11.9 Rewards Distribution Calculation

This section defines the reward calculation for the proof of stake leader election. Figure 46 defines the pool reward as described in section 5.5.2 of [SL-D1].

- The function maxPool gives the maximum reward a stake pool can receive in an epoch. This is a fraction of the total available rewards for the epoch. The result depends on the pool's relative stake, the pool's pledge and the following protocol parameters:
  - a_0, the leader-stake influence
  - n_{opt}, the optimal number of saturated stake pools
- The function mkApparentPerformance computes the apparent performance of a stake pool. It depends on the protocol parameter d, the relative stake σ, the number n of blocks the pool added to the chain and the total number $\overline{N}$ of blocks added to the chain in the last epoch.

Maximal Reward Function, called $f(s, \sigma)$ in section 5.5.2 of [SL-D1]

$$\mathsf{maxPool} \in \mathsf{PParams} \to \mathsf{Coin} \to [0,\ 1] \to [0,\ 1] \to \mathsf{Coin}$$

$$\mathsf{maxPool}\ pp\ R\ \sigma\ p_r = \left\lfloor \frac{R}{1+a_0} \cdot \left(\sigma' + p' \cdot a_0 \cdot \frac{\sigma' - p' \frac{z_0 - \sigma'}{z_0}}{z_0} \right) \right\rfloor$$

where

$$\begin{aligned} a_0 &= \mathsf{influence}\ pp \\ n_{opt} &= \mathsf{nopt}\ pp \\ z_0 &= 1/n_{opt} \\ \sigma' &= \min(\sigma,\ z_0) \\ p' &= \min(p_r,\ z_0) \end{aligned}$$

Apparent Performance, called $\hat{p}$ in section 5.5.2 of [SL-D1]

$$\mathsf{mkApparentPerformance} \in [0,\ 1] \to [0,\ 1] \to \mathbb{N} \to \mathbb{N} \to \mathbb{Q}$$

$$\mathsf{mkApparentPerformance}\ d\ \sigma\ n\ \overline{N} = \begin{cases} \frac{\beta}{\sigma} & \text{if } d < 0.8 \\ 1 & \text{otherwise} \end{cases}$$

where

$$\beta = \frac{n}{\max(1, \overline{N})}$$

Figure 46: Functions used in the Reward Calculation

Figure 47 gives the calculation for splitting the pool rewards with its members, as described in 6.5.2 of [SL-D1]. The portion of rewards allocated to the pool operator and owners is different than that of the members.

- The $\mathsf{r}_{\mathsf{operator}}$ function calculates the leader reward, based on the pool cost, margin and the proportion of the pool's total stake. Note that this reward will go to the reward account specified in the pool registration certificate.
- The $\mathsf{r}_{\mathsf{member}}$ function calculates the member reward, proportionally to their stake after the cost and margin are removed.

Pool leader reward, from section 5.5.3 of [SL-D1]

$$\mathsf{r}_{\mathsf{operator}} \in \mathsf{Coin} \to \mathsf{PoolParam} \to [0,\ 1] \to (0,\ 1] \to \mathsf{Coin}$$

$$\mathsf{r}_{\mathsf{operator}}\, \hat{f}\ pool\ s\ \sigma = \begin{cases} \hat{f} & \hat{f} \leq c \\ c + \left\lfloor (\hat{f} - c) \cdot (m + (1-m) \cdot \frac{s}{\sigma}) \right\rfloor & \text{otherwise.} \end{cases}$$

where

$$c = \mathsf{poolCost}\ pool$$
$$m = \mathsf{poolMargin}\ pool$$

Pool member reward, from section 5.5.3 of [SL-D1]

$$\mathsf{r}_{\mathsf{member}} \in \mathsf{Coin} \to \mathsf{PoolParam} \to [0,\ 1] \to (0,\ 1] \to \mathsf{Coin}$$

$$\mathsf{r}_{\mathsf{member}}\, \hat{f}\ pool\ t\ \sigma = \begin{cases} 0 & \hat{f} \leq c \\ \left\lfloor (\hat{f} - c) \cdot (1-m) \cdot \frac{t}{\sigma} \right\rfloor & \text{otherwise.} \end{cases}$$

where

$$c = \mathsf{poolCost}\ pool$$
$$m = \mathsf{poolMargin}\ pool$$

Figure 47: Functions used in the Reward Splitting

Finally, the full reward calculation is presented in Figure 48. The calculation is done pool-by-pool.

- The rewardOnePool function calculates the rewards given out to each member of a given pool. The pool leader is identified by the stake credential of the pool operator. The function returns the rewards, calculated as follows:
  - *pstake*, the total amount of stake controlled by the stake pool.
  - *ostake*, the total amount of stake controlled by the stake pool operator and owners
  - σ, the total proportion of stake controlled by the stake pool.
  - $\overline{N}$, the expected number of blocks the pool should have produced.
  - *pledge*, the pool's pledge in lovelace.
  - p_r, the pool's pledge, as a proportion of active stake.
  - *maxP*, maximum rewards the pool can claim if the pledge is met, and zero otherwise.

  - *poolR*, the pool's actual reward, based on its performance.
  - *mRewards*, the member's rewards as a mapping of reward accounts to coin.
  - *lReward*, the leader's reward as coin.
  - *potentialRewards*, the combination of *mRewards* and *lRewards*.
  - *rewards*, the restriction of *potentialRewards* to the active reward accounts.

- The reward function applies rewardOnePool to each registered stake pool.

Calculation to reward a single stake pool

$$
\begin{aligned}
&\mathsf{rewardOnePool} \in \mathsf{PParams} \to \mathsf{Coin} \to \mathbb{N} \to \mathbb{N} \to \mathsf{PoolParam} \\
&\quad \to \mathsf{Stake} \to \mathbb{Q} \to \mathbb{Q} \to \mathsf{Coin} \to \mathbb{P}\,\mathsf{Addr}_{\mathsf{rwd}} \to (\mathsf{Addr}_{\mathsf{rwd}} \mapsto \mathsf{Coin}) \\
&\mathsf{rewardOnePool}\ pp\ R\ n\ \overline{N}\ pool\ stake\ \sigma\ \sigma_a\ tot\ addrs_{rew} = rewards \\
&\quad \textbf{where} \\
&\quad\quad ostake = \sum_{\substack{hk \mapsto c \in stake \\ hk \in (\mathsf{poolOwners}\ pool)}} c \\
&\quad\quad pledge = \mathsf{poolPledge}\ pool \\
&\quad\quad p_r = pledge / tot \\
&\quad\quad maxP = \begin{cases} \mathsf{maxPool}\ pp\ R\ \sigma\ p_r & pledge \leq ostake \\ 0 & \text{otherwise.} \end{cases} \\
&\quad\quad appPerf = \mathsf{mkApparentPerformance}\ (\mathsf{d}\ pp)\ \sigma_a\ n\ \overline{N} \\
&\quad\quad poolR = \lfloor appPerf \cdot maxP \rfloor \\
&\quad\quad mRewards = \\
&\quad\quad\quad \left\{ \mathsf{addr}_{\mathsf{rwd}}\ hk \mapsto \mathsf{r}_{\mathsf{member}}\ poolR\ pool\ \frac{c}{tot}\ \sigma \ \middle|\ hk \mapsto c \in stake,\ \ hk \notin (\mathsf{poolOwners}\ pool) \right\} \\
&\quad\quad lReward = \mathsf{r}_{\mathsf{operator}}\ poolR\ pool\ \frac{ostake}{tot}\ \sigma \\
&\quad\quad potentialRewards = mRewards \cup \{(\mathsf{poolRAcnt}\ pool) \mapsto lReward\} \\
&\quad\quad rewards = addrs_{rew} \lhd potentialRewards
\end{aligned}
$$

Calculation to reward all stake pools

$$
\begin{aligned}
&\mathsf{reward} \in \mathsf{PParams} \to \mathsf{BlocksMade} \to \mathsf{Coin} \to \mathbb{P}\,\mathsf{Addr}_{\mathsf{rwd}} \to (\mathsf{KeyHash} \mapsto \mathsf{PoolParam}) \\
&\quad \to \mathsf{Stake} \to (\mathsf{KeyHash}_{stake} \mapsto \mathsf{KeyHash}_{pool}) \to \mathsf{Coin} \to (\mathsf{Addr}_{\mathsf{rwd}} \mapsto \mathsf{Coin}) \\
&\mathsf{reward}\ pp\ blocks\ R\ addrs_{rew}\ poolParams\ stake\ delegs\ total = rewards \\
&\quad \textbf{where} \\
&\quad\quad total_a = \sum_{\_ \mapsto c \in stake} c \\
&\quad\quad \overline{N} = \sum_{\_ \mapsto m \in blocks} m \\
&\quad\quad pdata = \left\{ hk \mapsto (p,\ n,\ \mathsf{poolStake}\ hk\ delegs\ stake) \ \middle|\ \begin{array}{lcl} hk \mapsto p & \in & poolParams \\ hk \mapsto n & \in & blocks \end{array} \right\} \\
&\quad\quad results = \\
&\quad\quad \left\{ hk \mapsto \mathsf{rewardOnePool}\ pp\ R\ n\ \overline{N}\ p\ s\ \frac{\sum s}{total}\ \frac{\sum s}{total_a}\ total\ addrs_{rew} \mid hk \mapsto (p, n, s) \in pdata \right\} \\
&\quad\quad rewards = \bigcup_{\_ \mapsto r \in results} r
\end{aligned}
$$

Figure 48: The Reward Calculation

11.10 Reward Update Calculation

This section defines the calculation of a reward update. A reward update is the information needed to account for the movement of lovelace in the system due to paying out rewards.

Figure 49 captures the potential movement of funds in the entire system, taking every transition system in this document into account. Value is moved between accounting pots, but the total amount of value in the system remains constant. In particular, the red subgraph represents the inputs and outputs to the "reward pot", a temporary variable used during the reward update calculation in Figure 51. The blue arrows represent the movement of funds that pass through the "reward pot".

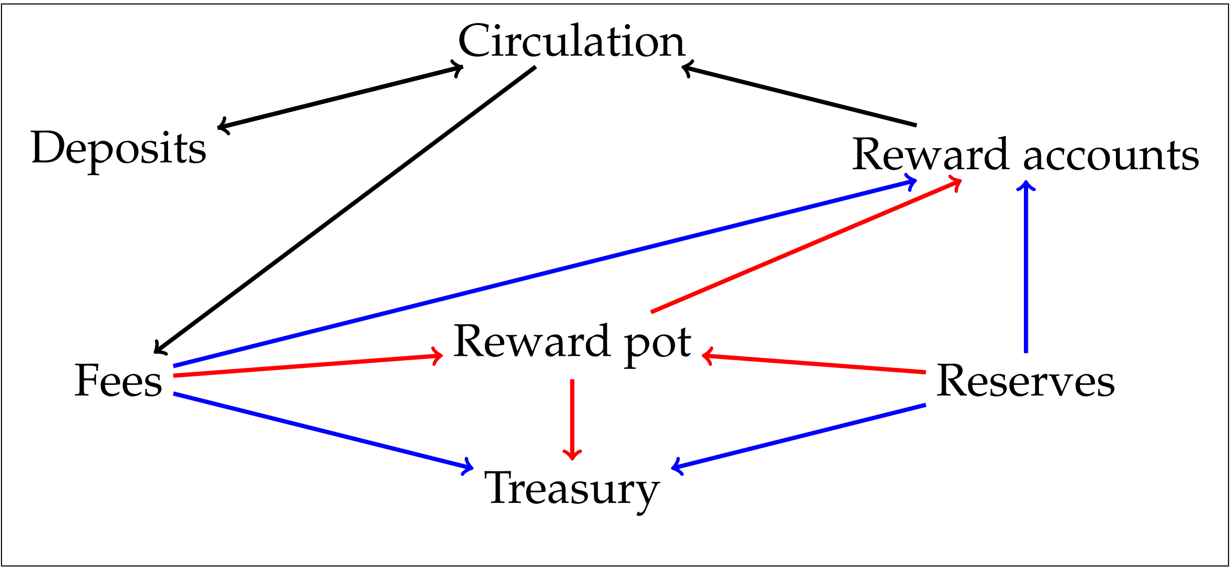

Figure 49: Preservation of Value

Figure 50 defines a reward update. It consists of four pots:

- The change to the treasury. This will be a positive value.
- The change to the reserves. This will be a negative value.
- The map of new individual rewards (to be added to the existing rewards).
- The change to the fee pot. This will be a negative value. rewards.

Reward Update

$$\mathsf{RewardUpdate} = \left(\begin{array}{rcll} \Delta t & \in & \mathsf{Coin} & \text{change to the treasury} \\ \Delta r & \in & \mathsf{Coin} & \text{change to the reserves} \\ rs & \in & \mathsf{Addr}_{\mathsf{rwd}} \mapsto \mathsf{Coin} & \text{new individual rewards} \\ \Delta f & \in & \mathsf{Coin} & \text{change to the fee pot} \end{array}\right)$$

Figure 50: Rewards Update type