The Token Economics of Sifchain

Dear Reader,

At long last, you are permitted to read about the token economics of the Sifchain protocol, and what makes Rowan so valuable. Your mind takes on a state of gratitude and focus and you begin to understand.

Sifchain is a DEX with automated market makers (AMMs) that hold liquidity providers' capital and trade it against requests from swappers. Sifchain aims to support the exchange of all cryptocurrencies. It employs a hybrid order matching model that allows cost effective order execution through continuous liquidity pools (CLPs) running the AMM model listed above.

Sifchain's economic system comprises two subsystems - the Validator Subsystem (V), and the Liquidity Provider Subsystem (LP). Agents in both systems are able to earn rewards by means of staking in the Validator Subsystem and by providing liquidity in the Liquidity Provider Subsystem.

The token economic model for Sifchain is simple. Sifchain consists of two-sided liquidity pools in which one of the tokens is Rowan and the other is an external token such as BTC, ETH, or XLM (or at least a pegged version thereof running on Sifchain). The value of either side of each liquidity pool is equal to the other. For example, if the RWN:YFI pool has $100,000 USD worth of YFI, it must also have $100,000 USD worth of RWN.

RWN holders who delegate to a pool earn swap fees for every swap in accordance with the fee structure below. RWN holders can also stake capital in a validator or delegate capital to validators to receive block rewards.

If the value of external assets on Sifchain is a modest $10M in total, the other side of the liquidity pools must have $10M total of RWN so that the pools stay evenly weighted. This means Sifchain has a combined $20M in liquidity.

The validators then must be securing at least $20M in RWN so that the network is secure and the cost of malicious activity (slashing) is not worth anything a validator can gain from fraudulence as per Tendermint consensus. This means that for every $10M in external capital, we can expect $30M in Rowan's total liquid market cap.

Here is a breakdown of how $10M in external capital would affect Rowan's other metrics based on the above:

- External Capital in Liquidity Pool - $10M
- Value of all Rowan on Sifchain (liquidity pools + validators) - $30M
- Liquid Market Cap - $30M
- Fully Diluted Market Cap (Current token distribution has 14.5% of all Rowan liquid and available during Sifchain's launch) - $206M

This, of course, does not include tokens stored outside of Sifchain such as in an investor wallet, on another exchange, or in cold storage. Further analysis on our core model can be found here from Gauntlet Network's analysis of Thorchain and we've posted modifications thereon.

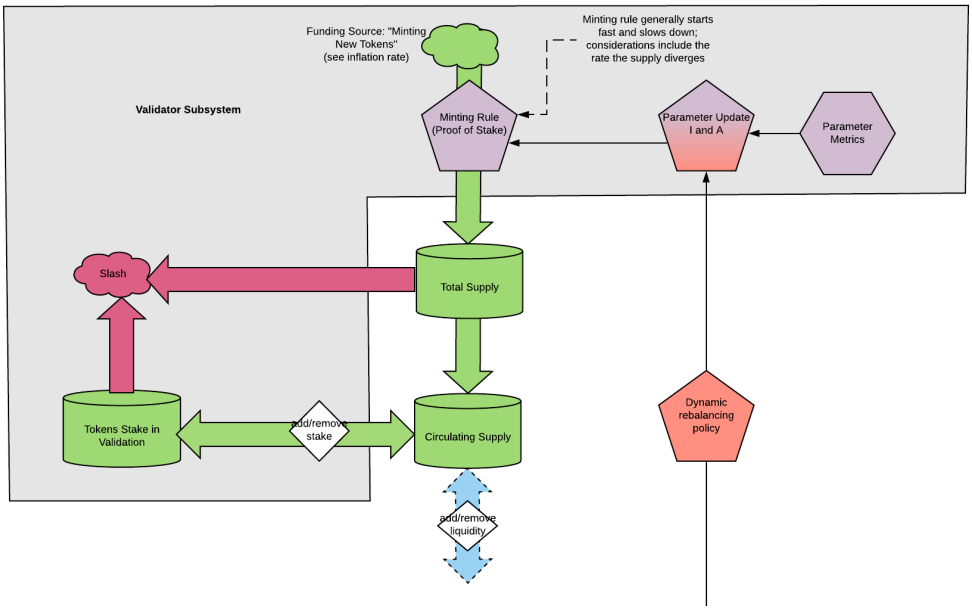
1 Sifchain Validator Subsystem

This Validator subsystem serving the Sifchain economy is separated from the system economy where possible, with interactions with system economy clearly delineated. Importantly, the only interaction with the Liquidity Provider subsystem occurs through the quantity of stock in the

Circulating Supply state. The system economy layer has control over the Validator subsystem through the Minting Policy and its parameters.

1.1 Stock and Flow Diagram

Static version as of October 9:



1.2 Token States

Partitioning of Token Stocks as state variables in the Sifchain economy. For any given valid state, there is an 'expected' payout that can be calculated as portions of held positions, that are compositions of system level variables. First, each position type is considered:

Name	Description	Initial Quantity	Dynamic Characteristic	Symbol
Total Rowan	All Rowan	Summation of Initial States	Increasing subject to other token states	$\mathbf{S}$
Circulating Supply	Slack Variable	Difference from all other token states	Subject to other token states	$\mathbf{S_C}$
Validator Stake	Locked in stake by Validators	v Validators at launch x s stake	Subject to actions of Validators and Staking Policy	$\mathbf{S_V}$

1.3 Stateful Parameters

Name	Description	Initial Quantity	Dynamic Characteristic	Symbol
Inflation	Inflation	Summation of Initial States	Increasing subjectto other token states	I
Annual Provisions	Expected block rewards over a year, however only used at each block	Subject to Inflation and Block Time	Subject to updates on Inflation and Blocks per Year	A

1.4 Parameter Metrics

Name	Description	Initial Quantity	Dynamic Characteristic	Symbol
Blocks Per Year	Expected quantity	$31,536,000$/ Estimated block time in seconds	Able to be updated on upon update	β
Inflation Max	Maximum Inflation Rate	Upped Bound	Not expected to change	I_{max}
Inflation Min	Minimum Inflation Rate	Lower Bound	Not expected to change	I_{min}
Inflation Rate Change	Bound on movement of Inflation Rate	$I_{max} - I_{min}$	Not expected to change	ΔI_{max}
Goal Bonded	Desired Percent of bonded tokens to circulating supply	Assume to be 66%	Ratio of Validator Economy Tokens	γ_v
Bonded Ratio	Actual Percent of bonded tokens	To be Determined	Subject to actions of entire economy, expected to change every block	ρ_v

1.5 Mechanism Difference Equations

1.5.1 Minting

Assume cosmos SDK minting, with the State holding two variables:

```
type Minter  struct
        Inflation     sdk.Dec//current annual inflation rate
   AnnualProvisions   sdk.Dec//current annual exptected provisions
```

}

Computation of the State is aided with the following parameters:

```
type Params  struct
          MintDenom     string//type of coin to mint
InflationRateChange     sdk.Dec//maximum annual change in inflation rate
       InflationMax     sdk.Dec//maximum inflation rate
       InflationMin     sdk.Dec//minimum inflation rate
         GoalBonded     sdk.Dec//goal of percent bonded atoms
      BlocksPerYear     uint64//expected blocks per year
```

}

The State Update to **Inflation** is recalculated each hour in current implementation (ability to increase frequency to block frequency):

```
NextInflationRate(params Params, bondedRatio sdk.Dec) (inflation sdk.Dec) {
    inflationRateChangePerYear = (1 - bondedRatio/params.GoalBonded) * params.InflationRateChange
    inflationRateChange = inflationRateChangePerYear/blocksPerYr

    // increase the new annual inflation for this next cycle
    inflation += inflationRateChange
    if inflation > params.InflationMax {
        inflation = params.InflationMax
    }
    if inflation < params.InflationMin {
        inflation = params.InflationMin
    }

    return inflation
}
```

Updates to the **Inflation**, I, occur via an update to the change in inflation, ΔI^+. The change in inflation is computed with the following update equation:

$$\Delta I^+ = \frac{(1 - \frac{\rho_v}{\gamma_v}) \cdot \Delta I_{max}}{\beta}$$

This update to the inflation rate occurs as an error term on the deviation of the percent of supply bonded in stake from that of the desired goal percentage for the system. This adjustment is limited by the maximum allowable change in annual inflation rate, ΔI_{max}.

The new inflation rate, I^+, is then updated accordingly:

$$I^+ = I + \Delta I^+$$

where I^+ is subject to upper and lower limits, I_{max} and I_{min}, respectively. Thus, I^+ will remain in the range:

$$I_{min} \leq I^+ \leq I_{max}$$

The set of ratios, $\overrightarrow{\rho}$, is defined as its portion of the supply with the respect to the total supply

$$\overrightarrow{\rho} = \{\rho_v, \rho_l, \rho_t, \rho_c ...\}$$

Assume Thus, `bondedRatio`, ρ_v, is computed as:

$$\rho_v = \frac{\mathbf{S_V}}{\mathbf{S}}$$

The input to the control system is the observed current state of the system, represented by vector $\overrightarrow{\rho}$.

- The observed supply in Validator Subsystem ρ_v
- The observed supply in Liquidity Provider Subsystem ρ_l
- The observed supply in Future Subsystem ρ_t
- The observed supply in Circulating Economy ρ_c

Via comment ALT definition:

$$\rho_v = \frac{\mathbf{S_V}}{\mathbf{S_C}}$$

If the ratio supply staked in validation to the circulating supply desired, then it can be obtained according to its ρ components:

$$\frac{\mathbf{S_v}}{\mathbf{S_c}} = \frac{\rho_v}{\rho_c}$$

IMPORTANT: This is the inclusion of the Sifchain system economy into the validation susbsytem.

Assume `BlocksPerYear`, β, is static and computed as:

$$\beta = \frac{31,536,000}{\mathbf{T_{avg-est}}}$$

IMPORTANT: This is the inclusion of time into the validation susbsytem.

Then the **AnnualProvisions** State, A, is updated with the new **Inflation**, I, according to:

```
NextAnnualProvisions(params Params, totalSupply sdk.Dec) (provisions sdk.Dec) {
    return Inflation * totalSupply
```

$$\mathsf{A} = I \cdot \mathbf{S}$$

If using ALT definition: minting from S (total) though $\mathbf{S_V}$ and $\mathbf{S_C}$ are used in the target.

Then, the block reward is computed for that block:

```
BlockProvision(params Params) sdk.Coin {
    provisionAmt = AnnualProvisions/ params.BlocksPerYear
    return sdk.NewCoin(params.MintDenom, provisionAmt.Truncate())
```

$$\Delta\mathbf{B} = \frac{\mathsf{A}}{\beta}$$

We can represent flows (change) as:

$$\mathbf{b} = \Delta\mathbf{B}$$

The total Rowan is updated as:

$$\mathbf{S}^+ = \mathbf{S} + \mathbf{b}$$

And these Rowan are deposited in the account of Validator's wallet, effectively being introduced into circulating supply:

$$\mathbf{S_C^+} = \mathbf{S_C} + \mathbf{b}$$

1.5.2 Minting Parameterization

The goal of this section is to characterize the minting mechanism that allows for a control policy to be implemented upon on it. Importantly, this characterization must be specified in a way to co-exist with the liquidity pool subsystem.

Name	Domain	Symbol	Notes
Blocks Per Year	$D = \mathbb{Z}^+$	β	31,536,000 / Estimated block time in seconds
Inflation Max	$D = (0, 1]$	I_{max}	Upped Bound on Inflation Rate
Inflation Min	$D = [0, 1)$	I_{min}	Lower Bound on Inflation Rate
Inflation Rate Change	$D = (0, 1)$	ΔI_{max}	Bound on movement of Inflation Rate Assume: $\Delta I_{max} = I_{max} - I_{min}$
Goal Bonded	$D = (0, 1)$	γ_v	Desired Ratio of Validator Economy Tokens
Bonded Ratio	$D = (0, 1)$	ρ_v	Actual Percent of bonded tokens
Validator Control Parameter	$D = [0, 1]$	λ_v	Validator Component of $\overrightarrow{\lambda}$

Control parameter, $\overrightarrow{\lambda}$ is defined as the set of components:

$$\overrightarrow{\lambda} = \{\lambda_v, \lambda_l, ..., \lambda_c, \lambda_t\}$$

where λ_v will provide an update (exert influence) to the goal bonded ratio, γ_v^+. γ_v^+ is one of a set defined of ratios of total supply, where:

$$\overrightarrow{\gamma} = \{\gamma_v, \gamma_l, ..., \gamma_c, \gamma_t\}$$

It is expected that only the validator and liquidity pool component of these parameters would be employed. However, this framework allows for the ability to control other subsets of supply tokens if they are not purely dependent upon $\mathbf{S_V}$ and $\mathbf{S_L}$.

Substituting the partitioned framework set of equations in the Minting Mechanism for the determination of block rewards:

$$\mathbf{b} = \frac{(I + \frac{(1 - \frac{\rho_v}{\gamma_v}) \cdot \Delta I_{max}}{\beta}) \cdot \mathbf{S}}{\beta}$$

Substituting ρ_v with $\frac{\mathbf{S_V}}{\mathbf{S}}$:

$$\mathbf{b} = \frac{(I + \frac{(1 - \frac{\frac{\mathbf{S_V}}{\mathbf{S}}}{\gamma_v}) \cdot \Delta I_{max}}{\beta}) \cdot \mathbf{S}}{\beta}$$

$$\mathbf{b} = \frac{(I + \frac{(1 - \frac{\frac{\mathbf{S_V}}{\mathbf{S_C}}}{\gamma_v}) \cdot \Delta I_{max}}{\beta}) \cdot \mathbf{S}}{\beta}$$

Distributing and separating terms:

$$\mathbf{b} = \frac{I \cdot \mathbf{S}}{\beta} + \frac{\Delta I_{max} \cdot \gamma_v \cdot \mathbf{S}}{\beta^2} - \frac{\Delta I_{max} \cdot \mathbf{S_v}}{\beta^2}$$

$$\mathbf{b} = \frac{I \cdot \mathbf{S}}{\beta} + \frac{\Delta I_{max} \cdot \gamma_v \cdot \mathbf{S}}{\beta^2} - \frac{\Delta I_{max} \cdot \mathbf{S} \cdot \frac{\mathbf{S_v}}{\mathbf{S_c}}}{\beta^2}$$

The first term represents the current inflation mechanism. The summation of the second and third terms provides the update to the minting mechanism. When the amount in staking is less than the goal, the second term outweighs the third term, the summation i positive and block rewards are increased. When the amount in staking is more than the goal, the third term outweighs the second term, the summation is negative and block rewards are decreased.

Introducing the system control parameter λ into the minting mechanism:

$$\mathbf{b}(\lambda_v) = \frac{I \cdot \mathbf{S}}{\beta} + \lambda_v \cdot \left[\frac{\Delta I_{max} \cdot \gamma_v \cdot \mathbf{S}}{\beta^2} - \frac{\Delta I_{max} \cdot \mathbf{S_v}}{\beta^2} \right]$$

$$\mathbf{b}(\lambda_v) = \frac{I \cdot \mathbf{S}}{\beta} + \lambda_v \cdot \left[\frac{\Delta I_{max} \cdot \gamma_v \cdot \mathbf{S}}{\beta^2} - \frac{\Delta I_{max} \cdot \mathbf{S} \cdot \frac{\mathbf{S_v}}{\mathbf{S_c}}}{\beta^2} \right]$$

where a λ_v value of 0, will exert no new influence on the mechanism and minting would continue at the same rate as in the previous block. A λ_v value of 1 would exert the full influence of the update to the minting rate, while still metered with respect to the difference between goal and desired ratios of validator tokens to supply, as well as the limits on I, specifically: ΔI_{max}, I_{max}, and I_{min}.

1.5.3 Staking

A delegated proof of stake protocol exists, where the top v validators are selected at each block.

A validator at any given time is considered to be in one of three states:

1. Bonded
2. Unbonding
3. Unbonded

Bonded validators are the elements v validators in the set that are chosen for that particular block. Only these validators are eligible for rewards.

The unbonding state provides validators with a transition period for validators not chosen for validation, whether due to not enough power or a slashing offense. Unbonded tokens exert a pressure on the bonded validators, quantifying the momentum of the validator subsystem. Tokens that are in the unbonded pool of tokens are not eligible for rewards. Therefore, the unbonded pool of tokens are not participating in the validator economy.

$$\mathbf{s^i_{unbonded}} \subset S_C$$

where

$$i \notin v$$

Newly staked funds arrive from circulating supply, regardless of whether the funds were previously in a liquidity pool:

$$\mathbf{S_V^+} = \mathbf{S_C} - \mathbf{s_v}$$

1.5.4 Slashing

Validators committing a slashable offense have their stake removed by a slash factor, σ. The rate of slash depends on the offense and state of the validator. Also the offense may result in the validator being jailed.

Slashing Percentages based on a small set of offenses and validator states with corresponding rates:

$$\mathbf{s^i_{slash}} = \sigma_i \cdot \mathbf{S^i_V}$$

For a given block across all slashed vaildators:

$$\mathbf{s_{slash}} = \sum^{i} \sigma_i \cdot \mathbf{S_V^i}$$

Slashed funds are removed from stake:

$$\mathbf{S_V^+} = \mathbf{S_V} - \mathbf{s_{slash}}$$

Slashed funds are burned:

$$\mathbf{S^+} = \mathbf{S} - \mathbf{s_{slash}}$$

1.5.5 Delegators

Delegators are considered to be a secondary set of actors in the validator subsystem economy. Delegators earn shares of claims to rewards with their delgated stake tokens. Delegators are incurring slashing risk if they choose an untrustworthy validator.

Delegated tokens are treated as bonded tokens and eligible for block rewards in the same exact manner that as if it were the validator putting up its own stake, from the point of view of the minting mechanism.

Validators may charge a commission fee to delegators. This again is a secondary flow for participation in the validator economy. Where essentially a slightly greater than proportional block rewards are flowing to the validator with respect to the delegator.

Delegators will choose to participate in this action where their portion of block rewards less commission fees is greater than a competing choice of action (provide liquidity) with their tokens.

Validators will continue to participate in this economy when their portion of rewards and commission fees less their operational costs is more profitable than a competing choice.

2 Sifchain Liquidity Pool Subsystem

Sifchain uses a 2-sided liquidity pool with Rowan and an external token. At launch, Sifchain will use a slip-based fee but with the following changes to its roadmap.

2.1 Notations

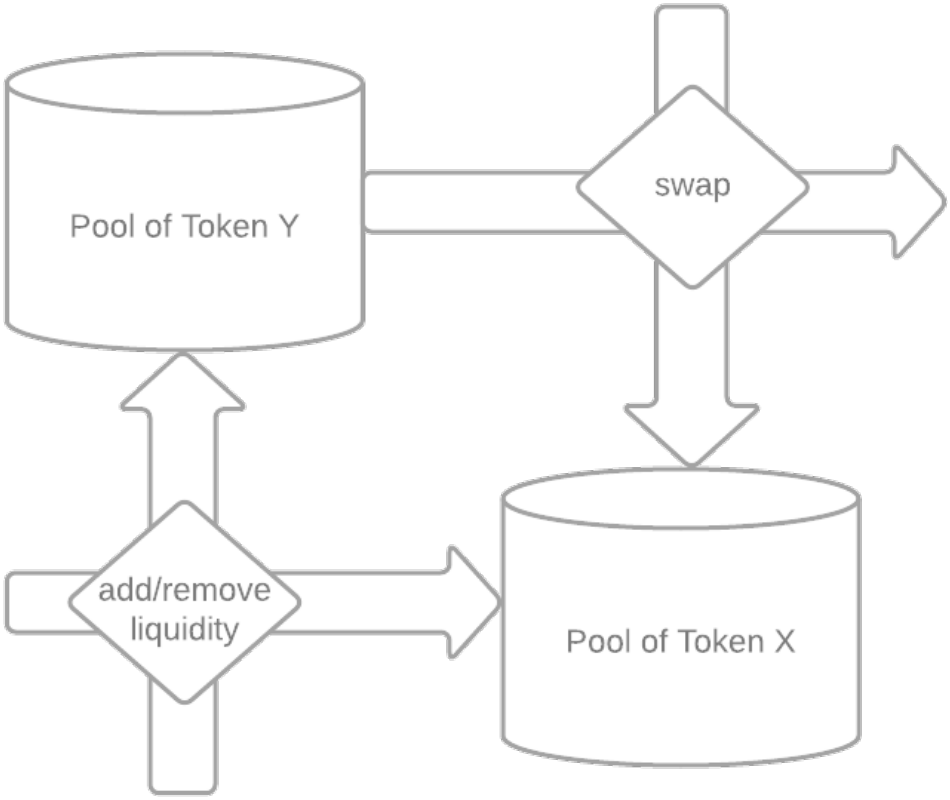
Name	Description	Dynamic Characteristic	Symbol
Time	Timesteps		t
Period	Time period		T
Liquidity Pools	Number of liquidity pools		n_{LP}
Liquidity Pool Identifier	Reference to a particular liquidity pool		i, j, k
Rowan	Number of Rowan tokens in liquidity pool		$\mathbf{S_L}$
M-Tokens	Number of M-tokens in liquidity pool		$\mathbf{M_L}$
Swap Fee	Fees for swapping in a liquidity pool		f_{swap}
Swap Volume	Volume of swaps in a liquidity pool		v_{swap}
Rowan Price	Price of Rowan in a liquidity pool		P_S
M-Token Price	Price of M-token in a liquidity pool		P_M
Liquidity Provider	Reference to a liquidity provider		π
Liquidity Provider	Reference to the liquidity provider subsystem		Π
Liquidity Provider Reward	Reward earned by a liquidity provider		$\mathbf{R}_\pi$
Liquidity Provider System Reward	Reward earned by the liquidity provider system		$\mathbf{R_\Pi}$
Liquidity Provider's Cost of Operation	Operational cost incurred by a liquidity provider		C_Π
Liquidity Provider System ROI	Return on investment of the liquidity provider system		Ω_Π
Awarded Subsidies	Proto rewards- tokens awarded to LPs and Validators		$\mathbf{S_P}$
Observed Ratio	Ratio of supply tokens in Liquidity Pool to the total supply		ρ_l
Target Ratio	Target ratio of supply tokens in Liquidity Pool to the total supply		γ_l
Swap fee parameter	Control parameter on the liquidity subsystem's swap fee		λ_l

2.2 Roles

A liquidity pool consists of two primary actors:

- **Liquidity Providers** who provide Rowan or N-tokens for a given duration of time to earn rewards.
- **Swappers** who participate in the liquidity pool to swap Rowan for N-tokens or vice versa.

2.3 Actions



2.3.1 Add and remove liquidity

The Rowan price P_S is defined as the units of Rowan **S** that can be bought per unit of M-token **M**.

$$P_S = \frac{\mathbf{S_L}}{\mathbf{M_L}}$$

Adding liquidity can be modeled as adding (m, s) such that

$$P_S^+ = P_S$$

$$P_S^+ = \frac{\mathbf{S_L} + s}{\mathbf{M_L} + m} = \frac{\mathbf{S_L}}{\mathbf{M_L}} = P_S$$

where

- P_S is the Rowan price at the current state,
- P_S^+ is the Rowan price at the next state,
- $s > 0$ is the amount of Rowan added through the action;
- $\mathbf{s} = \mathbf{\Delta S}$ - $m > 0$ is the amount of M-tokens required to make the action; $\mathbf{m} = \mathbf{\Delta M}$

Removing liquidity can be modeled as removing (m, s) such that

$$P_S^+ = \frac{\mathbf{S_L} + s}{\mathbf{M_L} + m} = \frac{\mathbf{S_L}}{\mathbf{M_L}} = P_S$$

where

- P_S is the Rowan price at the current state, - P_S^+ is the Rowan price at the next state, - $s < 0$ is some fraction of S; the amount of Rowan removed through the action; $\mathbf{s} = \mathbf{\Delta S}$
- $m < 0$ is some fraction of M; the amount of M-tokens removed through the action $\mathbf{m} = \mathbf{\Delta M}$

2.3.2 Swapping

A swap is defined as an exchange of M-tokens for Rowan or vice versa, while preserving the Continuous Liquidity Provider (CLP) formula.

Parameterized Swap Fees Consider a prior state (M, S). A swapper deposits $\mathbf{m} = \Delta\mathbf{M}$ to obtain $s = f(m; M, S)$ computed using the CLP model.

We need to design a parametrized swap fee policy in order to modulate the fee according to the dynamic rebalancing policy. We choose a parameter λ_l such that

- at $\lambda_l = 0$, the swapper incurs no fee to swap akin to the Constant Marker Maker Model (idealized Uniswap)
- at $0 < \lambda_l \leq 1$, the swapper incurs a combination slip-based fee to swap
- at $\lambda_l > 1$, the swapper incurs a magnifed slip-based fee

In the Constant Market Maker Model, the next state after a swap is

$$\mathbf{S_{L,cmm}^+} = \mathbf{S_L} - \frac{\mathbf{mS_L}}{\mathbf{m + M_L}}$$

In the Continuous Liquidity Pool Model, the next state after a swap is

$$\mathbf{S_{L,clp}^+} = \mathbf{S_L} - \frac{\mathbf{mS_L M_L}}{(\mathbf{m + M_L})^2}$$

In the case of the Constant Product Market Maker, the amount of tokens returned to the swapper who provided $\mathbf{m}$ is $\frac{\mathbf{mS_L}}{(\mathbf{m+M_L})}$, whereas in the case of the slip based fee it is $\frac{\mathbf{mS_L M_L}}{(\mathbf{m+M_L})^2}$.

Given that we want a parametrization that allows for both of these rules to be realized, we can express the Parametrized Continuous Liquidity Provider Model as follows:

$$\mathbf{S_{L,param}^+} = \mathbf{S_L} - \frac{\mathbf{mS_L}}{\mathbf{m + M_L}} + \lambda_l \left(\frac{\mathbf{mS_L}}{\mathbf{m + M_L}} - \frac{\mathbf{mS_L M_L}}{(\mathbf{m + M_L})^2} \right)$$

Note that

- if $\lambda_l = 0$ then we recover the Constant Product Market Maker $\mathbf{S_{L,cmm}}$ equation
- if $\lambda_l = 1$ the we recover the Continuous Liquidity Provider $\mathbf{S_{L,clp}}$ equation.

Furthermore, this is analytically equivalent to:

$$\mathbf{S_{L,param}^+} = \mathbf{S_L} - (1 - \lambda_l)\frac{\mathbf{mS_L}}{\mathbf{m + M_L}} - \lambda_l \frac{\mathbf{mS_L M_L}}{(\mathbf{m + M_L})^2}$$

That means the the quantity of tokens recieved by the swapper is:

$$\mathbf{S_L} - \mathbf{S_{L,param}^+} = (1 - \lambda_l)\frac{\mathbf{mS_L}}{\mathbf{m + M_L}} + \lambda_l \frac{\mathbf{mS_L M_L}}{(\mathbf{m + M_L})^2}$$

Furthermore, we can interpret the Constant Product Market Maker rule as being "without fees" (the slippage is not a fee). Thus, any fee rule such as the Continuous Liquidity Provider

rule can be interpreted relative to what the swapper would have received under the Constant Product Market Maker, giving the difference which is the "fee".

In this framing, the fee for the Continuous Liquidity Provider rule is

$$\begin{aligned}(\mathbf{S^+_{L,cmm}} - \mathbf{S_L}) - (\mathbf{S^+_{L,clp}} - \mathbf{S_L}) &= \mathbf{S^+_{L,cmm}} - \mathbf{S^+_{L,clp}} \\ &= \frac{\mathbf{mS_L}}{\mathbf{m + M_L}} - \frac{\mathbf{mS_L M_L}}{\mathbf{(m + M_L)^2}} \\ &= \frac{\mathbf{mS_L}}{\mathbf{m + M_L}} \left(1 - \frac{\mathbf{M_L}}{\mathbf{(m + M_L)}}\right) > 0\end{aligned}$$

The λ_l term needs to parameterize the fee directly such that

- when $\lambda_l = 0$ we have no fee - when $\lambda_l = 1$ we have the slip based fee
- when $\lambda_l > 1$ we have maginified (or suppresed) version of the slip-based fee.

$$\begin{aligned}(\mathbf{S^+_{L,cmm}} - \mathbf{S_L}) - (\mathbf{S^+_{L,param}} - \mathbf{S_L}) &= \mathbf{S^+_{L,cmm}} - \mathbf{S^+_{L,param}} \\ &= \lambda_l \left(\frac{\mathbf{mS_L}}{\mathbf{m + M_L}} - \frac{\mathbf{mS_L M_L}}{\mathbf{(m + M_L)^2}}\right) \\ &= \lambda_l \frac{\mathbf{mS_L}}{\mathbf{m + M_L}} \left(1 - \frac{\mathbf{M_L}}{\mathbf{(m + M_L)}}\right) > 0\end{aligned}$$

for any parameter $\lambda_l > 0$ where $m, M_L, S_L > 0$.

This gives the **Parametrized Swap Fee** for this Liquidity Provider system

$$f_{swap} = \lambda_l \frac{\mathbf{mS_L}}{\mathbf{m + M_L}} \left(1 - \frac{\mathbf{M_L}}{\mathbf{(m + M_L)}}\right)$$

Cases **No fees at** $\lambda_l = 0$

$$f_{swap,\lambda_l=0} = 0$$

Combination Slip-Based Fee at $0 < \lambda \leq 1$

$$f_{swap,0<\lambda_l\leq 1} = \lambda_l \frac{\mathbf{mS_L}}{\mathbf{m + M_L}} \left(1 - \frac{\mathbf{M_L}}{\mathbf{(m + M_L)}}\right)$$

Magnified Slip-Based Fee at $\lambda_l > 1$

$$f_{swap,\lambda_l>1} = \lambda_l \frac{\mathbf{mS_L}}{\mathbf{m + M_L}} \left(1 - \frac{\mathbf{M_L}}{\mathbf{(m + M_L)}}\right)$$

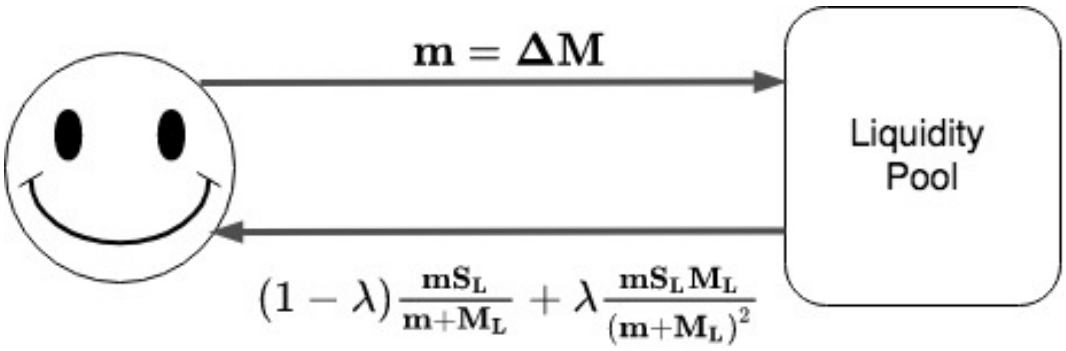
The Rebalancing Policy for Sifchain thus must provide instructions as to whether fees should rise or fall by providing an increase or decrease in λ of the form $\lambda^+ = \lambda + \Delta\lambda$ where $\Delta\lambda$ is determined as a function of the overall system state including token supply, tokens staked in the validator subsystem and locked across the liquidity pools.

Plugging in the slip-based fee $f_{swap} = \lambda \frac{\mathbf{mS_L}}{\mathbf{m+M_L}} \left(1 - \frac{\mathbf{M_L}}{\mathbf{(m+M_L)}}\right)$, the resulting Rowan obtained by the swapper is

$$\mathbf{S_L} - \mathbf{S^+_{L,param}} = (1 - \lambda_l) \frac{\mathbf{mS_L}}{\mathbf{m + M_L}} + \lambda_l \frac{\mathbf{mS_L M_L}}{\mathbf{(m + M_L)^2}}$$

We can define an invariant I in this subsystem to represent the conservation of the $\mathbf{M_L}, \mathbf{S_L}$ relationship.

$$I = f(f_{swap}, \mathbf{S_L}, \mathbf{M_L})$$



2.3.3 Create and destroy pool

A user can list a new pool with their choice of M-token and its ratio with Rowan. To create a pool, the user must provide the initial liquidity in its specified Rowan:M-token ratio.

Under reasonable grounds, a pool can be removed or destroyed from Sifchain.

2.4 State variables

Definition 2.1. ***Rowan in Liquidity Provider Subsystem*** $\mathbf{S_L}$ *The amount of Rowan in locked up as liquidity in all the liquidity pools of the Liquidity Provider Subsystem in the current state.*

Definition 2.2. ***Rowan in Validator Subsystem*** $\mathbf{S_V}$ *The amount of Rowan currently staked by all validators in the Validator Subsystem in the current state.*

Definition 2.3. ***Rowan in Circulating Supply*** $\mathbf{S_C}$ *Circulating supply. This contains all the Rowan circulating in the system, including the stock availed through the accumulation of swap fees.*

Definition 2.4. ***Liquidity Provider Rewards*** $\mathbf{R}_\pi$ *Rewards earned by the liquidity provider are a function of the swap fees and the dynamic rebalacing parameter.*

2.5 Metrics

Definition 2.5. ***M-token Supply*** *The amount of M-tokens in the liquidity pool in the current state.*

Definition 2.6. ***Swap Fees*** *Fees incurred by swapper while making a Rowan/M-token swap. Sifchain implements a* paramterized slip-based fee *for swaps, computed using the Continuous Liquidity Provider (CLP) model. The parameter* λ *The swap fee is a function of the swap volume and the demand for a pool's liquidity.*

$$f_{swap} = \lambda \frac{\mathbf{mS_L}}{\mathbf{m + M_L}} \left(1 - \frac{\mathbf{M_L}}{\mathbf{(m + M_L)}}\right)$$

Definition 2.7. ***Liquidity Provider System Revenue*** *We can define the revenue per period of the liquidity provider subsystem as a function of the swap fee policy and the volume of swaps that occur in that period.*

$$R_{\Pi,T} = \sum f_{swap} * v_{swap,T}$$

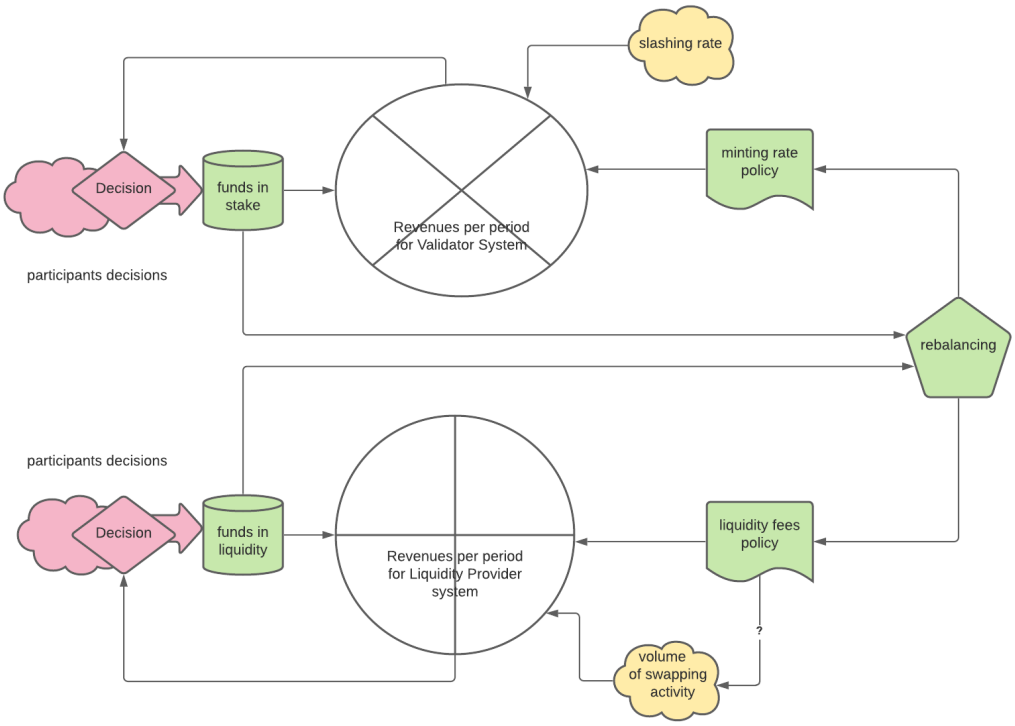
Definition 2.8. ***Liquidity Provider System ROI*** *We can define the ROI per period of the liquidity provider subsystem as a function of the revenue generated in that period and the cost of providing liquidity.*

$$\Omega_{\Pi,T} = \frac{R_{\Pi,T}}{C_{\Pi,T}}$$

3 Balancing the Sifchain Validator and Liquidity Pool Subsystems

In order to maintain system stability and ensure reasonable function of the V subsystem and the LP subsystem, the return on investment incurred by agents in either system must not significantly outweigh the other. In the case where rewards in the LP subsystem are more attractive, validator agents would choose to deploy their capital towards providing liquidity instead of towards staking. The result is a Validator Subsystem with dangerously low staked capital, thus leaving the network cryptoeconomically insecure. On the other hand, if validator rewards are substantially higher than liquidity provider rewards then the liquidity pools will be undercapitalized. The cannibalization of either subsystem function can be prevented by dynamically rebalacing the ROI across each subsystem such that in any given block, the agent's ROI in either subsystem is roughly equivalent.

The two subsystems described above have their rewards harmonized as per a rebalancing policy.



This policy ensures that the revenue earned by participants in the validator system and the liquidity system are balanced and equivalent. This is to prevent validators from jumping over to the liquidity system so as to earn higher rewards, setting off a positive feedback loop that brings in even more validators into the liquidity system, eventually driving the validator system empty and the Sifchain proof-of-stake system insecure. And vice versa on from the liquidity system to the validator system.

4 Rebalancing Policy

The rebalancing policy ensures that the revenue earned by participants in the validator system and the liquidity system are balanced at pre-defined target ratios $\gamma_t, \gamma_v, \gamma_l$ and γ_c. These four

are used to represent specific categories to allow the paper to be digested by a more general audience. but this system can scale to arbitary subsystems such as Ecosystem.

The rebalacing policy informs the control rules within each subsystem so as to

We want to design a rebalancing policy in a 3-dimensional stochastic vector that represents a basin of attraction, wherein the stable equilibrium are defined as the point where:

$$\rho_v = \gamma_v$$

$$\rho_l = \gamma_l$$

$$\rho_t = \gamma_t$$

Due to the the property of ratios we can compute the residual or slack supply that constitutes the circulating economy like so

$$\rho_v + \rho_l + \rho_t + \rho_c = \gamma_v + \gamma_l + \gamma_t + \gamma_c = 1$$

$$\rho_c = 1 - (\rho_v + \rho_l + \rho_t)$$

$$\gamma_c = 1 - (\gamma_v + \gamma_l + \gamma_t)$$

This ensures that the circulating supply is balanced

$$\rho_c = \gamma_c$$

The point where $\rho_v = \gamma_v$, $\rho_l = \gamma_l$, $\rho_t = \gamma_t$ and $\rho_c = \gamma_c$ is the joint stable equilibrium between the 4 dynamical systems: the Validator Subsystem, the Liquidity Provider Subsystem, the Future Subsystem, and the Circulating Economy.

4.1 Policy Inputs

The input to the control system is the observed current state of the system, represented by vector $\overrightarrow{\rho}$.

- The observed supply in Validator Subsystem ρ_v
- The observed supply in Liquidity Provider Subsystem ρ_l
- The observed supply in Future Subsystem ρ_t
- The observed supply in Circulating Economy ρ_c

$$\overrightarrow{\rho} = \{\rho_v, \rho_l, \rho_t ... \rho_c\}$$

where each ratio component is defined as its portion of the supply with the respect to the total supply:

$$\rho_X = \frac{\mathbf{S_X}}{\mathbf{S}}$$

This formulation allows for composition of any two sets of supply. For example, if the ratio supply staked in validation to the circulating supply desired, then it can be obtained according to its ρ components:

$$\frac{\mathbf{S_v}}{\mathbf{S_c}} = \frac{\rho_v}{\rho_c}$$

The control parameter, $\overrightarrow{\lambda}$ is defined as the set of components:

$$\overrightarrow{\lambda} = \{\lambda_v, \lambda_l, \lambda_t ... \lambda_c\}$$

4.2 Error

The error term, $\overrightarrow{\epsilon}$, in the controller is defined as the difference between the goal ratio and the desired ratio, where ratios are serving as indicators of economic activity:

$$\overrightarrow{e} = \overrightarrow{\gamma} - \overrightarrow{\rho}$$

The error function is defined as:

$$\Psi(\overrightarrow{\rho}) = \|\overrightarrow{\gamma} - \overrightarrow{\rho}\|^2$$

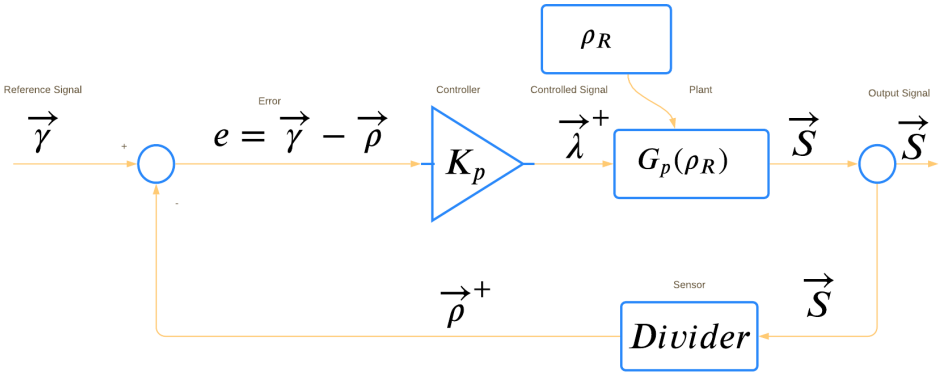
4.3 Objective Function

The error function provides an update to $\overrightarrow{\lambda}$ such that the update must result in decreasing the error function. Thus, the difference should be decreasing:

$$\nabla\Psi(\overrightarrow{\rho}) < 0$$

There is an expectation that this condition can be realized but not enforced through the behavior of actors on the system:

$$\mathbb{E}[\nabla\Psi(\overrightarrow{\rho})|\mathcal{B}]$$



4.4 Control Function

Proportional control provides the update to $\vec{\lambda}$ through directional gain. If $\vec{\lambda}$ is vectorized as:

$$\begin{bmatrix} \lambda_v \\ \lambda_l \\ \lambda_t \\ . \\ . \\ . \\ \lambda_c \end{bmatrix}$$

$$\vec{\lambda}^+ = K_p \vec{\lambda}$$

where:

$$K_p = \begin{bmatrix} K_v \\ 0 \\ 0 \\ 0 \\ 0 \\ K_l \\ 0 \\ 0 \\ 0 \\ 0 \\ K_t \\ 0 \\ 0 \\ 0 \\ 0 \\ K_c \end{bmatrix}$$

In the case of only considering the validator and liquidity subsystem in this control policy, K_P reduces to:

$$[K_l \ 0 \ 0 \ K_v]$$

In order to acheive directional correction over the error e, the sign of the error partitions the policy into a piecewise function.

4.4.1 Control Parameter Update

$$\overrightarrow{\lambda}^{+} = \begin{cases} 1 - (1 - |e|) \cdot (1 - \lambda) & \text{for} \quad e > 0 \\ (1 - |e|) \cdot \lambda & \text{for} \quad e < 0 \\ \lambda & \text{for} \quad e = 0 \end{cases}$$

- Note for $e = 0$, either of the piecewise partitions may be used, as each partition resolves to λ.

Psuedocode Implementation

```
error = gamma - rho
control = lambda

if error > 0:

    lambda = 1 - (1- np.abs(error)) * (1 - control)

else:

    lambda = (1- np.abs(error)) * (control)
```

4.4.2 Parameter Update Edge Cases

The maximum correction occurs when $|e| = 1$, although this extreme value is an edge case that is not actually achievable. The entirety of the supply would have to be completely committed to its portion of the economy ρ, with the desired portion γ being set to the to opposite extreme.

In the case where $e = -1$, meaning γ set to 0 and ρ being 1, the maximum change to λ is $-\lambda$. (See Appendix A) The lowest any given lambda can become is zero. This ensures that λ cannot become negative.

In the case where $e = 1$, meaning γ set to 1 and ρ being 0, the maximum change to λ is $1 - \lambda$. (See Appendix B) If λ was 0, λ^{+} would become 1 and if λ was 1, λ^{+} would become 0. This ensures that λ cannot be greater than 1.

4.4.3 Gain Control over Update Rate

K can be introduced into the control function to exercise control over the rate at which λ will update in response to e.

$$\overrightarrow{\lambda}^{+} = \begin{cases} 1 - \frac{(K - |e|)}{K} \cdot (1 - \lambda) & \text{for} \quad e > 0 \\ \frac{(K - |e|)}{K} \cdot \lambda & \text{for} \quad e < 0 \\ \lambda & \text{for} \quad e = 0 \end{cases}$$

- Note for $e = 0$, either of the piecewise partitions may be used, as each partition resolves to $\frac{K}{K}\lambda$ or just λ.

Where $K \geq 1$.

For the $K = 1$ edge case is equivalent to the control function without control over the update rate. In all other cases, K must chosen greater than 1.

```
if error > 0:

        lambda = 1 - (K- np.abs(error)) / K * (1 - lambda)

else:
        lambda = (K - np.abs(error)) / K * lambda
```

4.4.4 Bound

Introduce ϵ as a bound on both upper and lower values of λ. While approaching the edge cases of 0 and 1 are not a problem in a numerical environment, implementation on a blockchain presents a risk where approaching the edges of 0 and 1 can result equalling that value.

As a measure of safety against this rounding to 0 or 1, we can impose a small value bound, ϵ, on the update to the control function where:

$$\epsilon < \lambda^{+} < (1 - \epsilon)$$

Thus:

```
if lambda_update > 1 - bound:
    lambda_update = 1 - bound

if lambda_update < bound:
    lambda_update = bound
```

4.4.5 Target and Activity Proportion Range of Values

The rebalancing policy acts on the difference between a target proportion and actual activity proportion, and may be extended to more future developed subsystems. Both the target, which may be set through governance, and activity proportions have well-defined ranges of values that they make take on due to their proportional definition and this definition lends itself to that extendibility.

The space of ρ where each component is dependent upon the remaining slack variable. Governance over the choice of γ is a point (in the case of 2 subsystems) within this space.

4.4.6 Appendix A: Edge Case of $e = -1$

1. Negative e partition of λ^{+} function:
$$\overrightarrow{\lambda}^{+} = (1 - |e|) \cdot \lambda$$
2. Adding a $-\lambda$ to both sides:
$$\overrightarrow{\lambda}^{+} - \lambda = (1 - |e|) \cdot \lambda - \lambda$$
3. Distributing:
$$\overrightarrow{\lambda}^{+} - \lambda = \lambda - |e| \cdot \lambda - \lambda$$
4. Simplifying:
$$\overrightarrow{\lambda}^{+} - \lambda = -|e| \cdot \lambda$$
5. Substituting $e = -1$:
$$\overrightarrow{\lambda}^{+} - \lambda = -\lambda$$

4.4.7 Appendix B: Edge Case of $e = 1$

1. Positive e partition of λ^+ function:

$$\overrightarrow{\lambda}^+ = 1 - (1 - |e|) \cdot (1 - \lambda)$$

2. Distributing:

$$\overrightarrow{\lambda}^+ = 1 - [1 - \lambda - |e| + |e| \cdot \lambda]$$

3. Simplifying:

$$\overrightarrow{\lambda}^+ = 1 - 1 + \lambda + |e| - |e| \cdot \lambda$$

4. Simplifying:

$$\overrightarrow{\lambda}^+ - \lambda = |e| - |e| \cdot \lambda$$

5. Substituting $e = 1$:

$$\overrightarrow{\lambda}^+ - \lambda = 1 - \lambda$$