rational trader: use one bench model to help decide. If the insurance's expected payout exceeds its price, then it's a deal. Rational users adopt bench model or some smarter ones, while the pricing system uses embedded model. Rational users also would begin trading 7 days before departure.

- Attacker: entitled some certain ability to know the true depart delay, and would buy if current price falls below the expected payout. *Note that attackers can only buy* 3 *days before departure*. The prediction is generated from the uniform distribution of following intervals:

$$[\text{TrueValue} \times \mathit{ability},\ \text{TrueValue} \times (2 - \mathit{ability})]$$

By adjusting the ratio of attackers to the total number of buyers, we can see the model's performance in different environments. In the following, we present our backtest setup and simulation result.

4.1 Data Description

Historical flight data is collected from the Bureau of Transportation Statistics, USA, including airline conditions, flight summary, and time information. The simulation described here involves all American flights information from Jan 2020 to Jun 2021. Our backtest starts from Jan 2021 to not overlap with the model training part, and the pricing model updated accordingly over time.

4.2 Basic Statistics

1. Put 5,000 USD in liquidity pool at the very beginning.
2. Set a 2:1 ratio of buyers(RandomTrader & RationalTrader) versus attackers in the market.
3. Run the simulation.

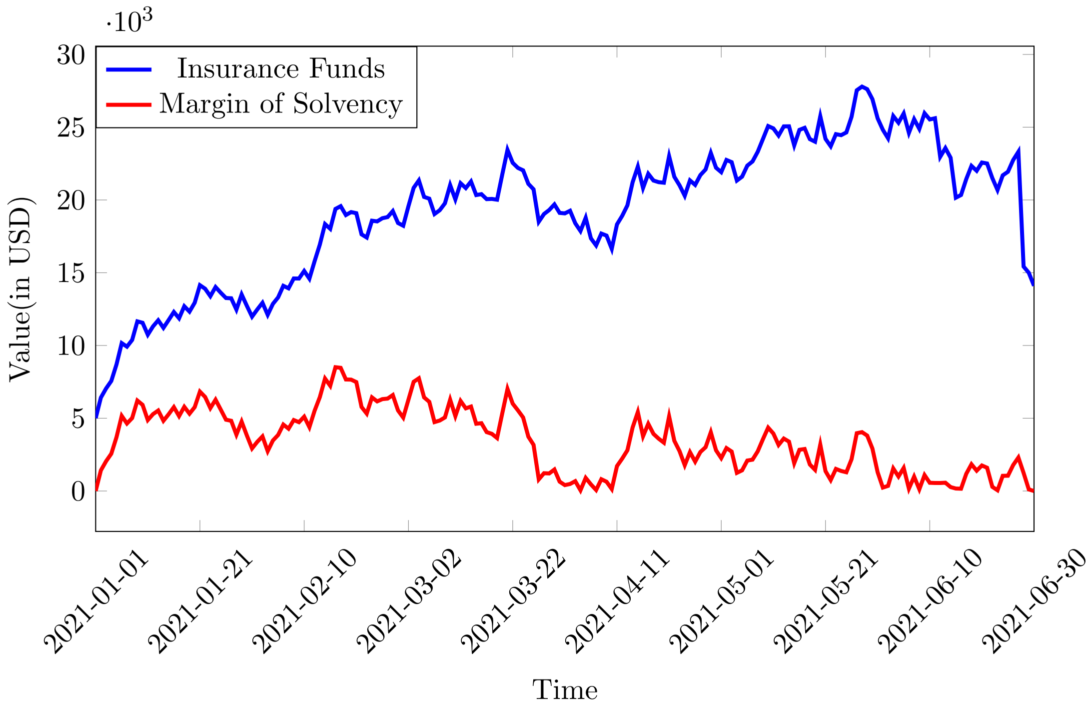

Figure 4.2: Insurance Funds and Margin of Solvency in pool

The chart (4.2) shows *Insurance Funds** & *Margin of Solvency** condition in pool, from which we can conclude that our insurance pool can grow steadily even with attackers engaged.

Margin of Solvency: margin to meet all potential future payout.*

Insurance Funds: all capital in pool, including liquidity, undistributed profits, and margin of solvency.*

And several important indicators for insurance product are calculated to be:

- Net Profit: 9,100
- Return on Equity: 64.54%
- Claim paying ability: 187%
- Annual Percentage Rate: 167%

4.3 Prohibit Inside Trading

(4.3) shows the gross profit ratio of insurance products under different attacker ratio, and our break-even line is 50%.

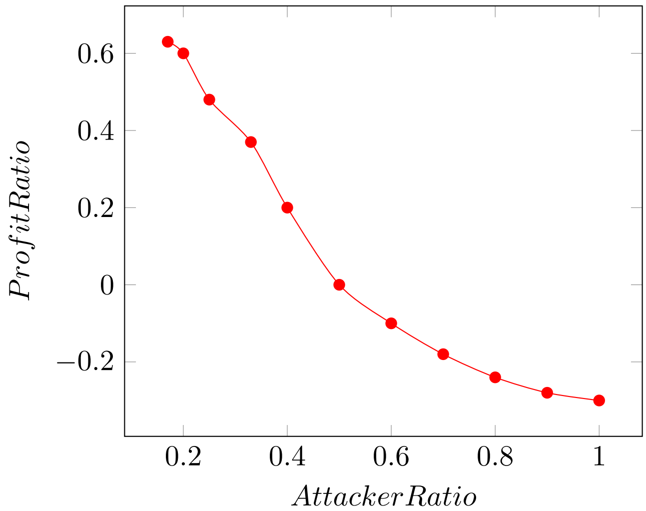

Figure 4.3: Relationship of profit ratio and attacker ratio. Because of the floating price policy, we have a fixed maximum loss on each delay. Even if all the agents are attackers, our loss will not exceed 30% of the pool.

5 Highlight Modules

Both NFT and GameFi features are entitled here on DEGIS to provide users with more practical and exciting functionality. The secondary market is also set to increase liquidity for DEGIS insurance products.

5.1 NFT Characteristics

At any stage, our insurance product will be packaged as NFT(Non-fungible token), with users' basic information on it. Once submit the insurance order, the corresponding NFT will be mint and sent to the user's wallet directly. On the one hand, such non-fungible tokens have the connotation of insurance service with certain intrinsic values. On the other, the NFT will not be burnt if backing insurance does not get expired, users can keep it for collection and would be able to exchange discounts for other DEGIS designs in the future.

5.2 Lottery

Recurrently held and lossless lottery games are set to benefit token holders on DEGIS. By staking a small fixed amount of DEGIS tokens, users can get one ticket for the game, inside which there is a 4-digit number. The lucky game pool receives the premium income from the insurance pool. Every week, there would be a new round, and users can join several rounds with the same number without withdrawing and depositing again. Winners can get

$$Winner\ Reward = Tickets\ Num\ \times \frac{Total\ Reward}{Total\ Tickets}$$

as rewards. The final remaining premium will be added to the next round.

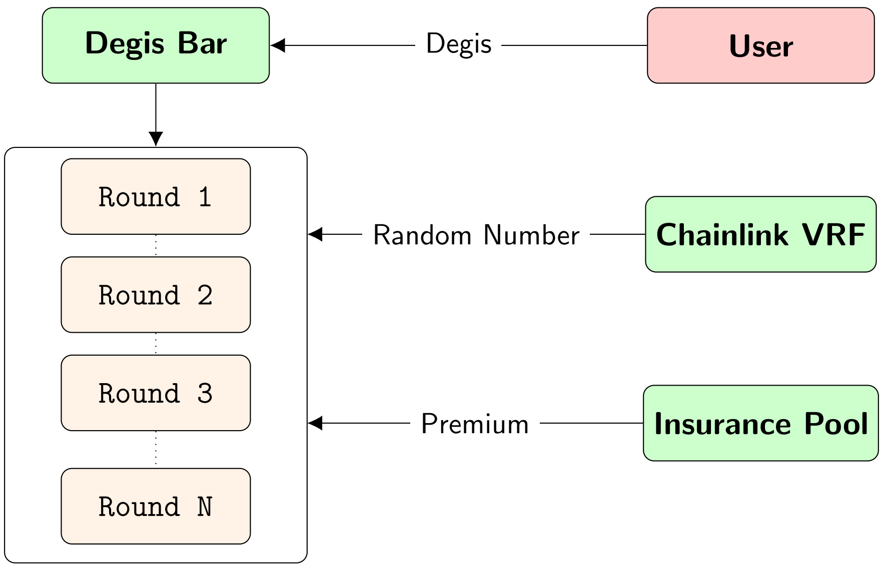

Figure 5.1: Recurrent lossless lottery game for DEGIS holders

5.3 Secondary Market

Secondary markets are set up on DEGIS for users to trade with each other. Each insurance sold will exist as an NFT, free to trade on any NFT marketplace like OpenSea. Also, we would set our DEGIS's own insurance exchange/trading market shortly. Transactions will be performed in an order book, where buyers and sellers make quotations, and DEGIS helps match automatically.

6 DEGIS Token

DEGIS token is one kind of ERC-20 token newly introduced here on DEGIS.

6.1 Utility

People holding DEGIS tokens can enjoy many rights on our platform.

- Users can stake their tokens in staking pool to participate into premium sharing.

$$\begin{aligned} Premium = 80\%\ LP + 10\% Emergency\ Pool \\ + 10\% Token\ Holder \end{aligned} \tag{6.1}$$

  Part of the premium income(10%) is divided into a separate pool and allocated to participants according to their share and staking time.

- Users can stake to get tickets for the lossless lottery game as introduced in Section 4 B.
- Users would get a discount when purchasing insurances for themselves after staking required amount of DEGIS tokens.
- DEGIS token servers as governance token on our platform.

6.2 Tokenomics

DEGIS(DEG) is the protocol token of DEGIS and its total supply is 100 million. Detailed distribution proportion is designed as follows:

- Purchase Incentives: 40%
  40 million DEG in total, released for buyers who purchased policy but didn't get a payoff. Only released in 6 months for the first two products, and released in 3 months for future products. If not totally used, theses tokens will go to the reserved part and wait for the community decision.
- Liquidity Reward: 15%
  15 million DEG in total, release for providers. Release 5 million DEG in 12 months for first two products and 10 million for the future products.
- Growth Fund: 10%
  10 million DEG in total, used for marketing and brand building.
- Team: 10%
  10 million DEG in total, locked for 6 months once DEGIS launched, then 12 months for linear releasing.
- Investors: 20%
- IDO(Public Sale): 2%
  See next section for details.
- Reserved: 3%
  3 million DEG in total, used for emergency events(decided by the community).

7 Data Oracle

7.1 Data Source

7.1.1 Flight Delay

We have multiple reliable data sources for flight information query services and status updates, sources including FlightRadar24, FlightStats and Bureau of Transportation Statistics, USA.

7.2 Chainlink Oracle

Parametric insurance built on top of decentralized infrastructure is one of the most exciting frontiers of innovation in the insurance industry. Not only does this insurance model foster new product designs and increase market participation, but it redefines underlying trust dynamics between insurance providers and policyholders.

Here we use one three-party trustworthy decentralized data oracle service, named Chainlink to update insurance status(e.g. flight delay status) information. All claims are transparent on-chain and settled fairly since there would be no information asymmetry.

Chainlink is a decentralized oracle network that gives smart contracts secure and reliable access to data providers, web APIs, etc. We are setting and operating Chainlink nodes to fetch data from trusted web APIs for status-related data.

References

[1] Friedman, J.H. Greedy function approximation: a gradient boosting machine. The Annals of Statistics, pp. 1189–1232, 2001.

[2] Connor Martin, Callil Capuozzo, The Uniswap V2 Protocol

[3] Ravi Myneni, The Pricing of the American Option, The Annals of Applied Probability Vol. 2, No. 1 (Feb., 1992), pp. 1-23

[4] Mark Broadie, Jerome Detemple, American Option Valuation: New Bounds, Approximations, and a Comparison of Existing Methods, The Review of Financial Studies, Volume 9, Issue 4, October 1996, Pages 1211–1250