Amplifying the Weight-copying Penalty in Bittensor (Working Paper)

Opentensor

June 26, 2024

Abstract

We propose an approach to amplify the penalty to validators who copy or act reactively to other validators' actions. This approach is designed to operate concurrently with the commitment scheme [1]. Once this penalty reaches a certain threshold, it becomes preferable to either perform miner-evaluation work as intended or delegate their stakes to other validators who perform such work. The approach that we propose is an adaptive alpha personalized to each miner.

1 Introduction

This paper is a companion to [1]. We also consider the problem of validators who selfishly report weights that optimize their utility, as opposed to reporting weights that are informative on the actual performance of miners. We present an additional solution approach that amplifies the disadvantage of validators who copy the consensus weights. This approach called "liquid alpha" sets an adaptive and personalized parameter $\alpha_t(i)$ for each round t and each miner i.

We present the model and define the notions of weight-copying penalty and miner-fairness in Section 2. Section 3 presents a heuristic solution approach (dubbed "liquid alpha") to amplify the weight-copying penalty. Once this penalty is large enough, there is no reason for a validator to copy another validator's weights. We present in Section 4 empirical evidence of the effectiveness—as well as the limitation—of liquid alpha by simulating an immediate version of the penalty. Section 5 outlines open problems. We refer the reader to [1] for a description of related works.

2 Model

The problem of weight copying arises in the interaction between validators[1]. The simplest model that describes this interaction is a repeated game between a set of validators. In such a setting, weight copying is simply the tit-for-tat strategy [2] where one participant j picks another participant k, and at every round t plays the same action that k played at the previous round $t-1$. This tit-for-tat strategy has the nice property of creating consensus and improving the long-term outcome for all players. In Bittensor, however, we do not simply want consensus: we also want fair weights for miners. Our model must therefore include miners.

We consider the following model of a repeated game with additional signals from miners. The set of players is the set of validators $\mathcal{Z}$. There is a set of miners $\{1, \ldots, d\}$. At every round $t = 1, 2, \ldots$, each validator j has the option to observe a signal $\mathbf{v}_t^j \in \mathcal{X} = \Delta^d$, where Δ^d denotes the simplex of probability vectors $\{z \in \mathbb{R}_+^d : \sum_{i=1}^d z_i = 1\}$. Each of the d elements corresponds to a miner's normalized performance. We assume that the signals[2] $\mathbf{v}_t^j$ are independent and identically distributed for every round t. The distribution of $\mathbf{v}_t^j$ is however unknown. Each validator j also has a stake value S_j associated to it that is known by all players; we assume that it stays fixed over time[3]. Without loss of generality, we assume that $\sum_{j \in \mathcal{Z}} S_j = 1$.

At round t, each validator j takes a pair of actions

1. $a_t^j \in \{0, 1\}$, representing whether the validator does make the effort of evaluating all the miners. The signal v_t^j is only observed by j if $a_t^j = 1$.
2. $w_t^j \in \mathcal{X}$, which is the reported weight vector.

The report profile at round t is a profile of reported weights in $\mathcal{X}^{|\mathcal{Z}|}$. There pairs of actions are generated by a strategy as follows. The set of possible histories observed by j up to round $t > 1$ is denoted

$$\mathcal{H}_{t-1} = (\underbrace{\mathcal{X}^{|\mathcal{Z}|}}_{\text{reports}} \times \underbrace{\mathcal{X}}_{\text{observed signal}})^{t-1}. \tag{1}$$

A strategy σ for j is a sequence of mappings from history and current signal to an action-pair:

$$\sigma_t : \mathcal{H}_{t-1} \times \underbrace{\mathcal{X}}_{\text{current signal}} \to \underbrace{\{0,1\}}_{\text{effort}} \times \underbrace{\mathcal{X}}_{\text{reported weight}}, \quad \text{for all } t \geq 1. \tag{2}$$

[1]Throughout this paper, a "validator" refers to a subnet validator, as opposed to a validator (operated by the Opentensor Foundation) that validates blocks on the Bittensor blockchain. Likewise, a "miner" refers to a subnet miner, as opposed to miners in that propose blocks in blockchains such as Bitcoin.

[2]The sequence $\mathbf{v}_t^j$ essentially replaces the notion of a static type in Bayesian games.

[3]In practice, this may not be true since validators and their nominators may re-stake the dividends that they earn over time.

Observe that j does not observe the action a_t^k for other validators, nor the signals $\mathbf{v}_t^k$ for other validators. The action-pair can be written as the strategy applied to the history H_{t-1} and signal v_t^j:

$$[a_t^j, w_t^j(a_t^j)] = [\sigma_t^a(H_{t-1}), \sigma_t^w(H_{t-1}, v_t^j a_t^j)], \tag{3}$$

where the decision to spend the effort a_t^j is only a function of the history H_{t-1}, and the reported weight is a function of the current signal v_t^j whenever $a{*}_t = 1$.

2.1 Yuma Consensus and utility functions

To define the utility function for validators, we first introduce the calculation of dividend shares. Let $w_t^j(i)$ denote the validator j's weight on miner i at time t, and let $w_t^j \in \Delta^d$ denote the normalized vector of weights for j. The consensus $\bar{w}_t(i)$ of miner i is calculated as follows. First, we sort the pairs $\{(w_t^j(i), S_j)\}_{j\in\mathcal{Z}}$ from largest to smallest weight value via permutation $\pi : \mathcal{Z} \to \mathcal{Z}$:

$$w_t^{\pi(1)}(i) \geq w_t^{\pi(2)}(i) \geq \dots \tag{4}$$

$$S_{\pi(1)}, S_{\pi(2)}, \dots \tag{5}$$

We then look for the smallest partial sum of the sequence $S_{\pi(m)}$ above a parameter $\kappa > 0$:

$$M^* = \arg \min_{M=1,\dots,|\mathcal{Z}|} \sum_{m=1}^{M} S_{\pi(m)} \tag{6}$$

$$\text{subject to:} \quad \sum_{m=1}^{M} S_{\pi(m)} > \kappa. \tag{7}$$

Finally, we set $\bar{w}_t(i) = w_t^{\pi(M^*)}(i)$.

Next, we define the consensus-clipped weight from validator j to miner i.

$$\hat{w}_t^j(i) = \min(w_t^j(i), \bar{w}_t(i)). \tag{8}$$

Validator trust is definded as the total validator weight after consensus-clipping:

$$T^j = \sum_{i=1}^{d} \hat{w}_t^j(i). \tag{9}$$

With bonds penalty $\beta \in [0, 1]$, e.g., $\beta = 1$ in the implementation, we define stake weighted as

$$\tilde{w}_t^j(i) = (1 - \beta) w_t^j(i) + \beta \hat{w}_t^j(i). \tag{10}$$

For a fixed sequence α_t, the validator bond that validator j has over miner i is

$$\Delta_t^j(i) = \frac{S_i \tilde{w}_t^j(i)}{\sum_{k\in\mathcal{Z}} S_k \tilde{w}_t^k(i)}, \tag{11}$$

$$B_t^j(i) = B_{t-1}^j(i) - \alpha_t[B_{t-1}^j(i) - \Delta_t^j(i)] \tag{12}$$

$$= \alpha_t \Delta_t^j(i) + (1-\alpha_t) B_{t-1}^j(i) \tag{13}$$

$$= \alpha_t \Delta_t^j(i) + (1-\alpha_t)\alpha_{t-1}\Delta_{t-1}^j(i) + \ldots + (1-\alpha_t)\alpha_{t-1}\ldots\alpha_1 B_1^j(i), \tag{14}$$

$$B_1^j(i) = 0, \quad \text{for all } i. \tag{15}$$

Lastly, validator j's dividend-share[4] is the sum of bonds scaled by miner incentives I_t^i:

$$D_t^j(w_t^j, \mathbf{w}_t^{-j}; \alpha_{1\ldots t}) = \sum_{i\in\mathcal{M}} B_t^j(i) \cdot I_t^i, \tag{16}$$

where $\alpha_{1\ldots t}$ denote the sequence $(\alpha_1, \ldots, \alpha_t)$.

Observe that the process $B_t^j(i)$ is an instance of stochastic approximation [3], and converges under appropriate conditions. In the current implementation (before liquid alpha), we have a constant sequence $\tilde{\alpha}_t = \alpha$. Observe that when $\tilde{\alpha}_t = \alpha$, we have

$$B_t^j(i) = \alpha\Delta_t^j(i) + \sum_{\ell=1}^{t-1}(1-\alpha)\alpha^\ell \Delta_{t-\ell}^j(i) \tag{17}$$

$$= \alpha\frac{S_i \tilde{w}_t^j(i)}{\sum_{k\in\mathcal{Z}} S_k \tilde{w}_t^k(i)} + \sum_{\ell=1}^{t-1}(1-\alpha)\alpha^\ell \frac{S_i \tilde{w}_{t-\ell}^j(i)}{\sum_{k\in\mathcal{Z}} S_k \tilde{w}_{t-\ell}^k(i)}. \tag{18}$$

Next, we the miners' incentives:

$$I_t^i = I_t^i(\mathbf{w}) = \frac{\sum_{j\in\mathcal{Z}} S_j \hat{w}_t^j(i)}{\sum_{i=1}^d \sum_{k\in\mathcal{Z}} S_k \hat{w}_t^k(i)} \tag{19}$$

Finally, when the sequence of signals observed by all validators is $\{\mathbf{v}\}$, the utility function of j adopting a strategy σ is

$$u_j(\sigma^j, \sigma^{-j}; \mathbf{v}) = \sum_{t=1}^{\infty} \gamma^t (D_t^j(w_t^j, \mathbf{w}_t^{-j}) - \mu a_t^j), \tag{20}$$

where μ is the cost of the effort of evaluating all the miners in one round.

2.2 Weight-copying penalty

In this paper, our objective is to design a mechanism that amplifies the weight-copying penalty and in turn amplifies the disadvantage of a validator that copies weights.

[4]Fractional share of the total amount of dividend from the subnet going to validator j.

Definition 1 (Weight-copying penalty)**.** *Let $\bar{w}_t \in \mathbb{R}^d_+$ denote the consensus weight vector at round t. Let validator j denote a validator whose weights follow the consensus without delay: $w^j_t = \bar{w}_t$. Consider a validator k with a delay $m > 0$ on the weights:*

$$\sigma^j = (a^j_t, w^j_t)_{t=1,2,\ldots}, \tag{21}$$

$$\sigma^k = (a^k_t, w^k_t)_{t=1,2,\ldots}, \tag{22}$$

$$w^k_t = w^j_{t-m}. \tag{23}$$

Suppose without loss of generality that validator j and k both have one unit of TAO in stake $S_j = S_k = 1$. The long-term weight-copying penalty of validator k relative to validator j is

$$\Gamma(j,k) = u_j(\sigma^j, \sigma^{-j}; \mathbf{v}) - u_k(\sigma^k, \sigma^{-k}; \mathbf{v}). \tag{24}$$

The (immediate) weight-copying penalty of validator k with respect to validator j is the difference in dividend-share[5] *at round t:*

$$G_t(j,k) = D^j_t(w^j_t, \boldsymbol{w}^{-j}_t) - \mu a^j_t - D^k_t(w^k_t, \boldsymbol{w}^{-k}_t) + \mu a^k_t. \tag{25}$$

Remark 1. *The notion of weight-copying penalty is similar to the advantage of the leader in the Stakelberg model of competition [4, 5]. A similar concept of last-mover advantage or first-mover disadvantage exists in ultimatum bargaining games.*

Our objective is to show that there is a sequence α_t such that for every delay m, the weight-copying penalty is bounded from below:

$$\Gamma(j,k) \geq \epsilon(m). \tag{26}$$

Moreover, if $\epsilon(m)$ is larger than the delegate-take of j, then there is no reason for a validator k to copy validator j when k can instead delegate to j.

2.3 Harmless or beneficial to miners

Next, we define criteria for a mechanism to be harmless or beneficial to miners. The harmless notion is that of fairness, which we define as follows. In the verifiable setting, we say that a mechanism is fair among miners if for every validator j and every round t: if two miners i_1 and i_2 have the (verifiable) error rate, and if $w^k_t(i_1) = w^k_t(i_2)$ for all $k \neq j$, then validator j maximizes its dividend-share by assigning the same weight to i_1 and i_2. In other words, if

$$w^{j*}_t \in \arg\max_{w \in \mathbb{R}^d} D^j_t(w, \mathbf{w}^{-j}_t), \tag{27}$$

then $w^{j*}_t(i_1) = w^{j*}_t(i_2)$. This definition using the one-round dividend-share assumes that the validators are myopics; in general, we can present an alternative definition using the strategy and utility function.

[5]The cost of validation is the same for j and k at every round t.

In the following section, we want to find an adaptive sequence α_t that is adapted to the filtration $\mathcal{F}_t$ generated by all observed random variables up to round t and that maximizes the first mover advantage $\Gamma(j,k)$ for every pair of validators j and k, while ensuring fairness among miners.

3 Liquid alpha solution

In this section, we propose a method ("liquid alpha" of Algorithm 1) to compute a sequence α_t that amplifies the weight-copying penalty. This method is adaptive—in the sense that each α_t is a function of all previously observed variables at round t, and personalized to each miner—in the sense that each miner i is assigned a distinct value $\alpha_t(i)$.

Algorithm 1 Liquid Alpha

```
Input: Let $\bar{w}_t = (\bar{w}_t)_{i=1,\dots,d} \in \mathbb{R}^d$ denote the vector of consensus weights of all miners.
Input: Let $\alpha^H$ and $\alpha^L$ be parameters with default values $\alpha^H = 0.9$ and $\alpha^L = 0.7$.
Input: Quantiles $q_H$ and $q_L$ with default values $q_H = 0.75$ and $q_L = 0.25$.
for $t = 1, 2, \dots$ do
    Set $C_t^H$ to $q_H$-quantile of $\bar{w}_t$.
    Set $C_t^L$ to $q_L$-quantile of $\bar{w}_t$.
    if $C_t^H \leq C_t^L$ then
        Set fallback value $\alpha_t(i) = 0.9$ for all $i = 1, \dots, d$,
    else
        Compute
            $$a = \frac{\log(1/\alpha^H - 1) - \log(1/\alpha^L - 1)}{C_t^L - C_t^H}, \quad (28)$$
            $$b = \log(1/\alpha^L - 1) + aC_t^L, \quad (29)$$
            $$\tilde{\alpha}_t(i) = \frac{1}{1 + \exp(-a\bar{w}_t(i) + b)}, \quad i = 1, \dots, d, \quad (30)$$
            $$\alpha_t(i) = [\tilde{\alpha}_t(i)]_{\alpha^L}^{\alpha^H}, \quad i = 1, \dots, d, \quad (31)$$
        where $[\cdot]_{\alpha_L}^{\alpha^H}$ denotes a clipping function.
    end if
    Output: $\alpha_t = (\alpha_t(i))_{i=1,\dots,d}$.
end for
```

If we let α_t denote the sequence output by Algorithm 1, and $\tilde{\alpha}_t = 0.9$ denote the constant sequence taking the value of the current implementation of Bittensor, we define the amplification factor of the weight-copying penalty as

$$H_t(j,k) = \frac{G_t(j,k;\alpha_t)}{G_t(j,k;\tilde{\alpha}_t)}. \tag{32}$$

Observe that the denominator $G_t(j,k;\tilde{\alpha}_t)$ is greater than zero due to the commitment scheme introduced in [1].

4 Empirical evidence

In this section, we present empirical evidence that the liquid alpha algorithm of Algorithm 1 indeed amplifies the advantage of an honest validator j that performs the validation work and reports the observed signal truthfully, over another validator k that does not spend the effort of the validation work and copies the consensus weight. We take as our baseline the advantage of a over b in setting with the commitment scheme of [1].

4.1 Simulated setting

The evidence in question comes from a simulation of the interactions of three validators j, k, c and four miners A, B, C, D. We assume that every validator in $\{j, k, c\}$ observes the same deterministic signal v_t if it chooses to perform the validation work. We introduce four miners in order to model a range of miners with different performance or consensus values $\bar{w}_t(i)$.

Let $\rho > 0$ denote the initial signal for miner A, and $\delta > 0$ the jump in signal for miner A. We assume that the signal sequences for the four miners are piecewise-linear as follows:

$$v_t(A) = \begin{cases} \rho & t \in [0,1) \\ \rho + \delta t/3 & t \in [1,3) \\ \rho + \delta & t \in [3,\infty), \end{cases} \tag{33}$$

$$v_t(B) = 0.5 - v_t(A), \tag{34}$$

$$v_t(C) = 0, \tag{35}$$

$$v_t(D) = 0.5. \tag{36}$$

We assume that both validator j and k have 1 TAO stake. The three validators adopt the following strategies:

1. Validator j does the honest work ($a_t^j = 1$) and reports the normalized observed signals: $w_t^j = \frac{v_t}{||v_t||_1}$. Observe that this normalization restricts ρ to values between 0 and 0.5, and δ to values between 0 and $0.5 - \rho$.

2. Validator c has 98 TAO in stake and also does the honest work ($a_t^c = 1$) and reports the observed signals: $w_t^c = \frac{v_t}{||v_t||_1}$.

3. Validator k does no work ($a_t^k = 0$) and reports the observed consensus with five rounds of delay:

$$w_t^k = \begin{cases} [\rho, 0.5-\rho, 0, 0.5] & t \in [0,5) \\ \bar{w}_{t-5} & t \in [5,\infty) \end{cases} \tag{37}$$

$$\approx \begin{cases} [\rho, 0.5-\rho, 0, 0.5] & t \in [0,5) \\ \frac{v_{t-5}}{||v_{t-5}||_1} & t \in [5,\infty). \end{cases} \tag{38}$$

   The delay of five rounds corresponds to the typical duration of the commit-reveal weights (CRW) interval in the commitment scheme of [1]. The red and blue lines in Figures 2 and 3 illustrate $w_t^j(A)$ and $w_t^j(B)$.

Each miner $X \in \{A, B, C, D\}$ gets the following vector of weights for rounds $t > 5$:

$$[w_t^j(X), w_t^k(X), w_t^c(X)]. \tag{39}$$

In turn, we have

$$\bar{w}_t(A) = F((S_j, w_t^j(A), (S_k, w_t^k(A), (S_c, w_t^c(A))) \tag{40}$$

$$C_t^L = q_{0.25}(\bar{w}_t(A), \ldots, \bar{w}_t(D)), \tag{41}$$

$$C_t^H = q_{0.75}(\bar{w}_t(A), \ldots, \bar{w}_t(D)). \tag{42}$$

Note that the small stakes for validator j and validator k are such that their weights have negligible effect on the consensus $\bar{w}_t$.

4.1.1 Simulation

In this section, we assume that the validation cost is zero ($\mu = 0$) and measure the immediate weight-copying penalty $G_t(j,k)$ for round t (cf. (25)). We also measure the amplification factor H of the weight-copying penalty (cf. (32)) of liquid alpha (Algorithm 1) on top of the commitment scheme introduced in [1].

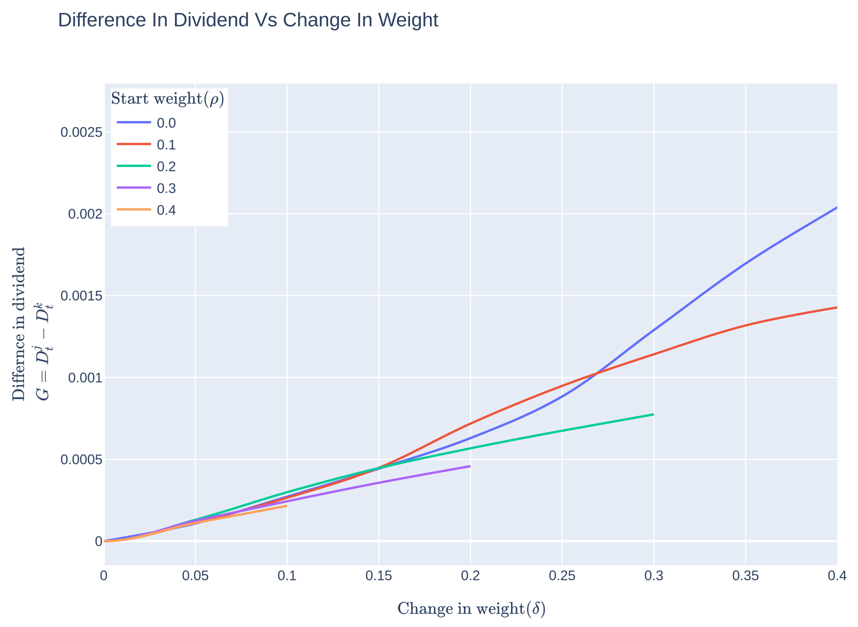

Figure 1: Immediate weight-copying penalty G as a function of the initial value $\rho \in [0, 0.5]$ and the jump $\delta \in [0, 0.5 - \rho]$ in the signal of miner A.

Figure 1 shows for different values of the initial signal ρ for miner A, that the immediate weight-copying penalty G is positive and monotone increasing in the increment δ in signal. This suggests that liquid alpha is most useful when there is a significant increment δ in the signal of miner A.

Figures 2 and 3 show that liquid alpha (Algorithm 1) reduces of the dividend-share of the copier (D_t^k) from the green line to the purple line.

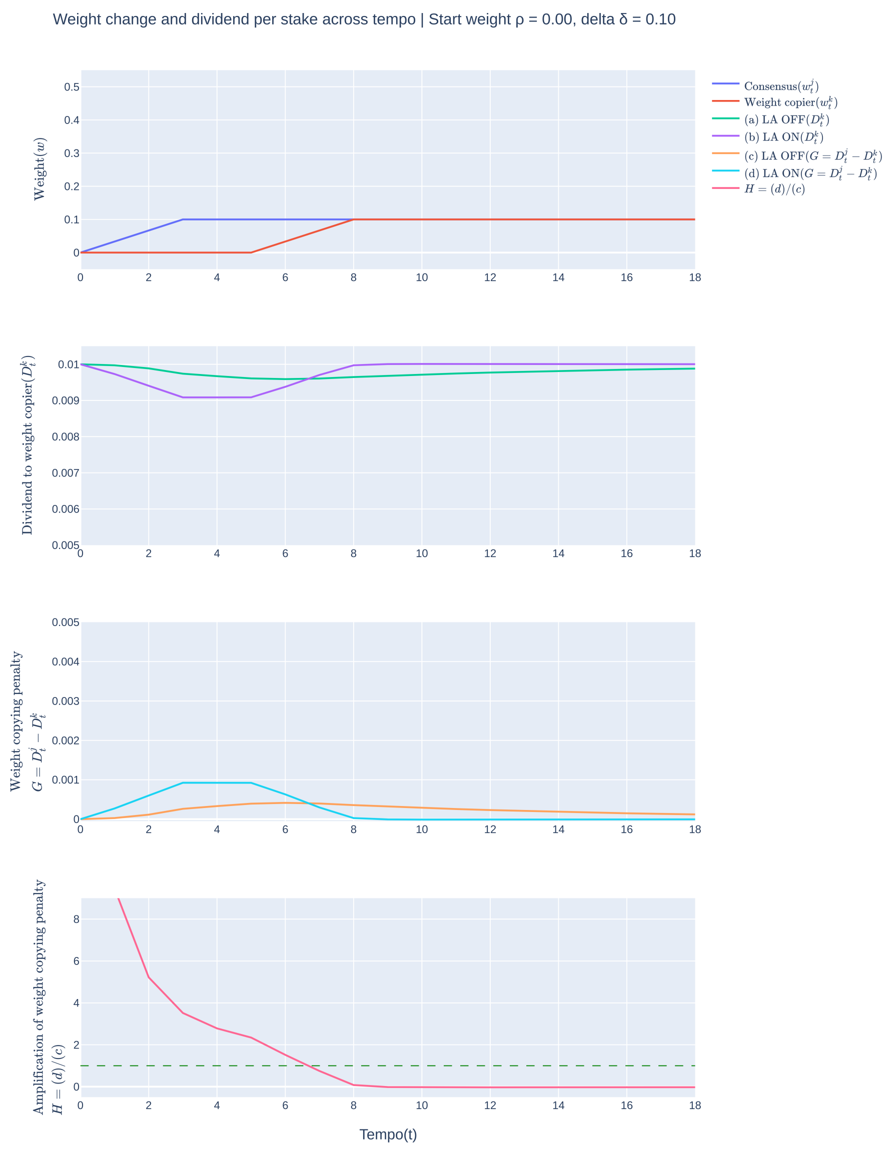

Figure 2: Dynamics of $w_t^j(A)$ thus consensus (blue), $w_t^k(A)$ (red), dividend-share to k with liquid alpha (purple) and without liquid alpha (cyan), weight-copying penalty with liquid alpha (light-blue) and without liquid alpha (orange), and amplification factor of weight-copying penalty (pink) when the jump in signal is $\delta = 0.1$.

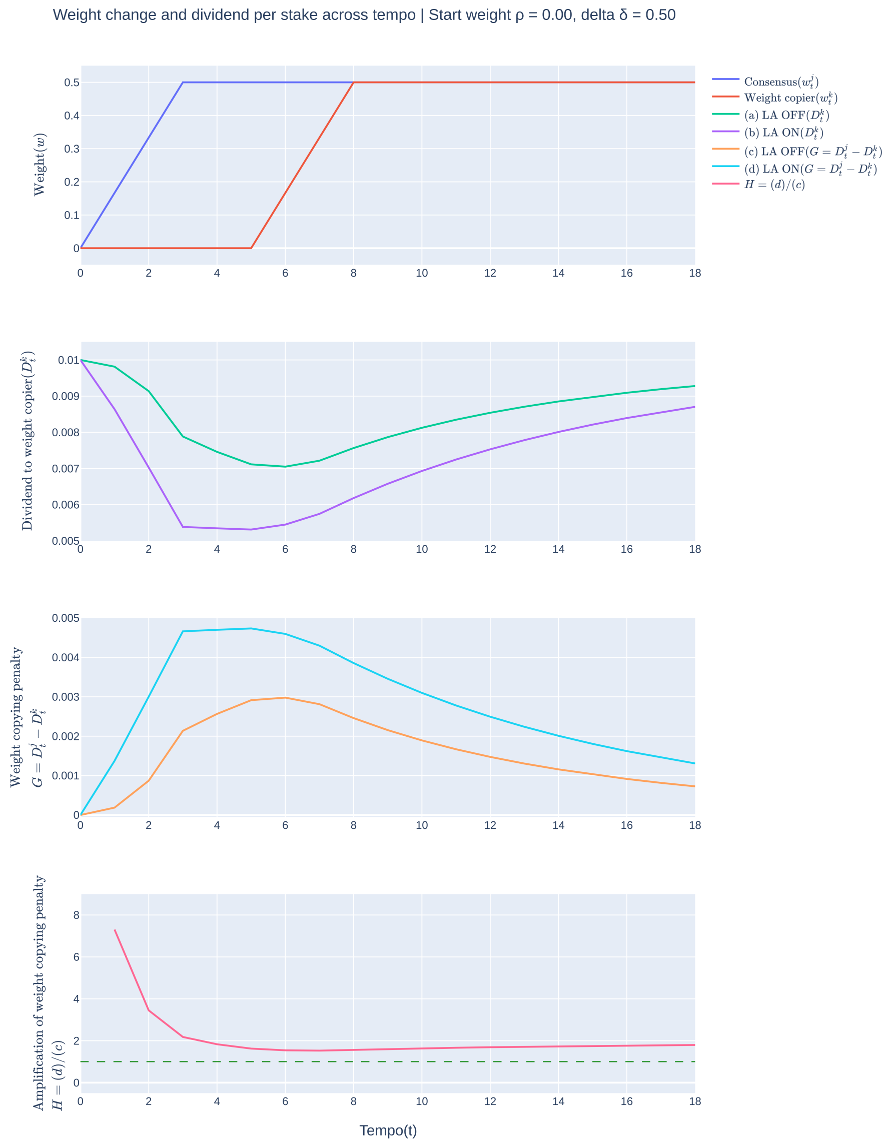

Figure 3: Same with Figure 2 but with $\delta = 0.5$.

In Figures 2 and 3, when the pink line is above 1, liquid alpha (Algorithm 1) reduces the dividend-share (D_t^k) of the copier k. The slopes of the purple line and green line indicate how much dividend-share the copier k accumulates with and without liquid alpha.

In the case of a signal jump of $\delta = 0.5$ as illustrated in Figure 3 - subfigure 1.

From Figure 3 - subfigure 2, we can see that Algorithm 1 gives a large reduction in D_t^k for the weight copier (k) for 33% from 0.0071 to 0.0053 at $t = 5$. Figure 3 - subfigure 4 shows that liquid alpha is effective from round $t = 0$ to $t = 18$ as the pink line at is always larger than 1.

In the case of a signal jump of $\delta = 0.1$ as illustrated in Figure 2 - subfigure 1, From Figure 2 - subfigure 2, there is only 6% in the reduction in D_t^k for the weight copier (k) by Algorithm 1 at $t = 5$. In Figure 2 - subfigure 3, observe that liquid alpha amplifies the weight-copying penalty from the orange line to the light blue line. However, this amplification turns into a reduction for $t \geq 8$: contrary to our objective, liquid alpha helps the weight copier k for later rounds. The same was as well shown in Figure 2 - subfigure 4 where liquid alpha is only effective from round $t = 0$ to $t = 8$, and fail to satisfy its objective as the pink line goes below 1.

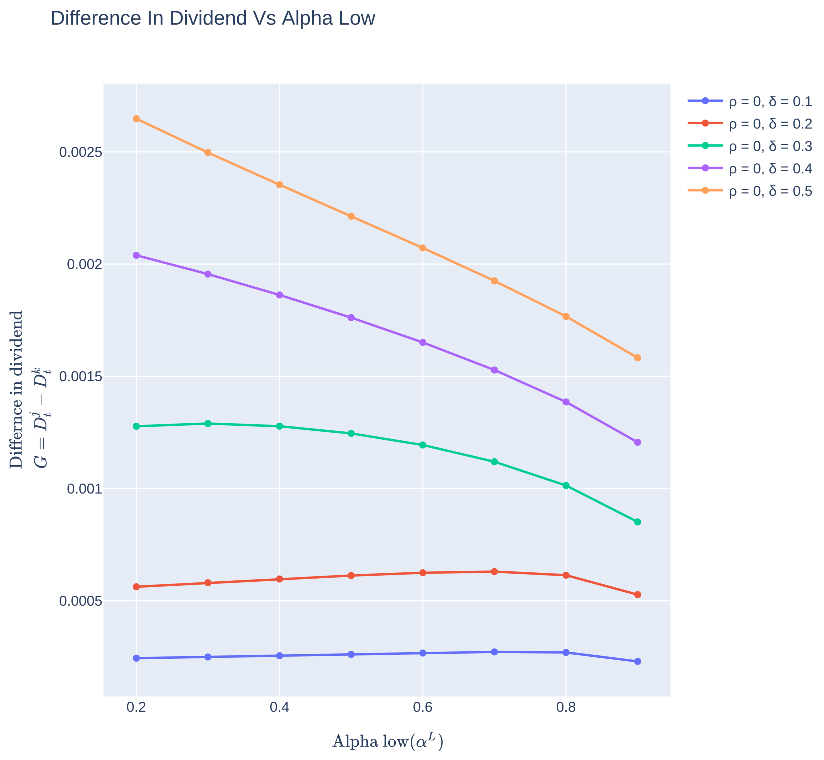

Figure 4: Optimal α^L for different signal jump values δ and a fixed initial signal $\rho = 0$. Recall that Figure 1 shows that there is negligible dependence on ρ. Note that when $\alpha^L = \alpha^H = 0.9$, this would be equivalent to when liquid alpha is in-active

Figure 4 shows that when the signal jump δ for miner A is 0.4, liquid alpha with an appropriate parameter α^L achieves an immediate weight-copying penalty G of the order of 69% from 0.0012 to 0.0020. Moreover, this G is not always maximal at a low value of α^L: for instance, when the signal jump is $\delta = 0.1$, the optimal value of G is achieved at $\alpha^L = 0.7$.

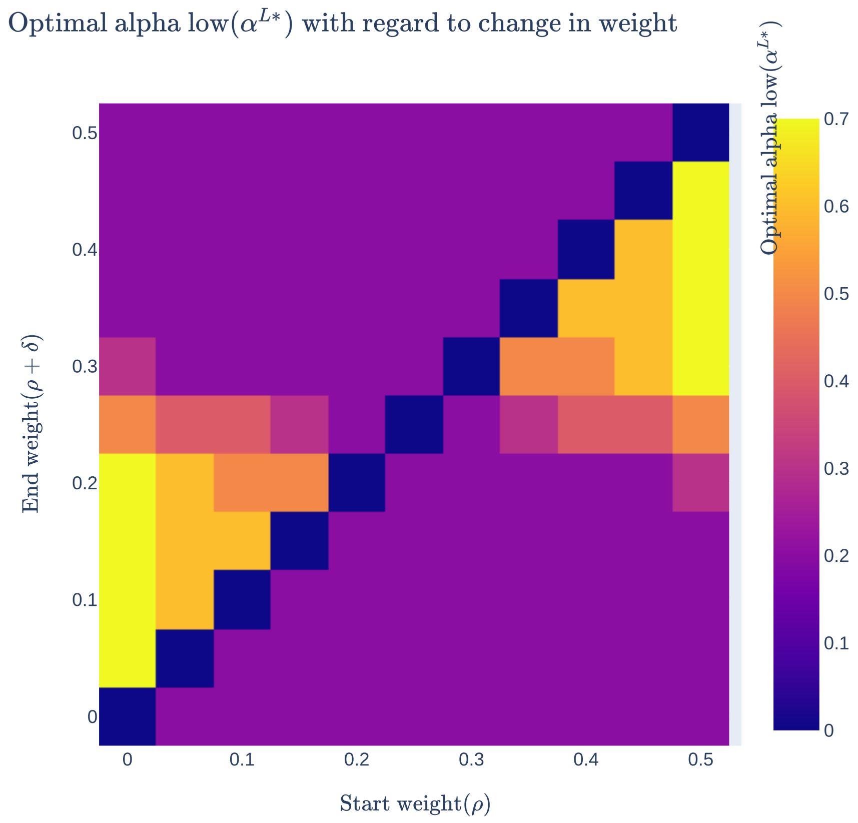

Figure 5: Optimal α^L for different change in weight, given that $\alpha^H = 0.9$. Note that there is a symmetry pattern: the optimal value of α^L for $(\rho, \rho + \delta) = (x, y)$ is the same as for $(\rho, \rho + \delta) = (0.5 - x, 0.5 - y)$.

4.2 Empirical Subnet data

In this section, we extend the analysis from the simulated setting of to the actual Bittensor setting using historical data. Table 1 shows that liquid alpha reduces the duration of the commit-reveal weight (CRW) interval (cf. [1] for definition) significantly. Moreover, any commit reveal weight interval set higher than specified would result in weight copiers earning less than honest validators. Note that Table 1 omits subnets where neither the commitment scheme of [1], nor liquid alpha (Algorithm 1) help amplify the weight-copying penalty enough to dissuade validators from copying weights.

Subnet	CRW interval duration	CRW interval duration with liquid alpha	reduction	optimal α^L
23	5	3	2	0.2
13	5	3	2	0.7
15	13	11	2	0.2
8	13	9	4	0.2
30	13	9	4	0.2
19	15	7	8	0.2
22	15	9	6	0.2
3	15	7	8	0.7

Table 1: Value of the parameter α^L that maximizes the immediate weight-copying penalty G (25) for different subnets during the block 2987500 to 3009100 (corresponding to the period from May 19, 2024 00:56:12 to May 22, 2024 01:10:48, UTC [6]).

5 Future Work

The following open questions remain.

1. With parameters $q_H = 0.75$ and $q_L = 0.25$, we see that liquid alpha does not amplify the weight-copying penalty on all subnets due to the fallback value of 0.9 for α_t in Algorithm 1. A possible solution is to implement adaptive values for q_H and q_L personalized to each subnet.

2. We would like to show that the proposed personalization of $\alpha_t(i)$ for each miner i does not introduce unfairness between miners.

3. We would like to characterize the convergence of the bond process for different sequences α_t in the style of [3].

References

[1] Opentensor, "Weight copying in Bittensor." `https://docs.bittensor.com/papers/BT_Weight_Copier-29May2024.pdf`, 2024.

[2] A. Rapoport and A. Chammah, *Prisoner's Dilemma: A Study in Conflict and Cooperation.* Ann Arbor paperbacks, University of Michigan Press, 1965.

[3] V. S. Borkar, *Stochastic approximation: a dynamical systems viewpoint*, vol. 48. Springer, 2009.

[4] W. Güth, R. Schmittberger, and B. Schwarze, "An experimental analysis of ultimatum bargaining," *Journal of Economic Behavior Organization*, vol. 3, no. 4, pp. 367–388, 1982.

[5] R. Weber, C. Camerer, and M. Knez, "Timing and Virtual Observability in Ultimatum Bargaining and "Weak Link" Coordination Games," *Experimental Economics*, vol. 7, pp. 25–48, February 2004.

[6] Taostats, "Bittensor (TAO) blockchain explorer." `https://x.taostats.io/block/2987500`, 2024.