Taraxa PBFT

Version 1.0

February 01, 2025

Contents

1 Consensus Protocol

Taraxa consensus is based on Algorand pBFT algorithm. User u starts new period p the moment he receives 2t + 1 cert-votes for some value v and valid block B_v, he starts new round r the moment he receives 2t + 1 next-votes for some value v or $\perp$ for round r - 1

1.1 Timing Parameters

- λ = 2000 ms: Intuitively corresponds to the time it takes for a small message (e.g., a vote) to propagate in good network conditions.
- $\lambda_1^{\min}$ = 500 ms: Minimum time the dynamic algorithm allows for a small message to propagate under ideal conditions for round 1, reverting to λ in future rounds if insufficient.
- $\lambda_1^{\max}$ = 1500 ms: Maximum time the dynamic algorithm allows for a small message to propagate under ideal conditions for round 1, reverting to λ in future rounds if insufficient.
- Λ = 17000 ms: Time it takes for a big message (e.g., a block) to propagate in reasonable network conditions.
- Λ_1 = 4000 ms: Time it takes for a big message to propagate in good conditions for round 1, reverting to Λ in future rounds if insufficient.
- λ_r: selected λ value for current round
- Λ_r: selected Λ value for current round

1.2 Protocol Instructions

User u starts period p round 0 when he first sees 2t + 1 cert-votes for some value v and valid block B_v. If u sees 2t + 1 next-votes for some value v or $\perp$, then u starts new round $r' + 1$.

Whenever u starts a new period (or a new round), he resets timer_u, used to decide when to vote for each step.

1.3 Round (p, r) parameters values

When user u starts round (p, r), they reset their timer_u to 0 and other constants as follows:

- If $r = 1$:
  - $\lambda_r \in (\lambda_1^{\min}, \lambda_1^{\max})$
  - $\Lambda_r = \Lambda_1$
  - $st_u^r = \perp$
- Otherwise if $r >= 2$:
  - $\lambda_r = \lambda$
  - $\Lambda_r = \Lambda$
  - $st_u^r = v$

1.4 Round (p, r) Voting Instructions

The voting instructions are as follows:

Step 1: **Proposal** - When $\text{timer}_u = 0$:

- If $r = 1$ or $r > 1$ and u has received 2t+1 next-votes for $\perp$ from round $(p, r-1)$, then u assembles a new block proposal v_u and propagates v_u together with his round r credential.
- Otherwise, if $r > 1$ and u has received 2t+1 next-votes for some value v $\neq \perp$ from $(p, r-1)$, then u proposes v, which he propagates together with his round r credential.

Step 2: **Filtering** - When $\text{timer}_u = 2\lambda_r$:

- If $r = 1$ or if $r > 1$ and u has received received 2t+1 next-votes for $\perp$, then u selects the proposal with the minimum credential and soft-votes for it.
- Otherwise, if $r > 1$ and u has received 2t+1 next-votes for some value v $\neq \perp$, then u soft-votes for v.

Step 3: **Certifying** - While $\text{timer}_u \in (2\lambda_r, \max(4\lambda_r, \Lambda_r))$:

– If u receives 2t+1 soft-votes for some value v $\neq \perp$ and a valid block B_v, then u cert-votes v.

Step $s = 2n$, where $n \in (2, \infty)$: **First Finishing Step** - When $\text{timer}_u = \max(4\lambda_r, \Lambda_r) + (s-4)\lambda_r$:

– If u has certified some value v for round r, he next-votes v.

– Else if ($r \geq 2$ and i has seen $2t+1$ next-votes for $\perp$ for round $r-1$), he next-votes $\perp$.

– Else he next-votes his starting value st_u^r.

Step $s = 2n+1$, where $n \in (2, \infty)$: **Second Finishing Step** - When $\text{timer}_u = \max(4\lambda_r, \Lambda_r) + (s-5)\lambda_r + 100\text{ms}$:

– If u sees $2t+1$ soft-votes for some value $v \neq \perp$ for round r, then u next-votes v.

– If ($r \geq 2$ and i sees $2t+1$ next-votes for $\perp$ for round $r-1$ and i has not certified in round r), then u next-votes $\perp$.

Λ_r is exponentially increased in second finish step after reaching step 15, max value is 60000 ms