```
1  ---------- MODULE EagerVoting ----------
2  EXTENDS Sets
3  ----------------------------------------
4  CONSTANT Value, Acceptor, Quorum

6  ASSUME QuorumAssumption ≜
7      ∧  ∀ Q ∈ Quorum : Q ⊆ Acceptor
8      ∧  ∀ Q1, Q2 ∈ Quorum : Q1 ∩ Q2 ≠ {}

10 THEOREM QuorumNonEmpty ≜ ∀ Q ∈ Quorum : Q ≠ {}
11 BY QuorumAssumption

13 Ballot ≜ Nat
14 ----------------------------------------
15 VARIABLES votes, maxBal

17 TypeOK ≜ ∧ votes ∈ [Acceptor → SUBSET (Ballot × Value)]
18          ∧ maxBal ∈ [Acceptor → Ballot ∪ {−1}]
19 ----------------------------------------
20 VotedFor(a, b, v) ≜ ⟨b, v⟩ ∈ votes[a]

22 DidNotVoteAt(a, b) ≜ ∀ v ∈ Value : ¬ VotedFor(a, b, v)

24 ShowsSafeAt(Q, b, v) ≜
25   ∧ ∀ a ∈ Q : maxBal[a] ≥ b  have promised
26   ∧ ∃ c ∈ −1 .. (b − 1) :
27       ∧ (c ≠ −1) ⇒ ∃ a ∈ Q : VotedFor(a, c, v)
28       ∧ ∀ d ∈ (c + 1) .. (b − 1), a ∈ Q : DidNotVoteAt(a, d)
29 ----------------------------------------
30 Init ≜
31     ∧ votes = [a ∈ Acceptor ↦ {}]
32     ∧ maxBal = [a ∈ Acceptor ↦ −1]

34 IncreaseMaxBal(a, b) ≜
35   ∧ b > maxBal[a]
36   ∧ maxBal′ = [maxBal EXCEPT ![a] = b]  make promise
37   ∧ UNCHANGED votes
```

The only difference between *EagerVoting* and *Voting* is:

In *Voting*, we have $maxBal' = [maxBal \text{ EXCEPT } ![a] = b]$.

```
44 VoteFor(a, b, v) ≜
45     ∧  maxBal[a] ≤ b  keep promise
46     ∧  ∀ vt ∈ votes[a] : vt[1] ≠ b
47     ∧  ∀ c ∈ Acceptor \ {a} :
48          ∀ vt ∈ votes[c] : (vt[1] = b) ⇒ (vt[2] = v)
49     ∧  ∃ Q ∈ Quorum : ShowsSafeAt(Q, b, v)  safe to vote
50     ∧  votes′ = [votes EXCEPT ![a] = votes[a] ∪ {⟨b, v⟩}]  vote
```

```
51      ∧   ∃ c ∈ Ballot :
52              ∧ c ≥ b
53              ∧ maxBal' = [maxBal EXCEPT ![a] = c]  make promise
54  ----------------------------------------------------------------------
55  Next ≜
56      ∃ a ∈ Acceptor, b ∈ Ballot :
57          ∨ IncreaseMaxBal(a, b)
58          ∨ ∃ v ∈ Value : VoteFor(a, b, v)

60  Spec ≜ Init ∧ □[Next]⟨votes, maxBal⟩
61  ----------------------------------------------------------------------
62  ChosenAt(b, v) ≜
63      ∃ Q ∈ Quorum : ∀ a ∈ Q : VotedFor(a, b, v)

65  chosen ≜ {v ∈ Value : ∃ b ∈ Ballot : ChosenAt(b, v)}

67  Consistency ≜ chosen = {} ∨ ∃ v ∈ Value : chosen = {v}  Cardinality(chosen) ≤ 1
68  ----------------------------------------------------------------------
69  CannotVoteAt(a, b) ≜
70      ∧ maxBal[a] > b
71      ∧ DidNotVoteAt(a, b)

73  NoneOtherChoosableAt(b, v) ≜
74      ∃ Q ∈ Quorum :
75          ∀ a ∈ Q : VotedFor(a, b, v) ∨ CannotVoteAt(a, b)

77  SafeAt(b, v) ≜
78      ∀ c ∈ 0 .. (b − 1) : NoneOtherChoosableAt(c, v)

80  VotesSafe ≜
81      ∀ a ∈ Acceptor, b ∈ Ballot, v ∈ Value :
82          VotedFor(a, b, v) ⇒ SafeAt(b, v)

84  OneVote ≜
85      ∀ a ∈ Acceptor, b ∈ Ballot, v, w ∈ Value :
86          VotedFor(a, b, v) ∧ VotedFor(a, b, w) ⇒ (v = w)

88  OneValuePerBallot ≜
89      ∀ a1, a2 ∈ Acceptor, b ∈ Ballot, v1, v2 ∈ Value :
90          VotedFor(a1, b, v1) ∧ VotedFor(a2, b, v2) ⇒ (v1 = v2)

92  Inv ≜ TypeOK ∧ VotesSafe ∧ OneValuePerBallot
93  ----------------------------------------------------------------------
94  THEOREM AllSafeAtZero ≜ ∀ v ∈ Value : SafeAt(0, v)
95    BY DEF SafeAt

97  THEOREM ChoosableThm ≜
98              ∀ b ∈ Ballot, v ∈ Value :
```

```
                ChosenAt(b, v) ⇒ NoneOtherChoosableAt(b, v)
  BY DEF ChosenAt, NoneOtherChoosableAt

THEOREM OneVoteThm ≜ OneValuePerBallot ⇒ OneVote
  BY DEF OneValuePerBallot, OneVote
----------------------------------------------------------------------------
THEOREM VotesSafeImpliesConsistency ≜
  ASSUME VotesSafe, OneVote, chosen ≠ {}
  PROVE  ∃ v ∈ Value : chosen = {v}
⟨1⟩1. PICK v ∈ Value : v ∈ chosen
  BY DEF chosen
⟨1⟩2. SUFFICES ASSUME NEW w ∈ chosen
                PROVE  w = v
  BY ⟨1⟩1, ⟨1⟩2
⟨1⟩3. ASSUME NEW b1 ∈ Ballot, NEW b2 ∈ Ballot, b1 < b2,
             NEW v1 ∈ Value, NEW v2 ∈ Value,
             ChosenAt(b1, v1) ∧ ChosenAt(b2, v2)
      PROVE  v1 = v2
  ⟨2⟩1. SafeAt(b2, v2)
    BY ⟨1⟩3, QuorumAssumption, SMT DEF ChosenAt, VotesSafe
  ⟨2⟩2. QED
    BY ⟨1⟩3, ⟨2⟩1, QuorumAssumption, Z3
    DEFS CannotVoteAt, DidNotVoteAt, OneVote,
         ChosenAt, NoneOtherChoosableAt, Ballot, SafeAt
⟨1⟩4. QED
  BY QuorumAssumption, ⟨1⟩1, ⟨1⟩2, ⟨1⟩3, Z3
  DEFS Ballot, ChosenAt, OneVote, chosen

THEOREM ShowsSafety ≜
            TypeOK ∧ VotesSafe ∧ OneValuePerBallot ⇒
              ∀ Q ∈ Quorum, b ∈ Ballot, v ∈ Value :
                ShowsSafeAt(Q, b, v) ⇒ SafeAt(b, v)
  BY QuorumAssumption, Z3
  DEFS Ballot, TypeOK, VotesSafe, OneValuePerBallot, SafeAt,
    ShowsSafeAt, CannotVoteAt, NoneOtherChoosableAt, DidNotVoteAt

THEOREM SafeAtStable ≜ Inv ∧ Next ∧ TypeOK′ ⇒
                                 ∀ b ∈ Ballot, v ∈ Value :
                                   SafeAt(b, v) ⇒ SafeAt(b, v)′
  OMITTED
----------------------------------------------------------------------------
THEOREM Invariant ≜ Spec ⇒ □Inv
⟨1⟩ USE DEF Inv
⟨1⟩1. Init ⇒ Inv
  BY DEF Init, TypeOK, VotesSafe, OneValuePerBallot, VotedFor
⟨1⟩2. Inv ∧ [Next]⟨votes, maxBal⟩ ⇒ Inv′
```

⟨2⟩ SUFFICES ASSUME Inv, $[Next]_{\langle votes,\, maxBal\rangle}$
PROVE Inv'
OBVIOUS
⟨2⟩1.CASE $Next$
⟨3⟩ SUFFICES ASSUME NEW $a \in Acceptor$, NEW $b \in Ballot$,
$\lor\ IncreaseMaxBal(a,\, b)$
$\lor\ \exists\, v \in Value : VoteFor(a,\, b,\, v)$
PROVE Inv'
BY ⟨2⟩1 DEF $Next$
⟨3⟩1.CASE $IncreaseMaxBal(a,\, b)$
⟨4⟩1. $TypeOK'$
BY ⟨3⟩1 DEF $TypeOK$, $IncreaseMaxBal$
⟨4⟩2. $VotesSafe'$
⟨5⟩ SUFFICES ASSUME NEW $a\_1 \in Acceptor'$, NEW $b\_1 \in Ballot'$, NEW $v \in Value'$
PROVE $VotedFor(a\_1,\, b\_1,\, v)' \Rightarrow SafeAt(b\_1,\, v)'$
BY DEF $VotesSafe$
⟨5⟩1. $\forall\, aa \in Acceptor,\, bb \in Ballot,\, vv \in Value :$
$VotedFor(aa,\, bb,\, vv) \equiv VotedFor(aa,\, bb,\, vv)'$
BY ⟨3⟩1 DEF $IncreaseMaxBal$, $VotedFor$
⟨5⟩2. $\forall\, aa \in Acceptor,\, bb \in Ballot :$
$maxBal[aa] > bb \Rightarrow maxBal'[aa] > bb$
BY ⟨3⟩1 DEF $IncreaseMaxBal$, $TypeOK$, $Ballot$
⟨5⟩3. $\forall\, aa \in Acceptor,\, bb \in Ballot :$
$DidNotVoteAt(aa,\, bb) \Rightarrow DidNotVoteAt(aa,\, bb)'$
BY ⟨3⟩1 DEF $IncreaseMaxBal$, $DidNotVoteAt$, $VotedFor$
⟨5⟩4. $\forall\, aa \in Acceptor,\, bb \in Ballot :$
$CannotVoteAt(aa,\, bb) \Rightarrow CannotVoteAt(aa,\, bb)'$
BY ⟨3⟩1, ⟨5⟩2, ⟨5⟩3 DEF $IncreaseMaxBal$, $CannotVoteAt$
⟨5⟩5. $\forall\, bb \in Ballot,\, vv \in Value :$
$NoneOtherChoosableAt(bb,\, vv) \Rightarrow NoneOtherChoosableAt(bb,\, vv)'$
BY ⟨5⟩1, ⟨5⟩4, $QuorumAssumption$DEFS $NoneOtherChoosableAt$
⟨5⟩6. QED
BY ⟨5⟩1, ⟨5⟩5 DEF $TypeOK$, $Ballot$, $VotesSafe$, $SafeAt$
⟨4⟩3. $OneValuePerBallot'$
BY ⟨3⟩1 DEF $IncreaseMaxBal$, $OneValuePerBallot$, $VotedFor$
⟨4⟩4. QED
BY ⟨4⟩1, ⟨4⟩2, ⟨4⟩3 DEF Inv
⟨3⟩2. ASSUME NEW $v \in Value$,
$VoteFor(a,\, b,\, v)$
PROVE Inv'
⟨4⟩ SUFFICES ASSUME NEW $Q \in Quorum$,
$ShowsSafeAt(Q,\, b,\, v)$
PROVE Inv'
BY ⟨3⟩2 DEF $VoteFor$
⟨4⟩1. $TypeOK'$

```
          BY ⟨3⟩2 DEF TypeOK, VoteFor
        ⟨4⟩2. VotesSafe′ Using OneValuePerBallot in SafeAtStable
          ⟨5⟩ SUFFICES ASSUME NEW aa ∈ Acceptor′, NEW bb ∈ Ballot′, NEW vv ∈ Value′,
                              VotedFor(aa, bb, vv)′
                       PROVE  SafeAt(bb, vv)′
            BY  DEF VotesSafe
          ⟨5⟩1.CASE VotedFor(aa, bb, vv)
            ⟨6⟩1. SafeAt(bb, vv)
              BY ⟨5⟩1 DEF VotesSafe
            ⟨6⟩ QED
              BY ⟨4⟩1, ⟨6⟩1, SafeAtStable DEF Next
          ⟨5⟩2.CASE ¬VotedFor(aa, bb, vv)
            ⟨6⟩1. aa = a ∧ bb = b ∧ vv = v ∧ VotedFor(a, b, v)′
              BY ⟨3⟩2, ⟨4⟩1, ⟨5⟩2 DEF VoteFor, VotedFor, TypeOK
            ⟨6⟩ QED
              BY ⟨4⟩1, ⟨6⟩1, ShowsSafety, SafeAtStable DEF VoteFor, Next
          ⟨5⟩ QED
            BY ⟨5⟩1, ⟨5⟩2
        ⟨4⟩3. OneValuePerBallot′
          BY ⟨3⟩2 DEF VoteFor, OneValuePerBallot, VotedFor, TypeOK
        ⟨4⟩4. QED
          BY ⟨3⟩2, ⟨4⟩1, ⟨4⟩2, ⟨4⟩3 DEF Inv
      ⟨3⟩3. QED
        BY ⟨2⟩1, ⟨3⟩1, ⟨3⟩2
    ⟨2⟩2.CASE UNCHANGED ⟨votes, maxBal⟩
      BY ⟨2⟩2
      DEFS TypeOK, Next, VotesSafe, OneValuePerBallot,
           VotedFor, SafeAt, NoneOtherChoosableAt, CannotVoteAt, DidNotVoteAt,
           IncreaseMaxBal, VoteFor
    ⟨2⟩3. QED
      BY ⟨2⟩1, ⟨2⟩2
  ⟨1⟩3. QED
    BY ⟨1⟩1, ⟨1⟩2, PTL DEF Spec
```

```
THEOREM Consistent ≜ Spec ⇒ □Consistency
⟨1⟩ USE  DEF Ballot
⟨1⟩1. Inv ⇒ Consistency
  ⟨2⟩ SUFFICES ASSUME Inv
               PROVE  Consistency
    OBVIOUS
  ⟨2⟩ QED
    BY VotesSafeImpliesConsistency, OneVoteThm DEF Inv, Consistency
⟨1⟩2. QED
  BY Invariant, ⟨1⟩1, PTL
```

```
C ≜ INSTANCE Consensus WITH chosen ← chosen

THEOREM Refinement ≜ Spec ⇒ C!Spec
⟨1⟩1. Init ⇒ C!Init
  BY QuorumAssumption, SetExtensionality, IsaM("force")
   DEF Init, C!Init, chosen, ChosenAt, VotedFor
⟨1⟩2. TypeOK′ ∧ Consistency′ ∧ [Next]⟨votes, maxBal⟩ ⇒ [C!Next]chosen
  ⟨2⟩1. UNCHANGED ⟨votes, maxBal⟩ ⇒ UNCHANGED chosen
    BY DEF chosen, ChosenAt, VotedFor
  ⟨2⟩2. TypeOK′ ∧ Consistency′ ∧ Next ⇒ C!Next ∨ UNCHANGED chosen
    ⟨3⟩1. SUFFICES ASSUME TypeOK′, Consistency′, Next
                   PROVE  C!Next ∨ UNCHANGED chosen
      OBVIOUS
    ⟨3⟩2. chosen ⊆ chosen′
      BY ⟨3⟩1, QuorumAssumption, Z3
      DEFS Next, IncreaseMaxBal, VoteFor, Inv, TypeOK, chosen, ChosenAt, VotedFor, Ballot
    ⟨3⟩3. chosen′ = {} ∨ ∃ v ∈ Value : chosen′ = {v}
      BY ⟨3⟩1 DEF Consistency
    ⟨3⟩4. QED
      BY ⟨3⟩1, ⟨3⟩2, ⟨3⟩3 DEF C!Next
  ⟨2⟩3. QED
    BY ⟨2⟩1, ⟨2⟩2
⟨1⟩3. QED
  BY ⟨1⟩1, ⟨1⟩2, Invariant, Consistent, PTL DEF Spec, Inv, C!Spec
```