Adaptive Risk Coverage (ARC) Framework - A dynamic approach of Next-Order Value Aggregation and Asymmetric Truth Hedging

Muhammad Idrees

November 2024

1 Introduction

The **Adaptive Risk Coverage (ARC) Framework** is a transformative system for building dynamic and resilient **coverage pools** that combine **truth hedging** with **value aggregation**. Engineered with the **Logarithmic Market Scoring Rule (LMSR)** at its core, ARC ensures inherent liquidity, scalable risk management, and intelligent value alignment across diverse applications.

Key features of the ARC Framework include:

- **Inherent Liquidity**: The LMSR model ensures markets remain liquid without requiring external liquidity providers, facilitating smooth participation and settlement.
- **Dynamic Pricing**: The cost function adjusts pricing based on market activity, reflecting real-time sentiment and preventing over-concentration on single outcomes.
- **Worst-Case Loss Coverage**: The framework guarantees payout feasibility by pre-calculating worst-case loss scenarios, ensuring participant trust and market stability.
- **Share Trading Mechanism**: Participants can dynamically trade outcome shares, with prices determined by the LMSR cost function, ensuring fair and adaptive odds.
- **Oracle Integration**: Modular oracle systems enable secure and customizable outcome verification, supporting trustless and accurate event settlement.
- **Incentive Mechanisms**: ARC employs reward systems that include activity-based incentives, issue resolution bonuses, and early participation rewards to drive meaningful engagement and truthful risk hedging.
- **Tokenomics**: The native ARC token underpins the ecosystem, driving participation, incentivizing value creation, and accruing long-term value through redistribution of transaction fees and controlled emissions.

What sets ARC apart is its innovative approach to **truth hedging**, which incentivizes accurate risk forecasting and fair resource distribution. Through its native token, **ARC**, the framework fosters robust participation, aligns stakeholder incentives, and accrues value over time via fee redistribution and token emissions.

By integrating modular oracles, dynamic pricing, and worst-case loss guarantees, the ARC Framework establishes a next-generation model for risk management. It empowers ecosystems to aggregate intelligence, mitigate risks, and drive impactful decision-making across industries.

2 System Architecture and Components

ARC Framework leverages a Logarithmic Market Scoring Rule (LMSR) model to dynamically adjust the costs and odds based on user participation, creating a responsive, self-regulating market. Key components of ARC Framework include a LMSR Module Manages market creation, share trading, and settlement using the LMSR cost function and oracles Trusted entities that verify event outcomes and ensure data accuracy.

2.1 LMSR Markets Module

2.1.1 LMSR Cost Function

In ARC Framework's LMSR-based module the cost function, $C(q)$, determines the price for purchasing shares in a particular outcome and adjusts as more bets are placed. This function ensures liquidity and dynamic pricing within the market.

The cost function for LMSR is defined as:

$$C(q) = b \cdot \ln\left(\sum_{i=1}^{n} e^{\frac{q_i}{b}}\right)$$

where:

- $C(q)$ is the total cost required to reach a state q, where qi is the total amount bet on outcome
- b is a liquidity parameter, which controls the sensitivity of prices to new bets (higher b means greater liquidity and less responsive odds).
- n is the number of outcomes.

This function dynamically increases costs as the demand for a particular outcome grows, preventing market manipulation and creating a balanced price discovery.

- **Dynamic Pricing:** The cost of betting increases with the total amount staked on a particular outcome, ensuring that odds for heavily favored outcomes adjust to discourage overloading.
- **Market Liquidity:** Implicit liquidity is provided through the cost function without external liquidity providers.
- **Initial Liquidity (Worst-Case Loss):** To guarantee that the market can cover potential payouts, we calculate initial liquidity or the worst-case loss, denoted as $Lmax$, based on the parameter b and initial conditions.

The worst-case loss can be calculated as:

$$L_{\max} = b \cdot \ln(n)$$

where n is the number of outcomes. This value represents the maximum amount of funds required to cover all possible payout scenarios.

- **Dynamic Odds:** The odds for each outcome i are calculated based on the relative stakes:

$$p_i = \frac{e^{\frac{q_i}{b}}}{\sum_{j=1}^{n} e^{\frac{q_j}{b}}}$$

2.1.2 Share Trading

When a user buys shares in an outcome, the cost of shares is calculated as the difference between the current and previous states of the cost function. $q_{current}$ represent the current total stake in each outcome and q_{new} represent the new stake after the user places a bet. The cost of buying additional shares for an outcome i is:

$$\text{Cost} = C(q_{\text{new}}) - C(q_{\text{current}})$$

where $C(q_{new})$ is evaluated with the updated stakes. This cost calculation ensures that as more funds are allocated to an outcome, the cost of further shares in that outcome increases.

The number of shares received by a users, when they place a bet of amount ΔC on outcome i is given by:

$$s = \frac{\Delta C}{p_i}$$

where p_i is the price per share for outcome i, derived from the LMSR function.

- **Impact of Bet Placement:** As more bets are placed on an outcome, p_i adjusts to reflect market sentiment, discouraging disproportionate betting on a single outcome by increasing its cost.
- **Share Distribution**: For a user placing a bet ΔC on outcome i, they receive outcome tokens calculated based on current market odds.
- **Market Pricing Feedback:** Each bet placed feeds back into the LMSR function, adjusting costs for future bets and providing real-time odds adjustments.
- **Redemption for Winning Outcome:** Upon event conclusion, holders of the winning outcome tokens can redeem them as per the formula above, ensuring fair distribution based on the amount staked on the losing side.

2.1.3 Market Settlement

After the event concludes and an outcome is verified by oracles, users holding winning shares can redeem their tokens for payouts based on their share relative to the total pool.

Payout Calculation: In LMSR, each share of the winning outcome can be redeemed at a fixed payout of 1 unit.

$$P_{\text{user}} = s_{\text{win}} \times (1\text{usd})$$

where: s_{win} is the number of shares the user holds in the winning outcome.

2.2 Oracles

In ARC Framework's platform, oracles are critical components responsible for determining the outcome of markets. They are whitelisted entities whose addresses are specified at the time of market creation. The oracles provide the necessary data to settle the outcome of each market accurately and securely. Once the market event concludes, the designated oracle is responsible for submitting the outcome to the platform's smart contracts. The process is as follows:

1. **Event Monitoring:** The oracle monitors the event associated with the market until it reaches a conclusion.
2. **Outcome Determination:** The oracle determines the correct outcome based on verifiable data.
3. **Outcome Submission:** The oracle submits the outcome to the smart contract using its whitelisted address.
4. **Market Settlement:** Upon receiving the outcome, the smart contract settles the market, allowing participants to redeem their winnings.

Category	Allocation	Vesting
Incentive Pool	38%	-
Liquidity	10%	-
Private Sales	12%	1 year with 3 months cliff
Team	17%	1 year with 6 months cliff
Treasury	23%	-

Table 1: Initial Token Supply Allocation

2.2.1 Oracle Implementation

When a market is created, the market creator specifies the address of the oracle that will be responsible for settling the market outcome. The oracle can be:

- **An Individual Wallet:** A single entity controlling a wallet address can act as the oracle, providing outcome data directly.
- **A Multisignature Wallet (Multisig):** A group of entities can collectively act as the oracle through a multisig wallet, requiring multiple approvals to submit the outcome.
- **A Smart Contract:** An autonomous contract can serve as the oracle, programmatically determining the outcome based on predefined conditions or external data feeds.

By allowing market creators to specify the oracle at the time of market creation, ARC Framework provides flexibility to tailor the oracle mechanism to the specific needs of each market. This enables:

- **Customization:** Market creators can choose oracles that are best suited for the event type or have expertise in the relevant domain. Oracles can implement any custom logic of quorum and dispute resolution and stake required based on type of markets
- **Diversity of Oracles:** With the flexibility of selecting oracles at market creation, market creators can implement any custom logic of oracle consensus. oracles can have their dispute resolution mechanism, quorum or any other custom logic of submitting outcomes

3 Token Economic Model

ARC Framework's native token (ARC Token) is integral to the platform's operation, serving the purpose of incentivizing market participation . This section outlines the token's utility, distribution model, economic incentives, and its expected impact on value over time.

3.1 Token Utility

Reward users for valuable participation, encouraging meaningful information contribution and highlighting critical issues. Activity-based rewards, issue resolution bonuses, and early participation bonuses are distributed in ARC tokens.

3.2 Token Supply

- **Initial Supply:** 30,000,000 ARC tokens
- **Supply Type:** Inflationary

3.3 Token Emissions

In addition to the initial supply, ARC tokens are designed to be inflationary to provide ongoing incentives for participants. Emissions are influenced by:

- **Number of Active Markets Settled (NAM):** Total markets settled during the epoch.
- **Number of Active Participants (NAP):** Unique participants who placed bets during the epoch.
- **Total Betting Volume (TBV):** Total USDC bet during the epoch.
- **Number of Issues Resolved (IR):** Issues resolved during the epoch.
- **Base Emission Rate (BER)/epoch**: 0.08%.
- **Maximum Emission Rate**: 0.67%

Activity Metrics and Weightings: NAM, NAP, TBV, IR.

- w1: Weight for NAM
- w2: Weight for NAP
- w3: Weight for TBV
- w4: Weight for IR

Weights should sum up to 1:

$$w_1 + w_2 + w_3 + w_4 = 1$$

Applied Emission Rate in epoch:

$$AER = BaseEmissionRate(BER) + (AdjustmentFactor(AF))$$

Adjustment Factor:

$$AF = (w_1 ln(NAM + 1) + w_2 ln(NAP + 1) + w_3 ln(TBV + 1) + w_4 ln(IR + 1))k$$

- **k:** Scaling constant to ensure the adjustment factor is in percentage terms.
- **Natural Logarithm (ln):** Used to dampen the effect of large numbers and keep the function manageable.
- Set a **Maximum Emission Rate** to prevent excessive inflation:

$$AER_{final} = min(AER, MaxEmissionRate)$$

3.4 Transaction Fees

Transaction fee is charged on each bet placed or shares sold, collected in USDC. Fee rate is specified by the market creator within platform-defined limits.

Fee collected by the markets is distributed among:

- **Token Holders (30%):** Fees are converted to ARC tokens and distributed to token holders as passive income.
- **Market Risk Pool (50%):** Reinvested in a market risk pool for potential payouts and the market maker's worst-case loss.
- **Oracle Rewards (20%):** Paid to oracles for settling market outcomes.

Incentive Category	Allocation from epoch emisisons
Activity Reward Pool (ARP)	30%
Issue Resolution Reward Pool (IRRP)	70%

Table 2: Emission Allocation

4 Incentive Mechanism Design

The incentive mechanisms are designed to:

- Encourage active and meaningful participation
- Promote the resolution of critical issues

There are two types of incentives in the system:

- **Initial Supply Incentive Pool:** These are incentives for network bootstrap phase which are allocated from initial token supply
- **Ongoing Token Emissions:** Ongoing token emissions are distributed among network participants based on their activity

4.1 Incentive Pool Distribution

4.1.1 Early Participation Bonuses

- **Purpose:** Encourage early market engagement.
- **Eligibility:** Users who place bets within the first 10% of the market duration.
- **Bonus:** Additional 10% on their activity-based rewards.

4.1.2 Airdrops (Optional)

4.2 Token Emission Distribution

The incentives, from the ongoing emissions, are distributed as follows:

4.2.1 Activity-Based Rewards

- **Purpose:** Encourage higher participation.
- **Distribution:** Proportional to the user's adjusted betting activity.
- **Total Incentives:** 30% of Token emissions for that epoch.

$$R_i^{\text{Activity}} = \frac{B_{i,\text{adjusted}}}{\sum_j B_{j,\text{adjusted}}} \times ARP$$

- $B_{i,adjusted} = B_i^a$: Adjusted bet amount using quadratic weighting.
- ARP**:** Activity reward pool (30% of epoch emissions).

4.2.2 Issue Resolution Reward

- **Purpose:** Align incentives with issue resolution.
- **Trigger:** Distributed when a market's underlying issue is resolved.
- **Eligibility:** Users who bet on the market for which issue is resolved.
- **Criteria:** Same as activity based reward based upon the adjusted bet amount

5 Appendix

5.1 Configurable Parameters

- **Emission Epoch:** Monthly
- **Base Emission Rate (BER)**: 0.083%
- **Maximum Emission Rate**: 0.67%
- α**:** Exponent of quadratic weighting function set to 0.5
- **k:** Scaling constant to ensure the adjustment factor is in range.
- Weighting Parameters
  - **w1 (Weight for NAM)**
  - **w2 (Weight for NAP)**
  - **w3 (Weight for TBV)**
  - **w4(Weight for IR)**

5.2 Incentive Alignment - Game Theoretic Analysis

5.2.1 Agents

Participants: Users who place bets in the markets.

Market Creators: Entities that create markets.

Oracles: Entities responsible for resolving market outcomes.

5.2.2 Behaviors

Participants: Decide on the amount to bet Bi, when to bet, and on which outcome.

Market Creators: Set market parameters, including fees.

Oracles: Choose to report outcomes truthfully or dishonestly.

5.2.3 Utility Functions

For participants, the utility function can be defined as:

$$U_i = \mathbb{E}[\text{Payout}_i] + R_i - C_i$$

Where:

- $E[Payout_i]$: Expected monetary gain from bets.
- R_i: Rewards from the incentive mechanisms.
- C_i: Costs incurred (bet amounts, fees, effort).

5.2.4 Activity-Based Rewards

Participants receive rewards based on their adjusted bet amounts:

$$R_i^{\text{Activity}} = \frac{B_{i,\text{adjusted}}}{\sum_j B_{j,\text{adjusted}}} \times ARP$$

Where:

- $B_i, adjusted = B_i^a$**:** Adjusted bet amount using quadratic weighting.
- **ARP:** Activity reward pool (30

Proportional Rewards: Rewards are distributed proportionally based on the adjusted bet amounts.

Diminishing Returns: The quadratic weighting ($\alpha = 0.5$) introduces diminishing returns for larger bets, discouraging participants from monopolizing rewards through large bets.

Objective: Maximize U_i by choosing B_i

Participants face a trade-off between increasing Bi to potentially earn higher payouts and rewards, and the diminishing returns due to quadratic weighting.

First-Order Condition: To find the optimal Bi, we take the derivative of Ui with respect to B_i and set it to zero:

$$\frac{\partial U_i}{\partial B_i} = \frac{1}{2\sqrt{B_i}} \left(\frac{IRRP}{\sum_j B_{j,\text{adjusted}}} - \frac{B_{i,\text{adjusted}} \times IRRP}{\left(\sum_j B_{j,\text{adjusted}}\right)^2} \right) + \frac{\partial \mathbb{E}[\text{Payout}_i]}{\partial B_i} - 1 = 0$$

This equation illustrates how participants consider both the adjusted rewards and the costs when deciding Bi To find the optimal Bi, we take the derivative of Ui with respect to Bi and set it to zero:

- The derivative is positive for small Bi and decreases as Bi increases.
- There is a point where increasing Bi further provides negligible additional rewards due to diminishing returns.

In a non-cooperative game setting:

- **Nash Equilibrium**: Each participant chooses Bi^* such that no one can increase their utility by unilaterally changing their bet.
- **Result**: The quadratic weighting leads to an equilibrium where bets are more evenly distributed among participants.

5.2.5 Issue Resolution Rewards

Participants betting on the correct outcome receive bonuses:

$$R_i^{\text{Issue}} = \delta_{i,\text{win}} \times \left(\frac{B_{i,\text{adjusted}}}{\sum_{j \in \text{winners}} B_{j,\text{adjusted}}} \times TIR \right)$$

Where:

- $\delta_{i,win} = 1$ if participant i bet on the winning outcome, else 0.
- IRRP: Issue Resolution Reward Pool (60% of the Incentive Pool).

Participants aim to maximize U_i by:

- Accurately predicting the outcome (increasing $\delta_{i,win}$)
- Balancing bet amounts considering quadratic weighting and probability of winning.

Expected Utility The expected utility for participant i is:

$$\mathbb{E}[U_i] = P_{\text{win}} \times \left(\mathbb{E}[\text{Payout}_i^{\text{win}}] + R_i^{\text{Issue}} \right) + P_{\text{lose}} \times \mathbb{E}[\text{Payout}_i^{\text{lose}}] - C_i$$

Where:

- P_{win} is the probability of the outcome i bets on being correct. - $P_{\text{lose}} = 1 - P_{\text{win}}$.

Expected Outcome

- **Truthful Betting as a Dominant Strategy**: When participants act rationally, betting truthfully based on their information maximizes expected utility.
- **Aggregate Information Revelation**: The market aggregates participants' private information, leading to accurate outcome.

5.2.6 Early Participation Bonuses

Participants who bet within the first 10% of the market duration receive an extra bonus:

$$R_i^{\text{Early}} = \delta_{i,\text{early}} \times 0.10 \times R_i^{\text{Activity}}$$

$\delta_{i,early} = 1$ if participant i bets early, else 0.

Participants decide whether to bet early to receive the bonus, considering:

- The potential for less information early on (higher risk).
- The additional rewards from early participation.

Early Participation Bonus and Information Asymmetry Participants must decide whether to bet early to receive the bonus, considering the expected value of betting early vs. later.

Expected Utility of Early Betting

$$U_{\text{early}} = \mathbb{E}[\text{Payout}_{\text{early}}] + R_{\text{early}} - B_{\text{early}}$$

Expected Utility of Late Betting

$$U_{\text{late}} = \mathbb{E}[\text{Payout}_{\text{late}}] + R_{\text{late}} - B_{\text{late}}$$

Information Value

Let V_1 represent the value of additional information obtained by waiting.

Participants will bet early if:

$$U_{\text{early}} + R_{\text{bonus}} \geq U_{\text{late}} + V_I$$

Where R_{bonus} is the early participation bonus.

Equilibrium Analysis

Participants will compare the marginal benefit of the early bonus with the marginal value of additional information. The equilibrium strategy depends on:

- **Magnitude of** *Rbonus*: Higher bonuses incentivize early betting.
- **Expected** V_1: If the value of additional information is high, participants may wait.

5.2.7 Collusion and Mechanism Design

Participants may form coalitions to manipulate outcomes.

Coalition Formation

Let C be a coalition of participants.

$$U_C = \sum_{i \in C} U_i$$

The coalition aims to maximize U_C by coordinating strategies.

Incentive Compatibility Constraints

Formally, we require that for each participant i:

$$U_i(\text{truthful action}) \geq U_i(\text{any other action})$$

Individual Rationality

Participants must have a non-negative utility for participating:

$$U_i \geq 0$$

Budget Balance

The mechanism should be financially sustainable:

$$\sum_i R_i \leq \text{Total Incentive Pool}$$

5.3 Quadratic Weighting and Strategic Behavior

The **Adjusted Bet Amount** replaces the actual bet amount when calculating a participant's share of rewards. Rewards are then distributed based on the proportion of each participant's adjusted bet to the total adjusted bets.

Quadratic Weighting applies a mathematical transformation to participants' bet amounts to reduce the disproportionate influence of larger bets

$$B_{i,\text{adjusted}} = B_i^{\alpha}$$

Sum of all participants' adjusted bets in a market:

$$Totaladjustedbetsforepoch = \sum_j B_{j,\text{adjusted}}$$

Where n is the number of participants and Bi is the actual bet of participant i.

Adjusting the Weighting Function

- The square root function uses an exponent of 0.5.
- We can adjust the exponent α to fine-tune the weighting.
- Balance between fairness and incentive for larger bettors.
  - $\alpha = 0.5$ (Moderate adjustment)
  - $\alpha = 0.3$ (Strong Adjustment)
  - α =0.7 (light adjustment)

The primary goals of implementing a Quadratic Weighting mechanism are to:

- **Prevent Whales from Dominating Rewards:** Without weighting, participants who place large bets (whales) could disproportionately earn more rewards, potentially discouraging smaller participants.
- **Encourage Wider Participation:** By adjusting the influence of bet sizes, smaller bettors receive more meaningful rewards, promoting inclusivity and fairness.
- **Diminishing Returns for Larger Bets:** As bet amounts increase, the incremental gain in adjusted bet amount decreases.
- **Enhanced Influence for Smaller Bets:** Smaller bets have a relatively higher adjusted value compared to their actual size.

Properties

- **Concavity**: For $0 < \alpha < 1$, the function is concave, leading to diminishing marginal returns.
- **First Derivative**:

$$\frac{dB_{i,\text{adjusted}}}{dB_i} = \alpha B_i^{\alpha-1}$$

- **Second Derivative**:

$$\frac{d^2 B_{i,\text{adjusted}}}{dB_i^2} = \alpha(\alpha-1)B_i^{\alpha-2} < 0$$

Since $\alpha 1 < 0$, the second derivative is negative, confirming concavity.

Strategic Behavior

Bet Splitting

Problem: Participants may split their bets across multiple accounts to circumvent the diminishing returns.

Suppose a participant with total bet B splits it into n equal parts:

$$\begin{aligned} B_{\text{split}} &= \sum_{i=1}^{n} \left(\frac{B}{n}\right)^{\alpha} \\ &= n\left(\frac{B}{n}\right)^{\alpha} \\ &= n\left(\frac{B^{\alpha}}{n^{\alpha}}\right) \\ &= B^{\alpha} n^{1-\alpha} \end{aligned}$$

Since $\alpha = 0.5, 1-\alpha = 0.5$), so:

$$B_{\text{split}} = B^{\alpha} n^{0.5}$$

- Original adjusted bet without splitting: B^{α}
- Adjusted bet with splitting: $B^{\alpha} n^{0.5}$.

5.4 Sustainability and Emission Modeling

5.4.1 Emission Rate Modeling

The emission rate ER is a function of network activity metrics:

$$ER = BER + k \times (w_1 \ln(NAM) + w_2 \ln(NAP) + w_3 \ln(TBV) + w_4 \ln(IR))$$

The emission rate increases with network activity but is capped:

$$ER = \min\left(BER + k \times \sum_{i=1}^{4} w_i \ln(M_i),\ ER_{\max}\right)$$

Where M_i represents each metric:

- $M_1 = NAM(NumberofActiveMarketsSettled)$.
- $M_2 = NAP(NumberofActiveParticipants)$
- $M_3 = TBV(TotalBettingVolume)$
- $M_4 = IR(IssuesResolved)$.

Participants might attempt to maximize ER by increasing Mi.

Artificially Inflating Metrics: Creating fake markets (increasing NAM), generating fake accounts (increasing NAP), and circulating funds (increasing TB).

Let C be the cost of artificially increasing metrics, and G be the gain from increased emissions. Participants will engage in metric inflation if:

$$G - C > 0$$

Participants choose Mi to maximize G - C , subject to their resource constraints.

5.4.2 Emissions impact on circulating supply

Let S_t be the token supply at time t.

Differential Equation

$$\frac{dS_t}{dt} = S_t \times ER_t$$

Where ER_t is the emission rate at time t.

Assuming ER_t is constant over small intervals:

$$S_t = S_0 \times e^{ER \times t}$$

This exponential growth highlights the importance of controlling ER to prevent excessive inflation.

Inflation Control

To ensure sustainability:

- **Emission Caps**: We set a maximum emission rate per epoch on order to keep emissions in control

5.5 Fee Redistribution and Token Holder Utility

Token Holder Rewards

Token holders receive a portion of the fees:

$$R_{\text{holders}} = 0.30 \times F_{\text{total}}$$

Where F_{total} is the total fees collected.

Utility Function for Token Holders

$$U_{\text{holders}} = R_{\text{holders}} + \Delta P_{\text{token}} \times H - C_{\text{holding}}$$

Where:

- ΔP_{token}: Change in token price.
- H: Number of tokens held.
- $C_{holding}$: Costs associated with holding tokens.

Token holders are incentivized to:

- Promote platform activity to increase F_{total}.
- Enhance token value P_{token}.

5.6 Token Value Accrual Mechanism

It will take approximately 17.8 years to reach the maximum supply of 100,000,000 ARC tokens with a 7% annual emission rate starting from an initial supply of 30,000,000 tokens

5.6.1 Token Value Accrual

One of the primary value accrual mechanisms for ARC tokens is the redistribution of transaction fees to token holders. ARC tokens grant holders rights to a share of the platform's revenue, making the token analogous to a dividend-paying asset.

- **Transaction Fees**: Each bet placed or share sold incurs a transaction fee, collected in USDC.
- **Fee Allocation**: 30% of these fees are used to buy ARC tokens from the market, which are then distributed to token holders.
- **Supply and Demand Dynamics**: The token's value is influenced by the expected future cash flows (fee distributions) to token holders and market perceptions.
- **Cash Flow**: Holding ARC tokens generates passive income in the form of fee distributions.
- **Incentive to Hold**: The prospect of receiving ongoing distributions incentivizes users to hold ARC tokens.

This mechanism creates continuous buying pressure for ARC tokens proportional to the platform's transaction volume, which can positively impact the token's price.

Let:

- F_t = Total transaction fees collected in USDC at time t.
- $F_{ARC,t} = 0.3F_t$ = Portion of fees distributed to ARC token holders at time t
- $P_{ARC,t}$ = Price of ARC token at time t.
- $Q_{dist,t} = \frac{F_{ARC,t}}{P_S PLS,t}$ = Quantity of ARC tokens distributed.
- r: Discount rate (reflecting the required rate of return or cost of capital).

The total quantity of ARC tokens distributed over a period T is:

$$Q_{dist} = \sum_{t=1}^{T} Q_{dist,t} = \sum_{t=1}^{T} \frac{F_{ARC,t}}{P_{ARC,t}}$$

This continuous buying of ARC tokens supports the token's price by increasing demand proportional to platform activity.

5.6.2 Discounted Cash Flow (DCF) Model for ARC Token

We use a **Discounted Cash Flow (DCF)** model to calculate the present value of expected future fee distributions per token. The intrinsic value of an ARC token is the present value of expected future fee distributions.

Fee distribution per ARC token at time t is:

$$D_t = \frac{F_{dist,t}}{S_t}$$

The intrinsic value of an ARC token is the present value of expected future fee distributions.

$$P_{ARC} = \sum_{t=1}^{T} \frac{D_t}{(1+r)^t}$$

For practical modeling, we can assume that fee distributions grow at a constant rate (g) after a certain period. This allows us to use the **Gordon Growth Mode**l for a perpetuity:

$$P_{ARC} = \frac{D_1}{r-g}$$

Where:

- D1: Fee distribution per token in the next period.
- g: Expected perpetual growth rate of fee distributions.
- Growth Rate (g): Reflects the expected growth in fee distributions due to platform growth.
- Discount Rate (r): Should account for the risk associated with the platform and the token.