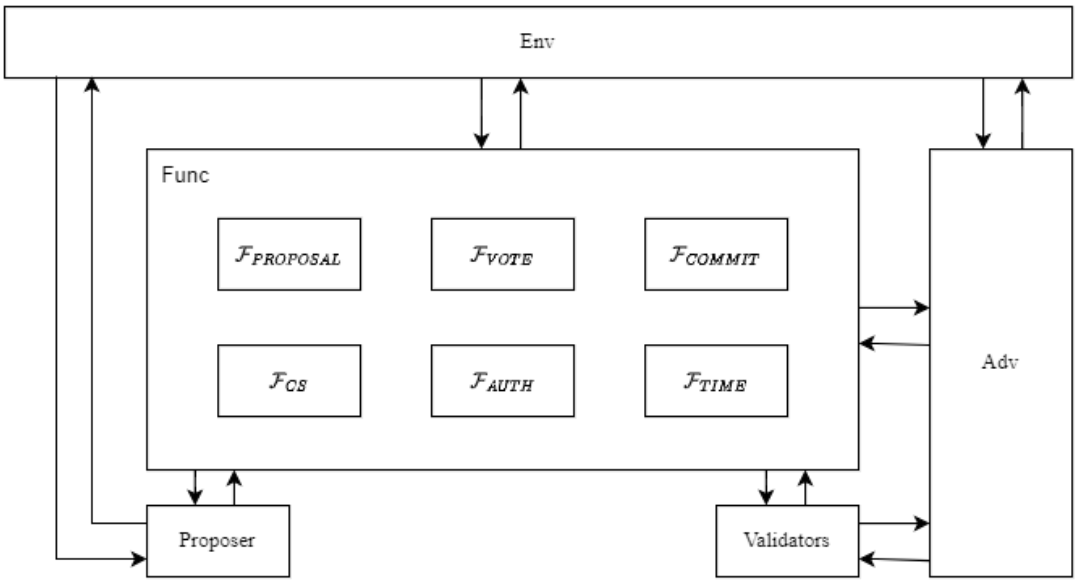
TBFT 建模进度

摘要

本文档介绍了 Tendermint-BFT（TBFT）共识机制的初步框架、核心功能以及协议描述。本文首先展示了协议框架，然后完善了各功能的描述步骤，加入了对于超时机制的描述。

一、 初步框架



二、功能描述

（一）功能$\mathcal{F}_{AUTH}$

初始化：变量M $\coloneqq \perp$和D $\coloneqq 0$。

- 当从$v_i \in V$接收到输入m时，执行以下操作：

  设置D $\coloneqq 1$和M $\coloneqq m$。将L(M)发送给敌手$\mathcal{A}$。将M广播给v_j，$(v_j \in V, j \neq i)$

- 当接收到来自敌手$\mathcal{A}$的$(delay,\ T)$时，执行以下操作：

  如果T是一个以一元表示法编码的自然数，则设置D $\coloneqq D + T$。否则忽略该消息。

- 当从敌手$\mathcal{A}$接收到消息$(corrupt,\ v_i,\ m',\ T')$时，执行以下操作：

  如果D > 0且T'是有效的延迟值，则设置D $\coloneqq T'$并将M $\coloneqq m'$。否则忽略该消息。

（二）功能$\mathcal{F}_{PROPOSAL}$

初始化：设置Proposal $\coloneqq \perp$和Round $\coloneqq 0$。

- 当收到消息$(startProposal)$时，
  - 通过 Round-robin 规则选定提议者Proposer $\in V$：

    -初始化 Validator 的votingPower为其质押资金：

    $$\text{votingPower}_i = \text{stake}_i, \quad \forall i \in \{1, \ldots, \text{N}\}$$

    -按 Round-robin 规则依次选举Proposer，更新Round $\coloneqq Round + 1$。
  - 更新votingPower：

    -未被选中的 Validator 更新为：

    $$\text{votingPower}_i \leftarrow v\text{otingPower}_i + \text{stake}_i$$

-被选中为Proposer的 Validator 更新为：

$$\text{votingPower}_i \leftarrow v\text{otingPower}_i - \sum_{j \neq i} stake_j$$

- （超时处理）：当从$\mathcal{A}$接收到$(timeout, T)$消息时，如果T有效，增加Round，并选择新的提议者。

（三）功能$\mathcal{F}_{VOTE}$

初始化：设置Prevotes $:= \perp$和Precommits $:= \perp$。向$\mathcal{F}_{TIME}$发送$(timeStart, \delta)$命令。如果在任何阶段从$\mathcal{F}_{TIME}$收到$(timeOver)$消息，直接投票给nil块。

- 当从验证者$\mathrm{v_i} \in$ V传入$(Prevote,\ B)$消息时，
  - 当收到Proposal时，向$\mathcal{F}_{CS}$发送$(\mathrm{v_i},\ queryState)$，获取PoLC：
    -若锁定上一轮Proposal，则广播$(\mathrm{v_i},\ prevote,\ B')$。
    -否则广播当前轮Proposal，即广播$(\mathrm{v_i},\ prevote,\ B)$。
  - 若未收到Proposal：
    -若未收到任何Proposal，则广播$(\mathrm{v_i},\ prevote,\ \perp)$。
- 当从验证者$\mathrm{v_i} \in V$传入$(Precommit,\ B)$消息时，持续接收网络中的prevote投票，
  - 若收到超过 $\frac{2}{3}$f的prevote投票：

    签名并广播 $(\mathrm{v_i},\ precommit,\ B)$，向$\mathcal{F}_{CS}$发送$(v,\ unlock,\ B')$释放之前锁定的区块，然后向$\mathcal{F}_{CS}$发送$(\mathrm{v_i},\ lock,\ B)$锁定该区块。
  - 若收到超过 $\frac{2}{3}$f 的空prevote投票：

    向$\mathcal{F}_{CS}$发送$(\mathrm{v_i},\ unlock,\ ALL)$释放所有锁定的区块
  - 否则，不锁定任何区块。

（四）功能$\mathcal{F}_{COMMIT}$

初始化:对每个验证者$v_i \in V$，设置$c_i \in C$，$c_i := 0$表示验证者是否已提交区块。向$\mathcal{F}_{TIME}$发送$(timeStart, \delta)$命令。如果在任何阶段从$\mathcal{F}_{TIME}$收到$(timeOver)$消息，直接提交nil块。

- 当从验证者$v_i \in V$传入$(Commit, B)$消息时，
  - 持续接收网络中的$precommit$投票，判断是否可以进入$Commit$阶段：

    -若收到超过 $\frac{2}{3}f$的$precommit$投票，进入$Commit$阶段。

    -否则，进入下一轮$Proposal$阶段。
  - $Commit$阶段并行步骤：
    -为区块B广播$commit$投票 $(v, commit, B)$。
    -为区块B收集全网的$commit$投票。
  - $Commit$阶段结束：

    -若v_i已为区块B广播$commit$投票且收集到超过$\frac{2}{3}f$的$commit$投票，则设置$c_i :=$ 1，设置$commitTime$为当前时间，向$\mathcal{F}_{CS}$发送(newHeight)。

    -若v_i没有为区块B收集到超过$\frac{2}{3}f$的$commit$投票，向$\mathcal{F}_{CS}$发送$(newRound)$。

    -否则$Commit$过程仍未完成。
  - 提前进入$Commit$阶段：

    若在任何阶段收到超过$\frac{2}{3}f$的$commit$投票，立即进入$Commit$阶段。

- 收到来自任意方v_k的消息$(request_status)$时：
  返回集合C并指示区块B是否已完成。

（五）功能$\mathcal{F}_{CS}$

初始化：设置Height $\coloneqq 0$，$Round \coloneqq 0$和$PoLC \coloneqq \perp$。

- 当从任意验证者$v_i \in V$接收到$(newHeight)$消息时，
  更新Height $\coloneqq$ Height $+ 1$并将$Round$重置为0。
- 当从任意验证者$v_i \in V$接收到$(newRound)$消息时，
  将$Round$重置为0，向$\mathcal{F}_{PROPOSAL}$发送$(startProposal)$消息
- 当从v_i接收到$(v_i, lock, B)$消息时，将v_i加入到$PoLC$中 $(Height, Round, B)$对应的$ValidatorSet$中。
- 当从v_i接收到$(v_i, unlock, B)$消息时，将v_i在对应的$PoLC$中 $(Height, Round, B)$的$ValidatorSet$中删除。
- 当从v_i接收到$(v_i, unlock, ALL)$消息时，则设置$PoLC \coloneqq \perp$。
- 当从v_i接收到$(v_i, queryState)$消息时，返回$PoLC$。

（六）功能$\mathcal{F}_{TIME}$

初始化：设置$t_i \in T$，$t_i \coloneqq \perp$。

- 当从任意验证者$v_i \in V$接收到$(timeStart, \delta)$请求时，将t_i更新为 $t_i \leftarrow \delta$，向验证者v_i返回一个$(timeOK)$消息, 然后开始倒计时。
- 当从任意验证者$v_i \in V$接收到$(GetTime)$请求时，它会将当前的t_i返回给请求方。
- 当从某一个$t_i \in T$, $t_i = 0$时，它会向对应的验证者v_i发送一个$(timeOver)$消息。
- 当从任意验证者$v_i \in V$接收到$(ResetTime)$请求时，将t_i重置为 $t_i \coloneqq \perp$，向验证者v_i返回一个$(timeOK)$消息。

三、协议描述

Tendermint-BFT 协议通过轮次机制和投票阶段确保多个验证者之间就区块达成一致，并最终提交区块。该协议支持容忍少量恶意节点，依赖于消息广播、延迟处理和投票收集来实现共识。

- Party Environment:

  Proposal：调用$\mathcal{F}_{PROPOSAL}$，更新轮次，选择一个提议者 Proposer 并激活。

- Party Proposer:

  Input: 在每一轮开始时，Proposer 首先向$\mathcal{F}_{TIME}$发送$(timeStart, \delta)$命令，然后从功能$\mathcal{F}_{CS}$中接收提案请求，并从中选择一个区块B作为提议区块。如果从$\mathcal{F}_{TIME}$收到$(timeOver)$消息，直接跳转执行 **RoundOK** 部分。

  BroadcastProposal: 当 Proposer 选择了一个区块B并确定该区块有效时，将提议信息$(Proposal, B)$发送给对手$\mathcal{A}$，并广播给所有验证者。

  RoundOK: Proposer 将调用$\mathcal{F}_{PROPOSAL}$更新轮次，并重新选择一个新的提议者，等待下一个提议轮次。

- Party Validator:

  Input: 在收到来自 Proposer 的提议消息$(Proposal,\ B)$后，Validator 验证区块B的有效性，并准备参与投票。

  Prevote: 根据收到的$Proposal$区块B，调用$\mathcal{F}_{VOTE}(Prevote,\ B)$。

  Precommit: 根据收到的$Proposal$区块B，调用$\mathcal{F}_{VOTE}(Precommit,\ B)$。

  Commit: 根据收到的$Proposal$区块B，调用$\mathcal{F}_{COMMIT}(Commit,\ B)$。

  RoundOK: Validator 将等待$\mathcal{F}_{PROPOSAL}$中的提议轮次更新，开始新的轮次。