$$
\text{PRTCL}\ \frac{
\begin{array}{c}
\eta := \mathsf{bnonce}~(\mathsf{bhbody}~bhb)\\[1ex]
\eta \vdash \begin{pmatrix}\eta_v\\ \eta_c\end{pmatrix} \xrightarrow[\text{UPDN}]{slot} \begin{pmatrix}\eta_v'\\ \eta_c'\end{pmatrix}\\[1ex]
\begin{array}{c} d\\ pd\\ dms\\ \eta_0\end{array} \vdash cs \xrightarrow[\text{OVERLAY}]{bh} cs'
\end{array}
}{
\begin{array}{c} d\\ pd\\ dms\\ \eta_0\end{array} \vdash \begin{pmatrix} cs\\ \eta_v\\ \eta_c\end{pmatrix} \xrightarrow[\text{PRTCL}]{bh} \begin{pmatrix} \boldsymbol{cs'}\\ \boldsymbol{\eta_v'}\\ \boldsymbol{\eta_c'}\end{pmatrix}
} \tag{41}
$$

Figure 70: Protocol rules

BBody environments

$$\mathsf{BBodyEnv} = \left(\begin{array}{rlr} \mathit{pp} & \in \mathsf{PParams} & \text{protocol parameters} \\ \mathit{acnt} & \in \mathsf{Acnt} & \text{accounting state} \end{array}\right)$$

BBody states

$$\mathsf{BBodyState} = \left(\begin{array}{rlr} \mathit{ls} & \in \mathsf{LState} & \text{ledger state} \\ b & \in \mathsf{BlocksMade} & \text{blocks made} \end{array}\right)$$

BBody Transitions

$$\_ \vdash \_ \xrightarrow[\mathsf{\scriptsize BBODY}]{\_} \_ \subseteq \mathbb{P}\left(\mathsf{BBodyEnv} \times \mathsf{BBodyState} \times \mathsf{Block} \times \mathsf{BBodyState}\right)$$

BBody helper function

$$\mathsf{incrBlocks} \in \mathsf{Bool} \to \mathsf{KeyHash}_{pool} \to \mathsf{BlocksMade} \to \mathsf{BlocksMade}$$

$$\mathsf{incrBlocks}\ \mathit{isOverlay}\ \mathit{hk}\ b = \begin{cases} b & \text{if } \mathit{isOverlay} \\ b \cup \{\mathit{hk} \mapsto 1\} & \text{if } \mathit{hk} \notin \operatorname{dom} b \\ b \mathbin{\underrightarrow{\cup}} \{\mathit{hk} \mapsto n+1\} & \text{if } \mathit{hk} \mapsto n \in b \end{cases}$$

Figure 71: BBody transition-system types

12.12 Block Body Transition

The Block Body Transition updates the block body state which comprises the ledger state and the map describing the produced blocks. The environment of the BBODY transition are the protocol parameters and the accounting state. The environments and states are defined in Figure 71, along with a helper function incrBlocks, which counts the number of non-overlay blocks produced by each stake pool.

The BBODY transition rule is shown in Figure 72, its sub-rule is LEDGERS which does the update of the ledger state. The signal is a block from which we extract:

- The sequence of transactions *txs* of the block.
- The block header body *bhb*.
- The verification key *vk* of the issuer of the *block* and its hash *hk*.

The transition is executed if the following preconditions are met:

- The size of the block body matches the value given in the block header body.
- The hash of the block body matches the value given in the block header body.
- The LEDGERS transition succeeds.

After this, the transition system updates the mapping of the hashed stake pool keys to the incremented value of produced blocks ($n + 1$), provided the current slot is not an overlay slot.

The BBODY rule has two predicate failures:

- if the size of the block body in the header is not equal to the real size of the block body, there is a *WrongBlockBodySize* failure.
- if the hash of the block body is not also the hash of transactions, there is an *InvalidBodyHash* failure.

$$
\text{Block-Body}\;\frac{
\begin{array}{c}
txs := \mathsf{bbody}\ block \qquad bhb := \mathsf{bhbody}\ (\mathsf{bheader}\ block) \qquad hk := \mathsf{hashKey}\ (\mathsf{bvkcold}\ bhb) \\[1ex]
\mathsf{bBodySize}\ txs = \mathsf{hBbsize}\ bhb \qquad\qquad \mathsf{bbodyhash}\ txs = \mathsf{bhash}\ bhb \\[1ex]
slot := \mathsf{bslot}\ bhb \qquad\qquad fSlot := \mathsf{firstSlot}\ (\mathsf{epoch}\ slot) \\[1ex]
\begin{array}{c} \mathsf{bslot}\ bhb \\ pp \\ acnt \end{array} \vdash ls \xrightarrow[\mathsf{LEDGERS}]{txs} ls'
\end{array}
}{
\begin{array}{c} pp \\ acnt \end{array} \vdash \left(\begin{array}{c} ls \\ b \end{array}\right) \xrightarrow[\mathsf{BBODY}]{block} \left(\begin{array}{c} ls' \\ \mathbf{incrBlocks}\ (\mathbf{isOverlaySlot}\ fSlot\ (\mathbf{d}\ pp)\ slot)\ hk\ b \end{array}\right)
}
\tag{42}
$$

Figure 72: BBody rules

12.13 Chain Transition

The CHAIN transition rule is the main rule of the blockchain layer part of the STS. It calls BHEAD, PRTCL, and BBODY as sub-rules.

The chain rule has no environment.

The transition checks six things (via chainChecks and prtlSeqChecks from Figure 74):

- The slot in the block header body is larger than the last slot recorded.
- The block number increases by exactly one.
- The previous hash listed in the block header matches the previous block header hash which was recorded.
- The size of *bh* is less than or equal to the maximal size that the protocol parameters allow for block headers.
- The size of the block body, as claimed by the block header, is less than or equal to the maximal size that the protocol parameters allow for block bodies. It will later be verified that the size of the block body matches the size claimed in the header (see Figure 72).
- The node is not obsolete, meaning that the major component of the protocol version in the protocol parameters is not bigger than the constant MaxMajorPV.

The chain state is shown in Figure 73, it consists of the following:

- The epoch specific state *nes*.
- The operational certificate issue number map *cs*.
- The epoch nonce η_0.
- The evolving nonce η_v.
- The candidate nonce η_c.
- The previous epoch hash nonce η_h.
- The last header hash h.
- The last slot s_ℓ.
- The last block number b_ℓ.

The CHAIN transition rule is shown in Figure 75. Its signal is a *block*. The transition uses a few helper functions defined in Figure 74.

The CHAIN rule has six predicate failures:

- If the slot of the block header body is not larger than the last slot or greater than the current slot, there is a *WrongSlotInterval* failure.
- If the block number does not increase by exactly one, there is a *WrongBlockNo* failure.
- If the hash of the previous header of the block header body is not equal to the hash given in the environment, there is a *WrongBlockSequence* failure.
- If the size of the block header is larger than the maximally allowed size, there is a *HeaderSizeTooLarge* failure.

Chain states

$$\mathsf{LastAppliedBlock} = \left(\begin{array}{rlr} b_\ell & \in \mathsf{Slot} & \text{last block number} \\ s_\ell & \in \mathsf{Slot} & \text{last slot} \\ h & \in \mathsf{HashHeader} & \text{latest header hash} \end{array}\right)$$

$$\mathsf{ChainState} = \left(\begin{array}{rlr} \mathit{nes} & \in \mathsf{NewEpochState} & \text{epoch specific state} \\ \mathit{cs} & \in \mathsf{KeyHash}_{pool} \mapsto \mathbb{N} & \text{operational certificate issue numbers} \\ \eta_0 & \in \mathsf{Seed} & \text{epoch nonce} \\ \eta_v & \in \mathsf{Seed} & \text{evolving nonce} \\ \eta_c & \in \mathsf{Seed} & \text{candidate nonce} \\ \eta_h & \in \mathsf{Seed} & \text{seed generated from hash of previous epoch's last block header} \\ \mathit{lab} & \in \mathsf{LastAppliedBlock}^? & \text{latest applied block} \end{array}\right)$$

Chain Transitions

$$\vdash \_ \xrightarrow[\text{CHAIN}]{\_} \_ \subseteq \mathbb{P}\,(\mathsf{ChainState} \times \mathsf{Block} \times \mathsf{ChainState})$$

Figure 73: Chain transition-system types

- If the size of the block body is larger than the maximally allowed size, there is a *BlockSizeTooLarge* failure.
- If the major component of the protocol version is larger than MaxMajorPV, there is a *ObsoleteNode* failure.