```
──────────────────── MODULE PaxosStore ────────────────────
Specification of the consensus protocol in PaxosStore.

See [PaxosStore@VLDB2017](https://www.vldb.org/pvldb/vol10/p1730-lin.pdf) by Tencent.
EXTENDS Integers, FiniteSets
───────────────────────────────────────────────────────────
Max(m, n) ≜ IF m > n THEN m ELSE n
Injective(f) ≜ ∀ a, b ∈ DOMAIN f : (a ≠ b) ⇒ (f[a] ≠ f[b])
───────────────────────────────────────────────────────────
CONSTANTS
    Participant,    the set of partipants
    Value           the set of possible input values for Participant to propose

None ≜ CHOOSE b : b ∉ Value
NP ≜ Cardinality(Participant)  number of p ∈  Participants

Quorum ≜ {Q ∈ SUBSET Participant : Cardinality(Q) ∗ 2 = NP + 1}
ASSUME QuorumAssumption ≜
    ∧  ∀ Q ∈ Quorum : Q ⊆ Participant
    ∧  ∀ Q1, Q2 ∈ Quorum : Q1 ∩ Q2 ≠ {}

Ballot ≜ Nat

PIndex ≜ CHOOSE f ∈ [Participant → 1 . . NP] : Injective(f)   TODO: (1) symmetry set? (2) model
Bals(p) ≜ {b ∈ Ballot : b%NP = PIndex[p] − 1}  allocate ballots for each p ∈ Participant
───────────────────────────────────────────────────────────
State ≜ [maxBal  : Ballot ∪ { − 1},
         maxVBal : Ballot ∪ { − 1}, maxVVal : Value ∪ {None}]

InitState ≜ [maxBal ↦ −1, maxVBal ↦ −1, maxVVal ↦ None]
For simplicity, in this specification, we choose to send the complete state of a participant each
time. When receiving such a message, the participant processes only the "partial" state it needs.
Message ≜ [type  : {"Prepare", "Accept", "ACK"},
           from : Participant, to : SUBSET Participant,   TODO: remove "to"
           state  : [Participant → State]]
───────────────────────────────────────────────────────────
VARIABLES
    state,   state[p][q]: the state of q ∈ Participant from the view of p ∈ Participant
    msgs     the set of messages that have been sent

vars ≜ ⟨state, msgs⟩

TypeOK ≜
    ∧  state ∈ [Participant → [Participant → State]]
    ∧  msgs ⊆ Message

Send(m) ≜ msgs′ = msgs ∪ {m}
───────────────────────────────────────────────────────────
```

```
$Init \triangleq$
    $\land\ state = [p \in Participant \mapsto [q \in Participant \mapsto InitState]]$
    $\land\ msgs = \{\}$
```

$p \in Participant$ starts the prepare phase by issuing a ballot $b \in Ballot$.

```
$Prepare(p, b) \triangleq$
    $\land\ state[p][p].maxBal < b$
    $\land\ b \in Bals(p)$
    $\land\ state' = [state \text{ EXCEPT } ![p][p].maxBal = b]$
    $\land\ Send([type \mapsto \text{“Prepare”}, from \mapsto p, to \mapsto Participant, state \mapsto state'[p]])$
```

$q \in Participant$ updates its own state $state[q]$ according to the actual state pp of $p \in Participant$ extracted from a message $m \in Message$ it receives. This is called by $OnMessage(q)$.

Note: pp is $m.state[p]$; it may not be equal to $state[p][p]$ at the time $UpdateState$ is called.

```
$UpdateState(q, p, pp) \triangleq$
    $state' = [state \text{ EXCEPT}$
                $![q][p].maxBal = Max(@, pp.maxBal),$
                $![q][p].maxVBal = Max(@, pp.maxVBal),$
                $![q][p].maxVVal = \text{IF } state[q][p].maxVBal < pp.maxVBal$
                                        $\text{THEN } pp.maxVVal \text{ ELSE } @,$
                $![q][q].maxBal = Max(@, pp.maxBal),$
                $![q][q].maxVBal = \text{IF } state[q][q].maxBal \leq pp.maxVBal$
                                        $\text{THEN } pp.maxVBal \text{ ELSE } @,$    make promise
                $![q][q].maxVVal = \text{IF } state[q][q].maxBal \leq pp.maxVBal$    $TODO$: write-once?
                                        $\text{THEN } pp.maxVVal \text{ ELSE } @]$    accept
```

$q \in Participant$ receives and processes a message in $Message$.

```
$OnMessage(q) \triangleq$
    $\exists\, m \in msgs :$
        $\land\ m.type = \text{“ACK”} \Rightarrow m.to = \{q\}$
        $\land\ \text{LET } p \triangleq m.from$
          $\text{IN }\ UpdateState(q, p, m.state[p])$
        $\land\ \text{IF } \lor\ m.state[q].maxBal < state'[q][q].maxBal$    $TODO$: delete “if”?
               $\lor\ m.state[q].maxVBal < state'[q][q].maxVBal$
          $\text{THEN } Send([type \mapsto \text{“ACK”}, from \mapsto q, to \mapsto \{m.from\}, state \mapsto state'[q]])$
          $\text{ELSE UNCHANGED } msgs$
```

$p \in Participant$ starts the accept phase by issuing the ballot $b \in Ballot$ with value $v \in Value$.

```
$Accept(p, b, v) \triangleq$
    $\land\ \neg\exists\, m \in msgs :$    $TODO$: delete it? to allow repeating $Phase2a$?
            $m.type = \text{“Accept”} \land m.state[p].maxBal = b$
    $\land\ b \in Bals(p)$    $TODO$: delete it? to break “client-restricted config”?
    $\land\ \exists\, Q \in Quorum : \forall q \in Q : state[p][q].maxBal = b$
    $\land\ \lor\ \forall q \in Participant : state[p][q].maxVBal = -1$    free to pick its own value
        $\lor\ \exists\, q \in Participant :$    $v$ is the value with the highest $maxVBal$
            $\land\ state[p][q].maxVVal = v$
            $\land\ \forall r \in Participant : state[p][q].maxVBal \geq state[p][r].maxVBal$
```

```
     $\land\ state' = [state\ \text{EXCEPT}\ ![p][p].maxVBal = b,$
                                       $![p][p].maxVVal = v]$
     $\land\ Send([type \mapsto \text{“Accept”},\ from \mapsto p,\ to \mapsto Participant,\ state \mapsto state'[p]])$
--------------------------------------------------------------------------------
$Next \triangleq \exists\, p \in Participant :\ \lor\ OnMessage(p)$
                                  $\lor\ \exists\, b \in Ballot :\ \lor\ Prepare(p,\ b)$
                                                    $\lor\ \exists\, v \in Value : Accept(p,\ b,\ v)$
$Spec \triangleq Init \land \Box[Next]_{vars}$
--------------------------------------------------------------------------------
$ChosenP(p) \triangleq$   the set of values chosen by $p \in Participant$
    $\{v \in Value : \exists\, b \in Ballot :$
                        $\exists\, Q \in Quorum : \forall\, q \in Q :\ \land\ state[p][q].maxVBal = b$
                                                      $\land\ state[p][q].maxVVal = v\}$

$chosen \triangleq \text{UNION}\ \{ChosenP(p) : p \in Participant\}$

$Consistency \triangleq Cardinality(chosen) \leq 1$

THEOREM $Spec \Rightarrow \Box Consistency$
================================================================================
```
\ ∗ Modification History
\ ∗ Last modified *Tue Jul* 30 13:09:11 *CST* 2019 by *hengxin*
\ ∗ Last modified *Mon Jul* 22 13:59:15 *CST* 2019 by *pure_*
\ ∗ Last modified *Mon Jun* 03 21:26:09 *CST* 2019 by *stary*
\ ∗ Last modified *Wed* May 09 21:39:31 *CST* 2018 by dell
\ ∗ Created *Mon Apr* 23 15:47:52 *GMT* + 08:00 2018 by *pure_*