MODULE *PaxosHistVar*

Basic *Paxos* verified using only history variables.

See https://*github.com*/sachand/*HistVar*/blob/master/Basic%20*Paxos*/*PaxosUs.tla*

EXTENDS *Integers*, *TLAPS*, *NaturalsInduction*

CONSTANTS *Acceptors*, *Values*, *Quorums*

ASSUME *QuorumAssumption* $\triangleq$
$\land$ *Quorums* $\subseteq$ SUBSET *Acceptors*
$\land \forall\, Q1,\, Q2 \in \mathit{Quorums} : Q1 \cap Q2 \neq \{\}$

$\mathit{Ballots} \triangleq \mathit{Nat}$

VARIABLES *sent*

$\mathit{vars} \triangleq \langle \mathit{sent} \rangle$

$\mathit{Send}(m) \triangleq \mathit{sent}' = \mathit{sent} \cup \{m\}$

$\mathit{None} \triangleq$ CHOOSE $v : v \notin \mathit{Values}$

$\mathit{Init} \triangleq \mathit{sent} = \{\}$

Phase 1a: A leader selects a ballot number b and sends a 1a message with ballot b to a majority of acceptors. It can do this only if it has not already sent a 1a message for ballot b.

$\mathit{Phase1a}(b) \triangleq \mathit{Send}([\mathit{type} \mapsto \text{“1a”},\ \mathit{bal} \mapsto b])$

Phase 1b: If an acceptor receives a 1a message with ballot b greater than that of any 1a message to which it has already responded, then it responds to the request with a promise not to accept any more proposals for ballots numbered less than b and with the highest-numbered ballot (if any) for which it has voted for a value and the value it voted for in that ballot. That promise is made in a 1b message.

$\mathit{last\_voted}(a) \triangleq$ LET $2bs \triangleq \{m \in \mathit{sent} : m.\mathit{type} = \text{“2b”} \land m.\mathit{acc} = a\}$
IN IF $2bs \neq \{\}$ THEN $\{m \in 2bs : \forall\, m2 \in 2bs : m.\mathit{bal} \geq m2.\mathit{bal}\}$
ELSE $\{[\mathit{bal} \mapsto -1,\ \mathit{val} \mapsto \mathit{None}]\}$

$\mathit{Phase1b}(a) \triangleq$
$\exists\, m \in \mathit{sent},\ r \in \mathit{last\_voted}(a) :$
$\land\ m.\mathit{type} = \text{“1a”}$
$\land \forall\, m2 \in \mathit{sent} : m2.\mathit{type} \in \{\text{“1b”},\ \text{“2b”}\} \land m2.\mathit{acc} = a \Rightarrow m.\mathit{bal} > m2.\mathit{bal}$
$\land\ \mathit{Send}([\mathit{type} \mapsto \text{“1b”},\ \mathit{bal} \mapsto m.\mathit{bal},$
$\mathit{maxVBal} \mapsto r.\mathit{bal},\ \mathit{maxVal} \mapsto r.\mathit{val},\ \mathit{acc} \mapsto a])$

Phase 2a: If the leader receives a response to its 1b message (for ballot b) from a quorum of acceptors, then it sends a 2a message to all acceptors for a proposal in ballot b with a value v, where v is the value of the highest-numbered proposal among the responses, or is any value if the responses reported no proposals. The leader can send only one 2a message for any ballot.

$\mathit{Phase2a}(b) \triangleq$
$\land\ \neg\exists\, m \in \mathit{sent} : (m.\mathit{type} = \text{“2a”}) \land (m.\mathit{bal} = b)$

```
63    ∧ ∃ v ∈ Values, Q ∈ Quorums, S ∈ SUBSET {m ∈ sent : m.type = "1b" ∧ m.bal = b} :
64        ∧ ∀ a ∈ Q : ∃ m ∈ S : m.acc = a
65        ∧ ∨ ∀ m ∈ S : m.maxVBal = −1
66          ∨ ∃ c ∈ 0 .. (b − 1) :
67              ∧ ∀ m ∈ S : m.maxVBal ≤ c
68              ∧ ∃ m ∈ S : ∧ m.maxVBal = c
69                          ∧ m.maxVal = v
70        ∧ Send([type ↦ "2a", bal ↦ b, val ↦ v])
```

Phase 2b: If an acceptor receives a 2a message for a ballot numbered b, it votes for the message's value in ballot b unless it has already responded to a 1a request for a ballot number greater than or equal to b.

```
78  Phase2b(a) ≜
79    ∃ m ∈ sent :
80      ∧ m.type = "2a"
81      ∧ ∀ m2 ∈ sent : m2.type ∈ {"1b", "2b"} ∧ m2.acc = a ⇒ m.bal ≥ m2.bal
82      ∧ Send([type ↦ "2b", bal ↦ m.bal, val ↦ m.val, acc ↦ a])

84  Next ≜ ∨ ∃ b ∈ Ballots : Phase1a(b) ∨ Phase2a(b)
85         ∨ ∃ a ∈ Acceptors : Phase1b(a) ∨ Phase2b(a)

87  Spec ≜ Init ∧ □[Next]_vars
88  ----------------------------------------------------------------------------
```

How a value is chosen:

This spec does not contain any actions in which a value is explicitly chosen (or a chosen value learned). Wnat it means for a value to be chosen is defined by the operator *Chosen*, where *Chosen*(v) means that v has been chosen. From this definition, it is obvious how a process learns that a value has been chosen from messages of type "2b".

```
98  VotedForIn(a, v, b) ≜ ∃ m ∈ sent : ∧ m.type = "2b"
99                                      ∧ m.val  = v
100                                     ∧ m.bal  = b
101                                     ∧ m.acc  = a

103 ChosenIn(v, b) ≜ ∃ Q ∈ Quorums :
104                    ∀ a ∈ Q : VotedForIn(a, v, b)

106 Chosen(v) ≜ ∃ b ∈ Ballots : ChosenIn(v, b)
```

The consistency condition that a consensus algorithm must satisfy is the invariance of the following state predicate *Consistency*.

```
112 Consistency ≜ ∀ v1, v2 ∈ Values : Chosen(v1) ∧ Chosen(v2) ⇒ (v1 = v2)
113 ----------------------------------------------------------------------------
```

This section of the spec defines the invariant *Inv*.

```
117 Messages ≜      [type : {"1a"}, bal : Ballots]
118             ∪   [type : {"1b"}, bal : Ballots, maxVBal : Ballots ∪ {−1},
119                  maxVal : Values ∪ {None}, acc : Acceptors]
120             ∪   [type : {"2a"}, bal : Ballots, val : Values]
```

$$\cup \quad [type : \{\text{“2b”}\},\ bal : Ballots,\ val : Values,\ acc : Acceptors]$$

$$TypeOK \triangleq sent \in \text{SUBSET } Messages$$

$WontVoteIn(a,\ b)$ is a predicate that implies that a has not voted and never will vote in ballot b.

$$\begin{aligned} WontVoteIn(a,\ b) \triangleq\ & \land \forall\, v \in Values : \neg VotedForIn(a,\ v,\ b) \\ & \land \exists\, m \in sent : m.type \in \{\text{“1b”},\ \text{“2b”}\} \land m.acc = a \land m.bal > b \end{aligned}$$

The predicate $SafeAt(v,\ b)$ implies that no value other than perhaps v has been or ever will be chosen in any ballot numbered less than b.

$$\begin{aligned} & SafeAt(v,\ b) \triangleq \\ & \quad \forall\, b2 \in 0\,..\,(b-1) : \\ & \qquad \exists\, Q \in Quorums : \\ & \qquad\quad \forall\, a \in Q : VotedForIn(a,\ v,\ b2) \lor WontVoteIn(a,\ b2) \end{aligned}$$

$$\begin{aligned} & MsgInv \triangleq \\ & \quad \forall\, m \in sent : \\ & \qquad \land\ m.type = \text{“1b”} \Rightarrow\ \land\ VotedForIn(m.acc,\ m.maxVal,\ m.maxVBal) \lor m.maxVBal = -1 \\ & \qquad\qquad\qquad\qquad\ \land \forall\, b \in m.maxVBal + 1\,..\,m.bal - 1 : \neg\exists\, v \in Values : VotedForIn(m.acc,\ v,\ b) \\ & \qquad \land\ m.type = \text{“2a”} \Rightarrow\ \land\ SafeAt(m.val,\ m.bal) \\ & \qquad\qquad\qquad\qquad\ \land \forall\, m2 \in sent : (m2.type = \text{“2a”} \land m2.bal = m.bal) \Rightarrow m2 = m \\ & \qquad \land\ m.type = \text{“2b”} \Rightarrow \exists\, m2 \in sent :\ \land\ m2.type = \text{“2a”} \\ & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\ \land\ m2.bal\ \ = m.bal \\ & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\ \land\ m2.val\ \ = m.val \end{aligned}$$

$$Inv \triangleq TypeOK \land MsgInv$$

The following two lemmas are simple consequences of the definitions.

$$\begin{aligned} & \text{LEMMA } VotedInv \triangleq \\ & \qquad MsgInv \land TypeOK \Rightarrow \\ & \qquad\quad \forall\, a \in Acceptors,\ v \in Values,\ b \in Ballots : \\ & \qquad\qquad VotedForIn(a,\ v,\ b) \Rightarrow SafeAt(v,\ b) \\ & \text{BY DEF } VotedForIn,\ MsgInv,\ Messages,\ TypeOK \end{aligned}$$

$$\begin{aligned} & \text{LEMMA } VotedOnce \triangleq \\ & \qquad MsgInv \Rightarrow\ \forall\, a1,\ a2 \in Acceptors,\ b \in Ballots,\ v1,\ v2 \in Values : \\ & \qquad\qquad VotedForIn(a1,\ v1,\ b) \land VotedForIn(a2,\ v2,\ b) \Rightarrow (v1 = v2) \\ & \text{BY DEF } MsgInv,\ VotedForIn \end{aligned}$$

The following lemma shows that (the invariant implies that) the predicate $SafeAt(v,\ b)$ is stable, meaning that once it becomes true, it remains true throughout the rest of the excecution.

$$\begin{aligned} & \text{LEMMA } SafeAtStable \triangleq Inv \land Next \Rightarrow \\ & \qquad\qquad \forall\, v \in Values,\ b \in Ballots : \\ & \qquad\qquad\quad SafeAt(v,\ b) \Rightarrow SafeAt(v,\ b)' \\ & \langle 1 \rangle \text{ SUFFICES ASSUME } Inv,\ Next, \\ & \qquad\qquad \text{NEW } v \in Values,\ \text{NEW } b \in Ballots,\ SafeAt(v,\ b) \\ & \qquad \text{PROVE } SafeAt(v,\ b)' \end{aligned}$$

OBVIOUS
⟨1⟩ USE DEF $Send$, Inv, $Ballots$
⟨1⟩ USE TRUE ∧ TRUE
⟨1⟩1. ASSUME NEW $bb \in Ballots$, $Phase1a(bb)$
PROVE $SafeAt(v, b)'$
BY ⟨1⟩1, SMT DEF $SafeAt$, $Phase1a$, $VotedForIn$, $WontVoteIn$
⟨1⟩2. ASSUME NEW $a \in Acceptors$, $Phase1b(a)$
PROVE $SafeAt(v, b)'$
BY ⟨1⟩2, $QuorumAssumption$, $SMTT(60)$ DEF $TypeOK$, $SafeAt$, $WontVoteIn$, $VotedForIn$, $Phase1b$
⟨1⟩3. ASSUME NEW $bb \in Ballots$, $Phase2a(bb)$
PROVE $SafeAt(v, b)'$
BY ⟨1⟩3, $QuorumAssumption$, SMT DEF $TypeOK$, $SafeAt$, $WontVoteIn$, $VotedForIn$, $Phase2a$
⟨1⟩4. ASSUME NEW $a \in Acceptors$, $Phase2b(a)$
PROVE $SafeAt(v, b)'$
⟨2⟩1. PICK $m \in sent : Phase2b(a)!(m)$
BY ⟨1⟩4 DEF $Phase2b$
⟨2⟩2 $\forall\, aa \in Acceptors,\ bb \in Ballots,\ vv \in Values :$
$VotedForIn(aa, vv, bb) \Rightarrow VotedForIn(aa, vv, bb)'$
BY ⟨2⟩1 DEF $TypeOK$, $VotedForIn$
⟨2⟩4. ASSUME NEW $a2 \in Acceptors$, NEW $b2 \in Ballots$,
$WontVoteIn(a2, b2)$, NEW $v2 \in Values$
PROVE $\neg VotedForIn(a2, v2, b2)'$
⟨3⟩1. PICK $m1 \in sent : m1.type \in \{\text{“1b”}, \text{“2b”}\} \land m1.acc = a2 \land m1.bal > b2$
BY ⟨2⟩4 DEF $WontVoteIn$
⟨3⟩2. $a2 = a \Rightarrow b2 \neq m.bal$
BY ⟨2⟩1, ⟨2⟩4, ⟨3⟩1, $a2 = a \Rightarrow m.bal \geq m1.bal$ DEF $TypeOK$, $Messages$
⟨3⟩3. $a2 \neq a \Rightarrow \neg VotedForIn(a2, v2, b2)'$
BY ⟨2⟩1, ⟨2⟩4 DEF $WontVoteIn$, $VotedForIn$
⟨3⟩.QED
BY ⟨2⟩1, ⟨2⟩2, ⟨2⟩4, ⟨3⟩2, ⟨3⟩3 DEF $Phase2b$, $VotedForIn$, $WontVoteIn$, $TypeOK$, $Messages$, $Send$
⟨2⟩5 $\forall\, aa \in Acceptors,\ bb \in Ballots : WontVoteIn(aa, bb) \Rightarrow WontVoteIn(aa, bb)'$
BY ⟨2⟩4, ⟨2⟩1 DEF $WontVoteIn$, $Send$
⟨2⟩ QED
BY ⟨2⟩2, ⟨2⟩5, $QuorumAssumption$ DEF $SafeAt$

⟨1⟩5. QED
BY ⟨1⟩1, ⟨1⟩2, ⟨1⟩3, ⟨1⟩4 DEF $Next$

THEOREM $Invariant \triangleq Spec \Rightarrow \Box Inv$
⟨1⟩ USE DEF $Ballots$, $last\_voted$
⟨1⟩1. $Init \Rightarrow Inv$
BY DEF $Init$, Inv, $TypeOK$, $MsgInv$, $VotedForIn$
⟨1⟩2. $Inv \land [Next]_{vars} \Rightarrow Inv'$
⟨2⟩ SUFFICES ASSUME Inv, $Next$
PROVE Inv'

BY DEF $vars$, Inv, $TypeOK$, $MsgInv$, $VotedForIn$, $SafeAt$, $WontVoteIn$
⟨2⟩ USE DEF Inv
⟨2⟩1. $TypeOK'$
⟨3⟩1. ASSUME NEW $b \in Ballots$, $Phase1a(b)$PROVE $TypeOK'$
BY ⟨3⟩1 DEF $TypeOK$, $Phase1a$, $Send$, $Messages$
⟨3⟩2. ASSUME NEW $b \in Ballots$, $Phase2a(b)$PROVE $TypeOK'$
⟨4⟩1. PICK $v \in Values$:
$Send([type \mapsto$ "2a", $bal \mapsto b$, $val \mapsto v])$
BY ⟨3⟩2 DEF $Phase2a$
⟨4⟩.QED
BY ⟨4⟩1 DEF $TypeOK$, $Send$, $Messages$
⟨3⟩3. ASSUME NEW $a \in Acceptors$, $Phase1b(a)$PROVE $TypeOK'$
⟨4⟩.PICK $m \in sent$, $r \in last\_voted(a) : Phase1b(a)!(m, r)$
BY ⟨3⟩3 DEF $Phase1b$
⟨4⟩.QED
BY DEF $Send$, $TypeOK$, $Messages$
⟨3⟩4. ASSUME NEW $a \in Acceptors$, $Phase2b(a)$PROVE $TypeOK'$
⟨4⟩.PICK $m \in sent : Phase2b(a)!(m)$
BY ⟨3⟩4 DEF $Phase2b$
⟨4⟩.QED
BY DEF $Send$, $TypeOK$, $Messages$
⟨3⟩.QED
BY ⟨3⟩1, ⟨3⟩2, ⟨3⟩3, ⟨3⟩4 DEF $Next$
⟨2⟩3. $MsgInv'$
⟨3⟩1. ASSUME NEW $b \in Ballots$, $Phase1a(b)$
PROVE $MsgInv'$
⟨4⟩1. $\forall\, aa, vv, bb : VotedForIn(aa, vv, bb)' \equiv VotedForIn(aa, vv, bb)$
BY ⟨3⟩1 DEF $Send$, $VotedForIn$, $Phase1a$
⟨4⟩ QED
BY ⟨3⟩1, ⟨4⟩1, $QuorumAssumption$, $SafeAtStable$ DEF $Phase1a$, $MsgInv$, $TypeOK$, $Messages$, $Send$
⟨3⟩2. ASSUME NEW $a \in Acceptors$, $Phase1b(a)$
PROVE $MsgInv'$
⟨4⟩.PICK $m \in sent$, $r \in last\_voted(a) : Phase1b(a)!(m, r)$
BY ⟨3⟩2 DEF $Phase1b$
⟨4⟩1. $\forall\, aa, vv, bb : VotedForIn(aa, vv, bb)' \equiv VotedForIn(aa, vv, bb)$
BY DEF $Send$, $VotedForIn$
⟨4⟩.DEFINE $m2 \triangleq [type \mapsto$ "1b", $bal \mapsto m.bal$, $maxVBal \mapsto r.bal$,
$maxVal \mapsto r.val$, $acc \mapsto a]$
⟨4⟩3. $VotedForIn(m2.acc, m2.maxVal, m2.maxVBal) \lor m2.maxVBal = -1$
BY DEF $TypeOK$, $Messages$, $VotedForIn$
⟨4⟩4. $\forall\, b \in (r.bal + 1) \,..\, (m2.bal - 1) :$
$\neg\exists\, v \in Values : VotedForIn(m2.acc, v, b)$
BY DEF $TypeOK$, $Messages$, $VotedForIn$, $Send$
⟨4⟩.QED
BY ⟨4⟩1, ⟨4⟩3, ⟨4⟩4, $SafeAtStable$ DEF $MsgInv$, $TypeOK$, $Messages$, $Send$

⟨3⟩3. ASSUME NEW $b \in Ballots$, $Phase2a(b)$
PROVE $MsgInv'$
⟨4⟩1. $\neg\exists\, m \in sent : (m.type = \text{“2a”}) \land (m.bal = b)$
BY ⟨3⟩3 DEF $Phase2a$
⟨4⟩2. PICK $v \in Values$, $Q \in Quorums$, $S \in$ SUBSET $\{m \in sent : m.type = \text{“1b”} \land m.bal = b\}$:
$\land \forall\, a \in Q : \exists\, m \in S : m.acc = a$
$\land\ \lor \forall\, m \in S : m.maxVBal = -1$
$\lor \exists\, c \in 0\,..\,(b-1) :$
$\land \forall\, m \in S : m.maxVBal \leq c$
$\land \exists\, m \in S :\ \land m.maxVBal = c$
$\land m.maxVal = v$
$\land\ Send([type \mapsto \text{“2a”},\ bal \mapsto b,\ val \mapsto v])$
BY ⟨3⟩3 DEF $Phase2a$
⟨4⟩.DEFINE $mm \triangleq [type \mapsto \text{“2a”},\ bal \mapsto b,\ val \mapsto v]$
⟨4⟩3. $sent' = sent \cup \{mm\}$
BY ⟨4⟩2 DEF $Send$
⟨4⟩4. $\forall\, aa,\ vv,\ bb : VotedForIn(aa,\ vv,\ bb)' \equiv VotedForIn(aa,\ vv,\ bb)$
BY ⟨4⟩3 DEF $VotedForIn$
⟨4⟩6. $\forall\, m,\ ma \in sent' : m.type = \text{“2a”} \land ma.type = \text{“2a”} \land ma.bal = m.bal$
$\Rightarrow ma = m$
BY ⟨4⟩1, ⟨4⟩3, *Isa* DEF $MsgInv$
⟨4⟩10. $SafeAt(v,\ b)$
⟨5⟩1.CASE $\forall\, m \in S : m.maxVBal = -1$
In that case, no acceptor in Q voted in any ballot less than b,
by the last conjunct of $MsgInv$ for type “1b” messages, and that's enough
BY ⟨5⟩1, ⟨4⟩2 DEF $TypeOK$, $MsgInv$, $SafeAt$, $WontVoteIn$
⟨5⟩2. ASSUME NEW $c \in 0\,..\,(b-1)$,
$\forall\, m \in S : m.maxVBal \leq c$,
NEW $ma \in S$, $ma.maxVBal = c$, $ma.maxVal = v$
PROVE $SafeAt(v,\ b)$
⟨6⟩.SUFFICES ASSUME NEW $d \in 0\,..\,(b-1)$
PROVE $\exists\, QQ \in Quorums : \forall\, q \in QQ :$
$VotedForIn(q,\ v,\ d) \lor WontVoteIn(q,\ d)$
BY DEF $SafeAt$
⟨6⟩1.CASE $d \in 0\,..\,(c-1)$
The “1b” message for v with $maxVBal$ value c must have been safe
according to $MsgInv$ for “1b” messages and lemma $VotedInv$,
and that proves the assertion
BY ⟨5⟩2, ⟨6⟩1, $VotedInv$ DEF $SafeAt$, $MsgInv$, $TypeOK$, $Messages$
⟨6⟩2.CASE $d = c$
⟨7⟩1. $VotedForIn(ma.acc,\ v,\ c)$
BY ⟨5⟩2 DEF $MsgInv$
⟨7⟩2. $\forall\, q \in Q,\ w \in Values : VotedForIn(q,\ w,\ c) \Rightarrow w = v$
BY ⟨7⟩1, $VotedOnce$, $QuorumAssumption$ DEF $TypeOK$, $Messages$
⟨7⟩.QED

BY $\langle 6\rangle 2$, $\langle 4\rangle 2$, $\langle 7\rangle 2$ DEF *WontVoteIn*

$\langle 6\rangle 3$.CASE $d \in (c+1) \,..\, (b-1)$

By the last conjunct of *MsgInv* for type "1b" messages, no acceptor in Q voted at any of these ballots.

BY $\langle 6\rangle 3$, $\langle 4\rangle 2$, $\langle 5\rangle 2$ DEF *MsgInv*, *TypeOK*, *Messages*, *WontVoteIn*

$\langle 6\rangle$.QED BY $\langle 6\rangle 1$, $\langle 6\rangle 2$, $\langle 6\rangle 3$

$\langle 5\rangle$.QED BY $\langle 4\rangle 2$, $\langle 5\rangle 1$, $\langle 5\rangle 2$

$\langle 4\rangle 11$. $(\forall\, m2 \in sent : m2.type = \text{"2a"} \Rightarrow SafeAt(m2.val,\, m2.bal))'$

BY $\langle 4\rangle 10$, $\langle 4\rangle 3$, *SafeAtStable* DEF *MsgInv*, *TypeOK*, *Messages*

$\langle 4\rangle$.QED

BY $\langle 4\rangle 3$, $\langle 4\rangle 4$, $\langle 4\rangle 6$, $\langle 4\rangle 11$, $\forall\, m2 \in sent' \setminus sent : m2.type \neq \text{"1b"}$ DEF *MsgInv*, *TypeOK*, *Messages*

$\langle 3\rangle 4$. ASSUME NEW $a \in Acceptors$, $Phase2b(a)$ PROVE $MsgInv'$

$\langle 4\rangle$.PICK $m \in sent : Phase2b(a)!(m)$

BY $\langle 3\rangle 4$ DEF *Phase2b*

$\langle 4\rangle 1$. $\forall\, aa,\, vv,\, bb : VotedForIn(aa,\, vv,\, bb) \Rightarrow VotedForIn(aa,\, vv,\, bb)'$

BY DEF *VotedForIn*, *Send*

$\langle 4\rangle 2$. $\forall\, mm \in sent : mm.type = \text{"1b"}$
$\Rightarrow \forall\, v \in Values,\, c \in (mm.maxVBal + 1) \,..\, (mm.bal - 1) :$
$\neg VotedForIn(mm.acc,\, v,\, c) \Rightarrow \neg VotedForIn(mm.acc,\, v,\, c)'$

BY DEF *Send*, *VotedForIn*, *MsgInv*, *TypeOK*, *Messages*

$\langle 4\rangle$.QED

BY $\langle 4\rangle 1$, $\langle 4\rangle 2$, *SafeAtStable*, $\langle 2\rangle 1$ DEF *MsgInv*, *Send*, *TypeOK*, *Messages*

$\langle 3\rangle 5$. QED

BY $\langle 3\rangle 1$, $\langle 3\rangle 2$, $\langle 3\rangle 3$, $\langle 3\rangle 4$ DEF *Next*

$\langle 2\rangle 4$. QED

BY $\langle 2\rangle 1$, $\langle 2\rangle 3$ DEF *Inv*

$\langle 1\rangle 3$. QED

BY $\langle 1\rangle 1$, $\langle 1\rangle 2$, *PTL* DEF *Spec*

THEOREM $Consistent \triangleq Spec \Rightarrow \Box Consistency$

$\langle 1\rangle$ USE DEF *Ballots*

$\langle 1\rangle 1$. $Inv \Rightarrow Consistency$

$\langle 2\rangle$ SUFFICES ASSUME Inv,
NEW $v1 \in Values$, NEW $v2 \in Values$,
NEW $b1 \in Ballots$, NEW $b2 \in Ballots$,
$ChosenIn(v1,\, b1)$, $ChosenIn(v2,\, b2)$,
$b1 \leq b2$
PROVE $v1 = v2$

BY DEF *Consistency*, *Chosen*

$\langle 2\rangle 1$.CASE $b1 = b2$

BY $\langle 2\rangle 1$, *VotedOnce*, *QuorumAssumption*, $SMTT(100)$ DEF *ChosenIn*, *Inv*

$\langle 2\rangle 2$.CASE $b1 < b2$

$\langle 3\rangle 1.$ $SafeAt(v2,\ b2)$
  BY $VotedInv$, $QuorumAssumption$ DEF $ChosenIn$, Inv
$\langle 3\rangle 2.$ PICK $Q2 \in Quorums$:
  $\forall\, a \in Q2 \quad : VotedForIn(a,\ v2,\ b1) \lor WontVoteIn(a,\ b1)$
  BY $\langle 3\rangle 1$, $\langle 2\rangle 2$ DEF $SafeAt$
$\langle 3\rangle 3.$ PICK $Q1 \in Quorums : \forall\, a \in Q1 : VotedForIn(a,\ v1,\ b1)$
  BY DEF $ChosenIn$
$\langle 3\rangle 4.$ QED
  BY $\langle 3\rangle 2$, $\langle 3\rangle 3$, $QuorumAssumption$, $VotedOnce$, $Z3$ DEF $WontVoteIn$, Inv
$\langle 2\rangle 3.$ QED
  BY $\langle 2\rangle 1$, $\langle 2\rangle 2$

$\langle 1\rangle 2.$ QED
  BY $Invariant$, $\langle 1\rangle 1$, PTL

---

* Modification History
* Last modified *Mon Jul* 22 20:43:22 *CST* 2019 by *hengxin*
* Last modified Sat *Dec* 09 09:56:40 *EST* 2017 by *Saksham*
* Last modified *Tue Nov* 21 19:12:25 *EST* 2017 by *saksh*
* Last modified *Fri Nov* 28 10:39:17 *PST* 2014 by *lamport*
* Last modified Sun *Nov* 23 14:45:09 *PST* 2014 by *lamport*
* Last modified *Mon Nov* 24 02:03:02 *CET* 2014 by *merz*
* Last modified Sat *Nov* 22 12:04:19 *CET* 2014 by *merz*
* Last modified *Fri Nov* 21 17:40:41 *PST* 2014 by *lamport*
* Last modified *Tue Mar* 18 11:37:57 *CET* 2014 by *doligez*
* Last modified Sat *Nov* 24 18:53:09 $GMT - 03{:}00$ 2012 by *merz*
* Created Sat *Nov* 17 16:02:06 *PST* 2012 by *lamport*