MODULE *AdaptiveBFT*

EXTENDS *Naturals*, *Integers*, *FiniteSets*, *Sequences*, *TLC*,
AdaptiveBFT_Types, *AVC_RVS*, *APS_Scheduler*, *APS_DecoupledPipeline*, *Mempool*

CONSTANTS

F,
Node,
Tx,
ValidTx,
HotTx,
WarmTx,
Quorum,
MaxView,
MaxBatchSize,
MaxPipelineDepth,
MaxTimeout,
MaxTxPerBlock,
MaxReputation,
RepThreshold,
AgingThreshold,
MaxAge,
DecayNumerator,
DecayDenominator

ASSUME

$\land\ F \in Nat$
$\land\ F \geq 1$
$\land\ Node \neq \{\}$
$\land\ Cardinality(Node) = (3 * F) + 1$
$\land\ Tx \neq \{\}$
$\land\ ValidTx \subseteq Tx$
$\land\ HotTx \subseteq Tx$
$\land\ WarmTx \subseteq Tx$
$\land\ Quorum = (2 * F) + 1$
$\land\ MaxView \in Nat$
$\land\ MaxView \geq 1$
$\land\ MaxBatchSize \in Nat$
$\land\ MaxBatchSize \geq 1$
$\land\ MaxPipelineDepth \in Nat$
$\land\ MaxPipelineDepth \geq 1$
$\land\ MaxTimeout \in Nat$
$\land\ MaxTimeout \geq 2$
$\land\ MaxTxPerBlock \in Nat$
$\land\ MaxTxPerBlock \geq 1$
$\land\ MaxReputation\ \in Nat$

$\land\ MaxReputation\ \geq 2$
$\land\ RepThreshold \in 0\ ..\ MaxReputation$
$\land\ AgingThreshold \in Nat$
$\land\ MaxAge \in Nat$
$\land\ MaxAge \geq AgingThreshold$
$\land\ DecayDenominator \in Nat \setminus \{0\}$
$\land\ DecayNumerator \in 0\ ..\ DecayDenominator$

VARIABLE st

$vars\ \triangleq\ \langle st \rangle$

$Min2(a,\ b)\ \triangleq\ \text{IF}\ a \leq b\ \text{THEN}\ a\ \text{ELSE}\ b$

$NPMessageHasValidSortition(np,\ reputation)\ \triangleq$
  $RVSVerifyPrimary($
    $reputation,$
    $RepThreshold,$
    $np.view,$
    $np.leader,$
    $np.ticket,$
    $np.strikes,$
    $np.proof$
  $)$

$ConfigType\ \triangleq\ APSConfigType(MaxBatchSize,\ MaxPipelineDepth,\ MaxTimeout)$

$QCType\ \triangleq\ \{[view \mapsto v] : v \in -1\ ..\ MaxView\}$

$TeProposalType\ \triangleq\ \{$
  $[$
    $type \mapsto \text{“TeProposal”},$
    $view \mapsto v,$
    $alist \mapsto a,$
    $qc \mapsto qc,$
    $parentView \mapsto p,$
    $from \mapsto n$
  $]:$
    $v \in 0\ ..\ MaxView,$
    $a \in \text{SUBSET}\ Tx,$
    $qc \in QCType,$
    $p \in -1\ ..\ MaxView,$
    $n \in Node$
$\}$

$FullProposalType\ \triangleq\ \{$
  $[$

$$
\begin{aligned}
&\quad\quad type \mapsto \text{“Full”},\\
&\quad\quad view \mapsto v,\\
&\quad\quad txs \mapsto t,\\
&\quad\quad qc \mapsto qc,\\
&\quad\quad parentView \mapsto p,\\
&\quad\quad from \mapsto n\\
&\quad] :\\
&\quad\quad v \in 0 \mathinner{..} MaxView,\\
&\quad\quad t \in \text{SUBSET } Tx,\\
&\quad\quad qc \in QCType,\\
&\quad\quad p \in -1 \mathinner{..} MaxView,\\
&\quad\quad n \in Node\\
\}&
\end{aligned}
$$

$$
\begin{aligned}
&VProposalType \triangleq \{\\
&\quad [\\
&\quad\quad type \mapsto \text{“VProposal”},\\
&\quad\quad view \mapsto v,\\
&\quad\quad rv \mapsto rv,\\
&\quad\quad qc \mapsto qc,\\
&\quad\quad parentView \mapsto p,\\
&\quad\quad from \mapsto n\\
&\quad] :\\
&\quad\quad v \in 0 \mathinner{..} MaxView,\\
&\quad\quad rv \in [Node \to 0 \mathinner{..} MaxReputation],\\
&\quad\quad qc \in QCType,\\
&\quad\quad p \in -1 \mathinner{..} MaxView,\\
&\quad\quad n \in Node\\
\}&
\end{aligned}
$$

$$
\begin{aligned}
&NPMessageType \triangleq \{\\
&\quad [\\
&\quad\quad type \mapsto \text{“NPMessage”},\\
&\quad\quad view \mapsto v,\\
&\quad\quad leader \mapsto l,\\
&\quad\quad ticket \mapsto ticket,\\
&\quad\quad strikes \mapsto strikes,\\
&\quad\quad proof \mapsto VRFProof(l,\, v,\, ticket,\, strikes,\, k),\\
&\quad\quad qc \mapsto qc,\\
&\quad\quad from \mapsto n\\
&\quad] :\\
&\quad\quad v \in 0 \mathinner{..} MaxView,\\
&\quad\quad l \in Node,\\
&\quad\quad ticket \in 0 \mathinner{..} (TicketBound(MaxReputation) - 1),\\
&\quad\quad strikes \in 0 \mathinner{..} Cardinality(Node),
\end{aligned}
$$

$k \in 1 \,..\, Cardinality(Node),$
$qc \in QCType,$
$n \in Node$
}

$SynMessageType \triangleq \{$
[
$type \mapsto$ "SynMessage",
$view \mapsto v,$
$leader \mapsto l,$
$rv \mapsto rv,$
$qc \mapsto qc,$
$from \mapsto n$
] :
$v \in 0 \,..\, MaxView,$
$l \in Node,$
$rv \in [Node \to 0 \,..\, MaxReputation],$
$qc \in QCType,$
$n \in Node$
}

$BlockType \triangleq \{$
[
$view \mapsto v,$
$txs \mapsto t,$
$parentView \mapsto p,$
$proposer \mapsto n$
] :
$v \in 0 \,..\, MaxView,$
$t \in \text{SUBSET } Tx,$
$p \in -1 \,..\, MaxView,$
$n \in Node$
}

$NoTeProposal \triangleq$
[
$type \mapsto$ "NoTeProposal",
$view \mapsto 0,$
$alist \mapsto \{\},$
$qc \mapsto NilQC,$
$parentView \mapsto -1,$
$from \mapsto \text{CHOOSE } n \in Node : \text{TRUE}$
]

$NoFullProposal \triangleq$
[

```
        type ↦ "NoFullProposal",
        view ↦ 0,
        txs ↦ {},
        qc ↦ NilQC,
        parentView ↦ −1,
        from ↦ CHOOSE n ∈ Node : TRUE
    ]

NoVProposal ≜
    [
        type ↦ "NoVProposal",
        view ↦ 0,
        rv ↦ [n ∈ Node ↦ 0],
        qc ↦ NilQC,
        parentView ↦ −1,
        from ↦ CHOOSE n ∈ Node : TRUE
    ]

NoNPMessage ≜
    [
        type ↦ "NoNPMessage",
        view ↦ 0,
        leader ↦ CHOOSE n ∈ Node : TRUE,
        ticket ↦ 0,
        strikes ↦ 0,
        proof ↦ [
            node ↦ CHOOSE n ∈ Node : TRUE,
            view ↦ 0,
            ticket ↦ 0,
            strikes ↦ 0,
            kappa ↦ 1
        ],
        qc ↦ NilQC,
        from ↦ CHOOSE n ∈ Node : TRUE
    ]

NoSynMessage ≜
    [
        type ↦ "NoSynMessage",
        view ↦ 0,
        leader ↦ CHOOSE n ∈ Node : TRUE,
        rv ↦ [n ∈ Node ↦ 0],
        qc ↦ NilQC,
        from ↦ CHOOSE n ∈ Node : TRUE
    ]
```

$$DefaultConfig \triangleq [batchSize \mapsto 1,\ pipelineDepth \mapsto 1,\ timeout \mapsto 1]$$

$$
\begin{array}{l}
Init \triangleq \\
\quad \text{LET } p0 \triangleq \text{CHOOSE } n \in Node : \text{TRUE} \\
\qquad rep0 \triangleq [n \in Node \mapsto MaxReputation \div 2] \\
\quad \text{IN} \\
\quad \land st = [\\
\qquad\qquad view \mapsto 0, \\
\qquad\qquad phase \mapsto \text{“CollectMinor”}, \\
\qquad\qquad primary \mapsto p0, \\
\qquad\qquad highQC \mapsto NilQC, \\
\qquad\qquad lockedQC \mapsto NilQC, \\
\qquad\qquad teProposal \mapsto NoTeProposal, \\
\qquad\qquad fullProposal \mapsto NoFullProposal, \\
\qquad\qquad vProposal \mapsto NoVProposal, \\
\qquad\qquad npMessage \mapsto NoNPMessage, \\
\qquad\qquad synMessage \mapsto NoSynMessage, \\
\qquad\qquad prepareVotes \mapsto \{\}, \\
\qquad\qquad precommitVotes \mapsto \{\}, \\
\qquad\qquad commitVotes \mapsto \{\}, \\
\qquad\qquad timeoutVotes \mapsto \{\}, \\
\qquad\qquad candidateReplicas \mapsto Node, \\
\qquad\qquad tentativePrimary \mapsto p0, \\
\qquad\qquad npConfirms \mapsto \{\}, \\
\qquad\qquad synAcks \mapsto \{\}, \\
\qquad\qquad rawMempool \mapsto \{\}, \\
\qquad\qquad validatedMempool \mapsto \{\}, \\
\qquad\qquad disseminationBuffer \mapsto \{\}, \\
\qquad\qquad orderingBuffer \mapsto \{\}, \\
\qquad\qquad executionBuffer \mapsto \{\}, \\
\qquad\qquad orderedPool \mapsto \{\}, \\
\qquad\qquad computeQueued \mapsto \{\}, \\
\qquad\qquad computeReady \mapsto \{\}, \\
\qquad\qquad txAge \mapsto [tx \in Tx \mapsto 0], \\
\qquad\qquad reputation \mapsto rep0, \\
\qquad\qquad chain \mapsto \langle\rangle, \\
\qquad\qquad localChain \mapsto [n \in Node \mapsto \langle\rangle], \\
\qquad\qquad decidedByView \mapsto [v \in 0 \,..\, MaxView \mapsto \{\}], \\
\qquad\qquad activeConfig \mapsto DefaultConfig, \\
\qquad\qquad pendingConfig \mapsto DefaultConfig, \\
\qquad\qquad schedulerState \mapsto \text{“Monitor”}, \\
\qquad\qquad networkCondition \mapsto \text{“Stable”}, \\
\qquad\qquad stageTimer \mapsto 0, \\
\qquad\qquad inFlight \mapsto 0 \\
\qquad]
\end{array}
$$

$$
\begin{aligned}
&\mathit{TypeOK} \triangleq \\
&\quad \land \quad \mathit{st.view} \in 0 \,..\, \mathit{MaxView} \\
&\quad \land \quad \mathit{st.phase} \in \mathit{ConsensusPhase} \\
&\quad \land \quad \mathit{st.primary} \in \mathit{Node} \\
&\quad \land \quad \mathit{st.highQC} \in \mathit{QCType} \\
&\quad \land \quad \mathit{st.lockedQC} \in \mathit{QCType} \\
&\quad \land \quad (\mathit{st.teProposal} = \mathit{NoTeProposal} \lor \mathit{st.teProposal} \in \mathit{TeProposalType}) \\
&\quad \land \quad (\mathit{st.fullProposal} = \mathit{NoFullProposal} \\
&\qquad\quad \lor \mathit{st.fullProposal} \in \mathit{FullProposalType}) \\
&\quad \land \quad (\mathit{st.vProposal} = \mathit{NoVProposal} \lor \mathit{st.vProposal} \in \mathit{VProposalType}) \\
&\quad \land \quad (\mathit{st.npMessage} = \mathit{NoNPMessage} \lor \mathit{st.npMessage} \in \mathit{NPMessageType}) \\
&\quad \land \quad (\mathit{st.synMessage} = \mathit{NoSynMessage} \lor \mathit{st.synMessage} \in \mathit{SynMessageType}) \\
&\quad \land \quad \mathit{st.prepareVotes} \subseteq \mathit{Node} \\
&\quad \land \quad \mathit{st.precommitVotes} \subseteq \mathit{Node} \\
&\quad \land \quad \mathit{st.commitVotes} \subseteq \mathit{Node} \\
&\quad \land \quad \mathit{st.timeoutVotes} \subseteq \mathit{Node} \\
&\quad \land \quad \mathit{st.candidateReplicas} \subseteq \mathit{Node} \\
&\quad \land \quad \mathit{st.candidateReplicas} \neq \{\} \\
&\quad \land \quad \mathit{st.tentativePrimary} \in \mathit{Node} \\
&\quad \land \quad \mathit{st.npConfirms} \subseteq \mathit{Node} \\
&\quad \land \quad \mathit{st.synAcks} \subseteq \mathit{Node} \\
&\quad \land \quad \mathit{st.rawMempool} \subseteq \mathit{Tx} \\
&\quad \land \quad \mathit{st.validatedMempool} \subseteq \mathit{Tx} \\
&\quad \land \quad \mathit{st.disseminationBuffer} \subseteq \mathit{Tx} \\
&\quad \land \quad \mathit{st.orderingBuffer} \subseteq \mathit{Tx} \\
&\quad \land \quad \mathit{st.executionBuffer} \subseteq \mathit{Tx} \\
&\quad \land \quad \mathit{st.orderedPool} \subseteq \mathit{Tx} \\
&\quad \land \quad \mathit{st.computeQueued} \subseteq \mathit{Tx} \\
&\quad \land \quad \mathit{st.computeReady} \subseteq \mathit{Tx} \\
&\quad \land \quad \mathit{st.txAge} \in [\mathit{Tx} \to 0 \,..\, \mathit{MaxAge}] \\
&\quad \land \quad \mathit{st.reputation} \in [\mathit{Node} \to 0 \,..\, \mathit{MaxReputation}] \\
&\quad \land \quad \forall\, i \in 1 \,..\, \mathit{Len}(\mathit{st.chain}) : \mathit{st.chain}[i] \in \mathit{BlockType} \\
&\quad \land \quad \forall\, n \in \mathit{Node} : \\
&\qquad\quad \forall\, i \in 1 \,..\, \mathit{Len}(\mathit{st.localChain}[n]) : \mathit{st.localChain}[n][i] \in \mathit{BlockType} \\
&\quad \land \quad \mathit{st.decidedByView} \in [0 \,..\, \mathit{MaxView} \to \textsc{subset}\ (\textsc{subset}\ \mathit{Tx})] \\
&\quad \land \quad \mathit{st.activeConfig} \in \mathit{ConfigType} \\
&\quad \land \quad \mathit{st.pendingConfig} \in \mathit{ConfigType} \\
&\quad \land \quad \mathit{st.schedulerState} \in \mathit{SchedulerStateType} \\
&\quad \land \quad \mathit{st.networkCondition} \in \mathit{NetworkConditionType} \\
&\quad \land \quad \mathit{st.stageTimer} \in 0 \,..\, \mathit{MaxTimeout} \\
&\quad \land \quad \mathit{st.inFlight} \in 0 \,..\, \mathit{MaxPipelineDepth}
\end{aligned}
$$

Scalable typing profile used by larger-node *TLC* sanity runs.
This avoids extensional membership in very large constructor sets
(*e.g.*, *NPMessageType*) while preserving field-level type discipline.

$$
\begin{array}{l}
TypeOKLite \triangleq \\
\quad \land\ st.view \in 0\,..\,MaxView \\
\quad \land\ st.phase \in ConsensusPhase \\
\quad \land\ st.primary \in Node \\
\quad \land\ st.highQC.view \in -1\,..\,MaxView \\
\quad \land\ st.lockedQC.view \in -1\,..\,MaxView \\
\quad \land\ st.teProposal.type \in \{\text{“NoTeProposal”},\ \text{“TeProposal”}\} \\
\quad \land\ st.teProposal.view \in 0\,..\,MaxView \\
\quad \land\ st.teProposal.alist \subseteq Tx \\
\quad \land\ st.teProposal.qc.view \in -1\,..\,MaxView \\
\quad \land\ st.teProposal.parentView \in -1\,..\,MaxView \\
\quad \land\ st.teProposal.from \in Node \\
\quad \land\ st.fullProposal.type \in \{\text{“NoFullProposal”},\ \text{“Full”}\} \\
\quad \land\ st.fullProposal.view \in 0\,..\,MaxView \\
\quad \land\ st.fullProposal.txs \subseteq Tx \\
\quad \land\ st.fullProposal.qc.view \in -1\,..\,MaxView \\
\quad \land\ st.fullProposal.parentView \in -1\,..\,MaxView \\
\quad \land\ st.fullProposal.from \in Node \\
\quad \land\ st.vProposal.type \in \{\text{“NoVProposal”},\ \text{“VProposal”}\} \\
\quad \land\ st.vProposal.view \in 0\,..\,MaxView \\
\quad \land\ st.vProposal.qc.view \in -1\,..\,MaxView \\
\quad \land\ st.vProposal.parentView \in -1\,..\,MaxView \\
\quad \land\ st.vProposal.from \in Node \\
\quad \land\ \text{DOMAIN}\ st.vProposal.rv = Node \\
\quad \land\ \forall\, n \in Node : st.vProposal.rv[n] \in 0\,..\,MaxReputation \\
\quad \land\ st.npMessage.type \in \{\text{“NoNPMessage”},\ \text{“NPMessage”}\} \\
\quad \land\ st.npMessage.view \in 0\,..\,MaxView \\
\quad \land\ st.npMessage.leader \in Node \\
\quad \land\ st.npMessage.ticket \in 0\,..\,(TicketBound(MaxReputation) - 1) \\
\quad \land\ st.npMessage.strikes \in 0\,..\,Cardinality(Node) \\
\quad \land\ st.npMessage.qc.view \in -1\,..\,MaxView \\
\quad \land\ st.npMessage.from \in Node \\
\quad \land\ st.synMessage.type \in \{\text{“NoSynMessage”},\ \text{“SynMessage”}\} \\
\quad \land\ st.synMessage.view \in 0\,..\,MaxView \\
\quad \land\ st.synMessage.leader \in Node \\
\quad \land\ st.synMessage.qc.view \in -1\,..\,MaxView \\
\quad \land\ st.synMessage.from \in Node \\
\quad \land\ \text{DOMAIN}\ st.synMessage.rv = Node \\
\quad \land\ \forall\, n \in Node : st.synMessage.rv[n] \in 0\,..\,MaxReputation \\
\quad \land\ st.prepareVotes \subseteq Node \\
\quad \land\ st.precommitVotes \subseteq Node \\
\quad \land\ st.commitVotes \subseteq Node \\
\quad \land\ st.timeoutVotes \subseteq Node \\
\quad \land\ st.npConfirms \subseteq Node \\
\quad \land\ st.synAcks \subseteq Node
\end{array}
$$

$\land\ st.candidateReplicas \subseteq Node$
$\land\ st.candidateReplicas \neq \{\}$
$\land\ st.tentativePrimary \in Node$
$\land\ st.rawMempool \subseteq Tx$
$\land\ st.validatedMempool \subseteq Tx$
$\land\ st.disseminationBuffer \subseteq Tx$
$\land\ st.orderingBuffer \subseteq Tx$
$\land\ st.executionBuffer \subseteq Tx$
$\land\ st.orderedPool \subseteq Tx$
$\land\ st.computeQueued \subseteq Tx$
$\land\ st.computeReady \subseteq Tx$
$\land\ \text{DOMAIN}\ st.txAge = Tx$
$\land\ \forall\, tx \in Tx : st.txAge[tx] \in 0\,..\,MaxAge$
$\land\ \text{DOMAIN}\ st.reputation = Node$
$\land\ \forall\, n \in Node : st.reputation[n] \in 0\,..\,MaxReputation$
$\land\ \forall\, i \in 1\,..\,Len(st.chain) :$
$\quad\land\ st.chain[i].view \in 0\,..\,MaxView$
$\quad\land\ st.chain[i].txs \subseteq Tx$
$\quad\land\ st.chain[i].parentView \in -1\,..\,MaxView$
$\quad\land\ st.chain[i].proposer \in Node$
$\land\ \text{DOMAIN}\ st.localChain = Node$
$\land\ \forall\, n \in Node :$
$\quad\forall\, i \in 1\,..\,Len(st.localChain[n]) :$
$\qquad\land\ st.localChain[n][i].view \in 0\,..\,MaxView$
$\qquad\land\ st.localChain[n][i].txs \subseteq Tx$
$\qquad\land\ st.localChain[n][i].parentView \in -1\,..\,MaxView$
$\qquad\land\ st.localChain[n][i].proposer \in Node$
$\land\ \text{DOMAIN}\ st.decidedByView = 0\,..\,MaxView$
$\land\ \forall\, v \in 0\,..\,MaxView :$
$\quad\forall\, s \in st.decidedByView[v] : s \subseteq Tx$
$\land\ st.activeConfig.timeout \in 1\,..\,MaxTimeout$
$\land\ st.activeConfig.batchSize \in 1\,..\,MaxBatchSize$
$\land\ st.activeConfig.pipelineDepth \in 1\,..\,MaxPipelineDepth$
$\land\ st.pendingConfig.timeout \in 1\,..\,MaxTimeout$
$\land\ st.pendingConfig.batchSize \in 1\,..\,MaxBatchSize$
$\land\ st.pendingConfig.pipelineDepth \in 1\,..\,MaxPipelineDepth$
$\land\ st.schedulerState \in SchedulerStateType$
$\land\ st.networkCondition \in NetworkConditionType$
$\land\ st.stageTimer \in 0\,..\,MaxTimeout$
$\land\ st.inFlight \in 0\,..\,MaxPipelineDepth$

$InjectTx(tx) \triangleq$
$\land\ tx \in Tx$
$\land\ tx \notin st.rawMempool \cup st.validatedMempool$
$\land\ st' = [st\ \text{EXCEPT}\ !.rawMempool = @ \cup \{tx\}]$

$PreValidate(tx) \triangleq$
$\land\ tx \in st.rawMempool$
$\land\ st' =$
IF $tx \in ValidTx$
THEN [
st EXCEPT
$!.rawMempool = @ \setminus \{tx\}$,
$!.validatedMempool = @ \cup \{tx\}$,
$!.txAge = [@ \text{ EXCEPT } ![tx] = 0]$
]
ELSE [
st EXCEPT
$!.rawMempool = @ \setminus \{tx\}$,
$!.txAge = [@ \text{ EXCEPT } ![tx] = 0]$
]

$AgeTx(tx) \triangleq$
$\land\ tx \in st.validatedMempool$
$\land\ st' = [st \text{ EXCEPT } !.txAge = AgeMapBump(@,\ tx,\ MaxAge)]$

$DispatchToDissemination \triangleq$
$\land\ st.phase = \text{“CollectMinor”}$
$\land\ st.orderedPool = \{\}$
$\land\ st.validatedMempool \neq \{\}$
$\land \exists\, moved \in \text{SUBSET } (st.validatedMempool \setminus st.disseminationBuffer) :$
$\land\ moved \neq \{\}$
$\land\ Cardinality(moved) \leq st.activeConfig.batchSize$
$\land\ st' =$
$[st \text{ EXCEPT}$
$!.disseminationBuffer = @ \cup moved]$

$PromoteToOrdering \triangleq$
$\land\ st.phase = \text{“CollectMinor”}$
$\land\ st.orderedPool = \{\}$
$\land\ st.disseminationBuffer \neq \{\}$
$\land \exists\, moved \in \text{SUBSET } (st.disseminationBuffer \setminus st.orderingBuffer) :$
$\land\ moved \neq \{\}$
$\land\ Cardinality(moved) \leq st.activeConfig.batchSize$
$\land\ st' =$
$[st \text{ EXCEPT}$
$!.orderingBuffer = @ \cup moved,$
$!.computeQueued = @ \cup moved,$
$!.computeReady = @ \cup moved]$

$PromoteToExecution \triangleq$
$\land\ st.phase = \text{“CollectMinor”}$

```
    ∧ st.orderedPool = {}
    ∧ st.computeReady ≠ {}
    ∧ ∃ moved ∈ SUBSET ((st.orderingBuffer \ st.executionBuffer) ∩ st.computeReady) :
        ∧ moved ≠ {}
        ∧ Cardinality(moved) ≤ st.activeConfig.batchSize
        ∧ st' =
            [st EXCEPT
               !.executionBuffer = @ ∪ moved]

PreOrder ≜
    ∧ st.phase = "CollectMinor"
    ∧ st.orderedPool = {}
    ∧ st.validatedMempool ≠ {}
    ∧ LET limit ≜ Min2(st.activeConfig.batchSize, MaxTxPerBlock)
          source ≜
           IF st.executionBuffer ≠ {}
            THEN st.executionBuffer
            ELSE
               IF st.computeReady ≠ {}
                THEN st.computeReady
                ELSE st.validatedMempool
          front ≜
           PriorityFront(
               source,
               st.txAge,
               HotTx,
               WarmTx,
               AgingThreshold
           )
          chosen ≜
           SelectBatch(
               front,
               st.txAge,
               HotTx,
               WarmTx,
               AgingThreshold,
               limit
           )
      IN
       ∧ chosen ≠ {}
       ∧ st' =
           [st EXCEPT
              !.orderedPool = chosen,
              !.disseminationBuffer = @ \ chosen,
              !.orderingBuffer = @ \ chosen,
```

```
                !.executionBuffer = @ \ chosen,
                !.computeQueued = @ \ chosen,
                !.computeReady = @ \ chosen]

GenerateTentativeProposal ≜
    ∧ st.phase = "CollectMinor"
    ∧ st.orderedPool ≠ {}
    ∧ st.inFlight < st.activeConfig.pipelineDepth
    ∧ st.primary ∈ CandidateReplicas(st.reputation, RepThreshold)
    ∧ (st.view < MaxView ∨ st.decidedByView[st.view] = {})
    ∧ st' =
        LET te ≜
              TeProposal(
                  st.view,
                  st.orderedPool,
                  st.highQC,
                  st.highQC.view,
                  st.primary
              )
        IN
        [st EXCEPT
           !.teProposal = te,
           !.fullProposal = NoFullProposal,
           !.phase = "Prepare",
           !.prepareVotes = {},
           !.precommitVotes = {},
           !.commitVotes = {},
           !.timeoutVotes = {},
           !.vProposal  = NoVProposal,
           !.npMessage = NoNPMessage,
           !.synMessage = NoSynMessage,
           !.npConfirms = {},
           !.synAcks = {},
           !.stageTimer = 0,
           !.inFlight = @ + 1]

RecoverFullProposal ≜
    ∧ st.phase = "Prepare"
    ∧ st.teProposal ≠ NoTeProposal
    ∧ LET recovered ≜
              RecoverTransactions(
                  st.teProposal.alist,
                  st.validatedMempool
              )
          full ≜
```

```
            FullMessage(
                st.view,
                recovered,
                st.teProposal.qc,
                st.teProposal.parentView,
                st.primary
            )
    IN
     ∧ recovered ≠ {}
     ∧ st' = [st EXCEPT !.fullProposal = full]

PrepareVote(n) ≜
    ∧ st.phase = "Prepare"
    ∧ st.fullProposal ≠ NoFullProposal
    ∧ n ∈ Node \ st.prepareVotes
    ∧ st' = [st EXCEPT !.prepareVotes = @ ∪ {n}]

PrepareQC ≜
    ∧ st.phase = "Prepare"
    ∧ Cardinality(st.prepareVotes) ≥ Quorum
    ∧ st' =
       [st EXCEPT
          !.highQC = QC(st.view),
          !.phase = "PreCommit",
          !.precommitVotes = {},
          !.stageTimer = 0]

PreCommitVote(n) ≜
    ∧ st.phase = "PreCommit"
    ∧ n ∈ Node \ st.precommitVotes
    ∧ st' = [st EXCEPT !.precommitVotes = @ ∪ {n}]

PreCommitQC ≜
    ∧ st.phase = "PreCommit"
    ∧ Cardinality(st.precommitVotes) ≥ Quorum
    ∧ st' =
       [st EXCEPT
          !.lockedQC = st.highQC,
          !.phase = "Commit",
          !.commitVotes = {},
          !.stageTimer = 0]

CommitVote(n) ≜
    ∧ st.phase = "Commit"
    ∧ n ∈ Node \ st.commitVotes
    ∧ st' = [st EXCEPT !.commitVotes = @ ∪ {n}]
```

```
DecideBlock ≜
    ∧ st.phase = "Commit"
    ∧ st.fullProposal ≠ NoFullProposal
    ∧ Cardinality(st.commitVotes) ≥ Quorum
    ∧ LET block ≜
            Block(
                st.view,
                st.fullProposal.txs,
                st.fullProposal.parentView,
                st.primary
            )
          rep1 ≜
            DecayUpdate(
                st.reputation,
                st.primary,
                TRUE,
                MaxReputation,
                DecayNumerator,
                DecayDenominator
            )
          nextView ≜ IF st.view < MaxView THEN st.view + 1 ELSE st.view
          nextPrimary ≜ RVSSelectPrimary(rep1, RepThreshold, nextView)
      IN
       ∧ st' = [st EXCEPT
                  !.chain = Append(@, block),
                  !.localChain =
                      [n ∈ Node ↦ Append(st.localChain[n], block)],
                  !.decidedByView =
                      [@ EXCEPT ![st.view] = @ ∪ {st.fullProposal.txs}],
                  !.reputation = rep1,
                  !.view = nextView,
                  !.primary = nextPrimary,
                  !.phase = "CollectMinor",
                  !.highQC = QC(st.view),
                  !.lockedQC = QC(st.view),
                  !.teProposal = NoTeProposal,
                  !.fullProposal = NoFullProposal,
                  !.vProposal  = NoVProposal,
                  !.npMessage = NoNPMessage,
                  !.synMessage = NoSynMessage,
                  !.prepareVotes = {},
                  !.precommitVotes = {},
                  !.commitVotes = {},
                  !.timeoutVotes = {},
                  !.candidateReplicas =
```

$$
\begin{array}{l}
\quad\quad CandidateReplicas(rep1,\ RepThreshold),\\
\quad !.tentativePrimary = nextPrimary,\\
\quad !.npConfirms = \{\},\\
\quad !.synAcks = \{\},\\
\quad !.rawMempool = @ \setminus st.fullProposal.txs,\\
\quad !.validatedMempool = @ \setminus st.fullProposal.txs,\\
\quad !.disseminationBuffer = @ \setminus st.fullProposal.txs,\\
\quad !.orderingBuffer = @ \setminus st.fullProposal.txs,\\
\quad !.executionBuffer = @ \setminus st.fullProposal.txs,\\
\quad !.orderedPool = \{\},\\
\quad !.computeQueued = @ \setminus st.fullProposal.txs,\\
\quad !.computeReady = @ \setminus st.fullProposal.txs,\\
\quad !.txAge = ResetAges(@,\ st.fullProposal.txs),\\
\quad !.stageTimer = 0,\\
\quad !.inFlight = \text{IF } @ > 0 \text{ THEN } @ - 1 \text{ ELSE } 0]
\end{array}
$$

$$
\begin{array}{l}
Tick \triangleq\\
\quad \land\ st.phase \in \{\text{“Prepare”},\ \text{“PreCommit”},\ \text{“Commit”}\}\\
\quad \land\ st.stageTimer < MaxTimeout\\
\quad \land\ st' = [st \text{ EXCEPT } !.stageTimer = @ + 1]
\end{array}
$$

$$
\begin{array}{l}
CastTimeoutVote(n) \triangleq\\
\quad \land\ st.phase \in \{\text{“Prepare”},\ \text{“PreCommit”},\ \text{“Commit”}\}\\
\quad \land\ st.stageTimer \geq st.activeConfig.timeout\\
\quad \land\ n \in Node \setminus st.timeoutVotes\\
\quad \land\ st' = [st \text{ EXCEPT } !.timeoutVotes = @ \cup \{n\}]
\end{array}
$$

$$
\begin{array}{l}
StartViewChange \triangleq\\
\quad \land\ st.phase \in \{\text{“Prepare”},\ \text{“PreCommit”},\ \text{“Commit”}\}\\
\quad \land\ Cardinality(st.timeoutVotes) \geq Quorum\\
\quad \land \text{LET } rep1 \triangleq\\
\qquad\quad DecayUpdate(\\
\qquad\qquad st.reputation,\\
\qquad\qquad st.primary,\\
\qquad\qquad \text{FALSE},\\
\qquad\qquad MaxReputation,\\
\qquad\qquad DecayNumerator,\\
\qquad\qquad DecayDenominator\\
\qquad\quad)\\
\qquad\quad cand \triangleq CandidateReplicas(rep1,\ RepThreshold)\\
\qquad\quad leader \triangleq RVSSelectPrimary(rep1,\ RepThreshold,\ st.view)\\
\qquad\quad vp \triangleq\\
\qquad\quad VProposal(\\
\qquad\qquad st.view,\\
\qquad\qquad rep1,\\
\qquad\qquad st.lockedQC,
\end{array}
$$

```
                st.lockedQC.view,
                st.primary
            )
        IN
         ∧ st' = [st EXCEPT
                !.phase = "ViewChange",
                !.reputation = rep1,
                !.candidateReplicas = cand,
                !.tentativePrimary = leader,
                !.vProposal  = vp,
                !.npMessage = NoNPMessage,
                !.synMessage = NoSynMessage,
                !.teProposal  = NoTeProposal,
                !.fullProposal  = NoFullProposal,
                !.prepareVotes = {},
                !.precommitVotes = {},
                !.commitVotes = {},
                !.timeoutVotes = {},
                !.disseminationBuffer = {},
                !.orderingBuffer = {},
                !.executionBuffer = {},
                !.orderedPool = {},
                !.computeQueued = {},
                !.computeReady = {},
                !.npConfirms = {},
                !.synAcks = {},
                !.stageTimer = 0,
                !.inFlight = 0]

BroadcastNP ≜
    ∧ st.phase = "ViewChange"
    ∧ st.vProposal  ≠ NoVProposal
    ∧ st.npMessage = NoNPMessage
    ∧ st.tentativePrimary ∈ st.candidateReplicas
    ∧ st' =
       LET evidence ≜
            RVSPrimaryEvidence(
                st.reputation,
                RepThreshold,
                st.view,
                st.tentativePrimary
            )
            np ≜
            NPMessage(
                st.view,
```

```
                st.tentativePrimary,
                evidence.ticket,
                evidence.strikes,
                evidence.proof,
                st.highQC,
                st.tentativePrimary
            )
        IN
        [st EXCEPT
           !.npMessage = np,
           !.npConfirms = {}]

ConfirmNewPrimary(n) ≜
    ∧ st.phase = "ViewChange"
    ∧ st.npMessage ≠ NoNPMessage
    ∧ NPMessageHasValidSortition(st.npMessage, st.reputation)
    ∧ n ∈ Node \ st.npConfirms
    ∧ st' = [st EXCEPT !.npConfirms = @ ∪ {n}]

BroadcastSyn ≜
    ∧ st.phase = "ViewChange"
    ∧ st.npMessage ≠ NoNPMessage
    ∧ NPMessageHasValidSortition(st.npMessage, st.reputation)
    ∧ Cardinality(st.npConfirms) ≥ Quorum
    ∧ st.synMessage = NoSynMessage
    ∧ st' =
        LET syn ≜
            SynMessage(
                st.view,
                st.npMessage.leader,
                st.reputation,
                st.highQC,
                st.npMessage.leader
            )
        IN
        [st EXCEPT
           !.synMessage = syn,
           !.synAcks = {}]

SendSynAck(n) ≜
    ∧ st.phase = "ViewChange"
    ∧ st.synMessage ≠ NoSynMessage
    ∧ st.synMessage.rv = st.reputation
    ∧ n ∈ Node \ st.synAcks
    ∧ st' = [st EXCEPT !.synAcks = @ ∪ {n}]
```

$CompleteViewChange \triangleq$
$\land\ st.phase = \text{“ViewChange”}$
$\land\ st.npMessage \neq NoNPMessage$
$\land\ st.synMessage \neq NoSynMessage$
$\land\ Cardinality(st.synAcks) \geq Quorum$
$\land$ LET $nextView \triangleq$ IF $st.view < MaxView$ THEN $st.view + 1$ ELSE $st.view$
IN
$\land\ st' = [st$ EXCEPT
$!.view = nextView,$
$!.primary = st.npMessage.leader,$
$!.phase = \text{“CollectMinor”},$
$!.teProposal = NoTeProposal,$
$!.fullProposal = NoFullProposal,$
$!.vProposal = NoVProposal,$
$!.npMessage = NoNPMessage,$
$!.synMessage = NoSynMessage,$
$!.prepareVotes = \{\},$
$!.precommitVotes = \{\},$
$!.commitVotes = \{\},$
$!.timeoutVotes = \{\},$
$!.disseminationBuffer = \{\},$
$!.orderingBuffer = \{\},$
$!.executionBuffer = \{\},$
$!.orderedPool = \{\},$
$!.computeQueued = \{\},$
$!.computeReady = \{\},$
$!.candidateReplicas =$
$CandidateReplicas(st.reputation,\ RepThreshold),$
$!.npConfirms = \{\},$
$!.synAcks = \{\},$
$!.stageTimer = 0]$

$DetectAnomaly \triangleq$
$\land\ st.schedulerState = \text{“Monitor”}$
$\land\ st.phase = \text{“CollectMinor”}$
$\land \exists\, nc \in NetworkConditionType :$
$\land\ st' =$
$[st$ EXCEPT
$!.networkCondition = nc,$
$!.schedulerState = \text{“Sample”}]$

$SampleGrid \triangleq$
$\land\ st.schedulerState = \text{“Sample”}$
$\land\ st.phase = \text{“CollectMinor”}$
$\land \exists\, cfg \in ConfigType :$

```
        ∧ cfg.timeout = st.activeConfig.timeout
        ∧ st' =
            [st EXCEPT
              !.pendingConfig = cfg,
              !.schedulerState = "Estimate"]

EstimateGrid ≜
    ∧ st.schedulerState = "Estimate"
    ∧ st.phase = "CollectMinor"
    ∧ LET tunedTimeout ≜
            RefineTimeout(
                st.pendingConfig.timeout,
                st.networkCondition,
                MaxTimeout
            )
          tunedCfg ≜ [st.pendingConfig EXCEPT !.timeout = tunedTimeout]
      IN
       ∧ tunedCfg ∈ ConfigType
       ∧ st' =
            [st EXCEPT
              !.pendingConfig = tunedCfg,
              !.schedulerState = "Explore"]

ExploreGrid ≜
    ∧ st.schedulerState = "Explore"
    ∧ st.phase = "CollectMinor"
    ∧ ∃ candidate ∈ ConfigType :
        ∧ ConfigSatisfiesNetwork(candidate, st.networkCondition)
        ∧ LET chosen ≜
                ChooseBetterConfig(
                    st.pendingConfig,
                    candidate,
                    st.networkCondition
                )
          IN  st' =
                [st EXCEPT
                  !.pendingConfig = chosen,
                  !.schedulerState = "Deploy"]

DeployConfig ≜
    ∧ st.schedulerState = "Deploy"
    ∧ st.phase = "CollectMinor"
    ∧ st.pendingConfig ∈ ConfigType
    ∧ ConfigSatisfiesNetwork(st.pendingConfig, st.networkCondition)
    ∧ st' =
        [st EXCEPT
```

$!.activeConfig = st.pendingConfig,$
$!.schedulerState = \text{“Monitor”}]$

$InjectTxStep \triangleq \exists\, tx \in Tx : InjectTx(tx)$
$PreValidateStep \triangleq \exists\, tx \in st.rawMempool : PreValidate(tx)$
$AgeTxStep \triangleq \exists\, tx \in st.validatedMempool : AgeTx(tx)$
$PrepareVoteStep \triangleq \exists\, n \in Node : PrepareVote(n)$
$PreCommitVoteStep \triangleq \exists\, n \in Node : PreCommitVote(n)$
$CommitVoteStep \triangleq \exists\, n \in Node : CommitVote(n)$
$CastTimeoutVoteStep \triangleq \exists\, n \in Node : CastTimeoutVote(n)$
$ConfirmNewPrimaryStep \triangleq \exists\, n \in Node : ConfirmNewPrimary(n)$
$SendSynAckStep \triangleq \exists\, n \in Node : SendSynAck(n)$

$ConsensusNext \triangleq$
$\lor\ InjectTxStep$
$\lor\ PreValidateStep$
$\lor\ AgeTxStep$
$\lor\ DispatchToDissemination$
$\lor\ PromoteToOrdering$
$\lor\ PromoteToExecution$
$\lor\ PreOrder$
$\lor\ GenerateTentativeProposal$
$\lor\ RecoverFullProposal$
$\lor\ PrepareVoteStep$
$\lor\ PrepareQC$
$\lor\ PreCommitVoteStep$
$\lor\ PreCommitQC$
$\lor\ CommitVoteStep$
$\lor\ DecideBlock$
$\lor\ Tick$
$\lor\ CastTimeoutVoteStep$
$\lor\ StartViewChange$
$\lor\ BroadcastNP$
$\lor\ ConfirmNewPrimaryStep$
$\lor\ BroadcastSyn$
$\lor\ SendSynAckStep$
$\lor\ CompleteViewChange$

$APSNext \triangleq$
$\lor\ DetectAnomaly$
$\lor\ SampleGrid$
$\lor\ EstimateGrid$
$\lor\ ExploreGrid$
$\lor\ DeployConfig$

$Next \triangleq ConsensusNext \lor APSNext$

$$
\begin{array}{l}
Consistency \triangleq \\
\quad \forall\, n1,\, n2 \in Node : st.localChain[n1] = st.localChain[n2] \\
\\
NoForkPerView \triangleq \\
\quad \forall\, v \in 0\,..\,MaxView : Cardinality(st.decidedByView[v]) \leq 1 \\
\\
PipelineBounded \triangleq st.inFlight \leq st.activeConfig.pipelineDepth \\
\\
QCLocked \triangleq st.lockedQC.view \leq st.highQC.view \\
\\
QCViewSafety \triangleq st.highQC.view \leq st.view \\
\\
MempoolSoundness \triangleq \\
\quad \land\, st.orderedPool \subseteq st.validatedMempool \\
\quad \land\, st.validatedMempool \subseteq Tx \\
\\
ProposalFlowSafety \triangleq \\
\quad \land\, st.phase \in \{\text{“Prepare”},\ \text{“PreCommit”},\ \text{“Commit”}\} \\
\qquad \Rightarrow st.teProposal \neq NoTeProposal \\
\quad \land\, st.phase \in \{\text{“PreCommit”},\ \text{“Commit”}\} \\
\qquad \Rightarrow st.fullProposal \neq NoFullProposal \\
\quad \land\, st.teProposal \neq NoTeProposal \Rightarrow st.teProposal.alist \neq \{\} \\
\quad \land\, st.fullProposal \neq NoFullProposal \Rightarrow st.fullProposal.txs \subseteq Tx \\
\\
ViewChangeMessageSafety \triangleq \\
\quad \land\, st.phase \neq \text{“ViewChange”} \Rightarrow \\
\qquad \land\, st.vProposal\ = NoVProposal \\
\qquad \land\, st.npMessage = NoNPMessage \\
\qquad \land\, st.synMessage = NoSynMessage \\
\quad \land\, st.phase = \text{“ViewChange”} \Rightarrow \\
\qquad \land\, st.vProposal \neq NoVProposal \\
\qquad \land\, st.vProposal.view = st.view \\
\qquad \land\, st.vProposal.rv = st.reputation \\
\qquad \land\, st.vProposal.qc.view \leq st.highQC.view \\
\quad \land\, st.npMessage \neq NoNPMessage \Rightarrow \\
\qquad \land\, st.npMessage.view = st.view \\
\qquad \land\, st.npMessage.leader \in st.candidateReplicas \\
\qquad \land\, NPMessageHasValidSortition(st.npMessage,\ st.reputation) \\
\quad \land\, st.synMessage \neq NoSynMessage \Rightarrow \\
\qquad \land\, st.synMessage.view = st.view \\
\qquad \land\, st.synMessage.leader = st.tentativePrimary \\
\qquad \land\, st.synMessage.rv = st.reputation \\
\\
ChainParentSafety \triangleq \\
\quad \land\, \forall\, i \in 1\,..\,Len(st.chain) : \\
\qquad \land\, st.chain[i].parentView \in\ -1\,..\,MaxView \\
\qquad \land\, st.chain[i].parentView \leq st.chain[i].view
\end{array}
$$

```
        ∧ (st.chain[i].parentView < st.chain[i].view
            ∨ st.chain[i].view = MaxView)
    ∧ ∀ i ∈ 2 .. Len(st.chain) :
        ∧ st.chain[i − 1].view < st.chain[i].view
        ∧ st.chain[i].parentView ≥ st.chain[i − 1].view

PrimaryEligibilitySafety ≜
    ∧ st.phase = "ViewChange" ⇒
        ∧ st.tentativePrimary ∈ st.candidateReplicas
        ∧ (st.npMessage = NoNPMessage
            ∨ ∧ st.npMessage.leader ∈ st.candidateReplicas
              ∧ NPMessageHasValidSortition(st.npMessage, st.reputation))
    ∧ st.phase ≠ "ViewChange" ⇒
        ∨ st.primary ∈ CandidateReplicas(st.reputation, RepThreshold)
        ∨ ∧ st.phase ∈ {"Prepare", "PreCommit", "Commit"}
          ∧ (st.stageTimer ≥ st.activeConfig.timeout
            ∨ st.timeoutVotes ≠ {})

ReconfigurationSafety ≜
    ∧ st.activeConfig ∈ ConfigType
    ∧ st.pendingConfig ∈ ConfigType
    ∧ st.schedulerState ≠ "Monitor"
        ∨ ConfigSatisfiesNetwork(
            st.activeConfig,
            st.networkCondition
        )

DecoupledPipelineSafety ≜
    LET inv ≜
            DecouplingInvariant(
                st.disseminationBuffer,
                st.orderingBuffer,
                st.executionBuffer,
                st.computeQueued,
                st.computeReady,
                Tx
            )
        flow ≜
            DecouplingFlowSafety(
                st.validatedMempool,
                st.orderedPool,
                st.fullProposal.txs,
                st.disseminationBuffer,
                st.orderingBuffer,
                st.executionBuffer,
                st.computeQueued,
```

$st.computeReady$

)

IN

$\wedge\ inv$

$\wedge\ flow$

$EventuallyOneCommit \triangleq \Diamond(Len(st.chain) \geq 1)$

$EventuallyTwoCommits \triangleq \Diamond(Len(st.chain) \geq 2)$

$InfiniteCollectMinor \triangleq \Box\Diamond(st.phase = \text{“CollectMinor”})$

$EventuallyViewProgress \triangleq \Diamond(st.view \geq 1)$