MODULE *HPaxos_2*

EXTENDS *HQuorum*, *HLearnerGraph*, *HMessage*, *TLAPS*

$Assert(P,\ str) \triangleq P$

Algorithm specification

****

```
--algorithm HPaxos2{
  variables msgs = {},
            known_msgs = [x ∈ Acceptor ∪ Learner ↦ {}],
            recent_msgs = [a ∈ Acceptor ↦ {}],
            prev_msg = [a ∈ Acceptor ↦ NoMessage],
            decision  = [lb ∈ Learner × Ballot ↦ {}],
            BVal ∈ [Ballot → Value] ;

  define {
    Get1a(m) ≜
        {x ∈ Tran(m) :
           ∧ x.type = "1a"
           ∧ ∀ y ∈ Tran(m) :
              y.type = "1a" ⇒ y.bal ≤ x.bal}

    B(m, bal) ≜ ∃ x ∈ Get1a(m) : bal = x.bal

    V(m, val) ≜ ∃ x ∈ Get1a(m) : val = BVal[x.bal]

    SameBallot(x, y) ≜
        ∀ b ∈ Ballot : B(x, b) ≡ B(y, b)

     Maximal ballot number of any messages known to acceptor a
    MaxBal(a, mbal) ≜
         ∧ ∃ m ∈ known_msgs[a] : B(m, mbal)
         ∧ ∀ x  ∈ known_msgs[a] :
            ∀ b ∈ Ballot : B(x, b) ⇒ b ≤ mbal

    KnownRefs(a, m) ≜ ∀ r ∈ m.ref : r ∈ known_msgs[a]

     The acceptor is _caught_ in a message x if the transitive references of x
     include evidence such as two different messages both signed by the acceptor,
     which have equal previous messges.
    CaughtMsg(x) ≜
        {m ∈ Tran(x) :
            ∧ m.type ≠ "1a"
            ∧ ∃ m1 ∈ Tran(x) :
                ∧ m1.type ≠ "1a"
                ∧ m.acc = m1.acc
```

$$
\begin{array}{l}
\quad\quad \land\, m \neq m1 \\
\quad\quad \land\, m \notin PrevTran(m1) \\
\quad\quad \land\, m1 \notin PrevTran(m)\}
\end{array}
$$

$$Caught(x) \triangleq \{m.acc : m \in CaughtMsg(x)\}$$

$$
\begin{array}{l}
CaughtMsg0(x) \triangleq \\
\quad \{m \in Tran(x) : \\
\quad\quad \land\, m.type \neq \text{“1a”} \\
\quad\quad \land\, \exists\, m1 \in Tran(x) : \\
\quad\quad\quad \land\, m1.type \neq \text{“1a”} \\
\quad\quad\quad \land\, m.acc = m1.acc \\
\quad\quad\quad \land\, m \neq m1 \\
\quad\quad\quad \land\, m.prev = m1.prev\}
\end{array}
$$

$$Caught0(x) \triangleq \{m.acc : m \in CaughtMsg0(x)\}$$

Connected

$$
\begin{array}{l}
ConByQuorum(a,\, b,\, x,\, S) \triangleq \quad \text{a} : Learner,\, b : Learner,\, x : 1b,\, S \in ByzQuorum \\
\quad \land\, [from \mapsto a,\, to \mapsto b,\, q \mapsto S] \in TrustSafe \\
\quad \land\, S \cap Caught(x) = \{\}
\end{array}
$$

$$
\begin{array}{l}
Con(a,\, x) \triangleq \quad \text{a} : Learner,\, x : 1b \\
\quad \{b \in Learner : \\
\quad\quad \exists\, S \in ByzQuorum : ConByQuorum(a,\, b,\, x,\, S)\}
\end{array}
$$

$2a$-message is *_buried_* if there exists a quorum of acceptors that have seen $2a$-messages with different values, the same learner, and higher ballot numbers.

$$
\begin{array}{l}
Buried(l,\, x,\, y) \triangleq \quad x : 2a,\, y : 1b \\
\quad \text{LET } Q \triangleq \{m \in Tran(y) : \\
\quad\quad\quad\quad \exists\, z \in Tran(m) : \\
\quad\quad\quad\quad\quad \land\, z.type = \text{“2a”} \\
\quad\quad\quad\quad\quad \land\, l \in x.lrn \\
\quad\quad\quad\quad\quad \land\, l \in z.lrn \\
\quad\quad\quad\quad\quad \land\, \forall\, bx,\, bz \in Ballot : \\
\quad\quad\quad\quad\quad\quad B(x,\, bx) \land B(z,\, bz) \Rightarrow bx < bz \\
\quad\quad\quad\quad\quad \land\, \forall\, vx,\, vz \in Value : \\
\quad\quad\quad\quad\quad\quad V(x,\, vx) \land V(z,\, vz) \Rightarrow vx \neq vz\} \\
\quad \text{IN} \quad [lr \mapsto l,\, q \mapsto \{m.acc : m \in Q\}] \in TrustLive
\end{array}
$$

Connected $2a$ messages and learners

$$
\begin{array}{l}
Con2as(l,\, x) \triangleq \quad l : Learner,\, x : 1b \\
\quad \{m \in Tran(x) : \\
\quad\quad \land\, m.type = \text{“2a”} \\
\quad\quad \land\, m.acc = x.acc \\
\quad\quad \land\, \exists\, beta \in Con(l,\, x) : \neg Buried(beta,\, m,\, x)\}
\end{array}
$$

Fresh $1b$ messages

$Fresh(l,\, x) \triangleq \quad l : Learner,\, x : 1b$

$\forall\, m \in Con2as(l,\, x) : \forall\, v \in Value : V(x,\, v) \equiv V(m,\, v)$

Quorum of messages referenced by $2a$ for a learner instance

$q(l,\, x) \triangleq \quad x : 2a$

LET $Q \triangleq \{m \in Tran(x) :$
$\wedge\, m.type = \text{“1b”}$
$\wedge\, Fresh(l,\, m)$
$\wedge \forall\, b \in Ballot : B(m,\, b) \equiv B(x,\, b)\}$

IN $\{m.acc : m \in Q\}$

$ChainRef(m) \triangleq$
$\vee\, m.prev = NoMessage$
$\vee\, m.prev \in m.ref \wedge m.prev.acc = m.acc$

$WellFormed1b(m) \triangleq$
$\forall\, y \in Tran(m) :$
$m \neq y \wedge SameBallot(m,\, y) \Rightarrow y.type = \text{“1a”}$

$WellFormed2a(m) \triangleq$
$m.lrn = \{l \in Learner : [lr \mapsto l,\, q \mapsto q(l,\, m)] \in TrustLive\}$

$WellFormed(m) \triangleq$
$\wedge\, m \in Message$
$\wedge \exists\, b \in Ballot : B(m,\, b)$ *TODO* prove it
$\wedge\, ChainRef(m)$
$\wedge\, m.type = \text{“1b”} \Rightarrow$
$\wedge (\exists\, r \in m.refs : r.type = \text{“1a”})$
$\wedge\, WellFormed1b(m)$
$\wedge\, m.type = \text{“2a”} \Rightarrow$
$\wedge\, m.ref \neq \{\}$
$\wedge\, WellFormed2a(m)$

$Known2a(l,\, b,\, v) \triangleq$
$\{x \in known\_msgs[l] :$
$\wedge\, x.type = \text{“2a”}$
$\wedge\, l \in x.lrn$
$\wedge\, B(x,\, b)$
$\wedge\, V(x,\, v)\}$

$ChosenIn(l,\, b,\, v) \triangleq$
$\exists\, S \in \text{SUBSET}\ Known2a(l,\, b,\, v) :$
$[lr \mapsto l,\, q \mapsto \{m.acc : m \in S\}] \in TrustLive$

}

macro *Send*(m) **{** $msgs := msgs \cup \{m\}$ **}**

```
macro Send1a( b ) {
  Send([type ↦ "1a", bal ↦ b, prev ↦ NoMessage, ref ↦ {}])
 }

macro Recv( m ) {
  when ∧ m ∉ known_msgs[self]
       ∧ KnownRefs(self, m) ;
  known_msgs[self] := known_msgs[self] ∪ {m}
 }

macro ProposerSendAction( b ) { Send1a(b) }

macro Process1a( m ) {
  when m.type = "1a" ;
  with ( new1b = [type ↦ "1b",
                  acc ↦ self,
                  prev ↦ prev_msg[self],
                  ref ↦ recent_msgs[self] ∪ {m}] )
  {
    assert new1b ∈ Message ;
    either {
      when WellFormed1b(new1b) ;
      prev_msg[self] := new1b ;
      recent_msgs[self] := {new1b} ;
      Send(new1b)
     }
    or {
      when ¬ WellFormed1b(new1b) ;
      skip
     }
   }
 }

macro Process1b( m ) {
  when m.type = "1b" ;
  with ( LL ∈ SUBSET Learner,
         new2a = [type ↦ "2a",
                  lrn ↦ LL,
                  acc ↦ self,
                  prev ↦ prev_msg[self],
                  ref ↦ recent_msgs[self] ∪ {m}] )
  {
    assert new2a ∈ Message      ;
    when WellFormed2a(new2a) ;
    prev_msg[self] := new2a ;
    recent_msgs[self] := {new2a} ;
```

```
    Send(new2a)
   }
 }

macro Process2a( m ) {
  when m.type = "2a" ;
  recent_msgs[self] := recent_msgs[self] ∪ {m}
 }

macro FakeSend1b( ) {
  with ( fin ∈ FINSUBSET(msgs, RefCardinality),
         new1b = [type ↦ "1b", acc ↦ self, ref ↦ fin] )
  {
    when WellFormed(new1b) ;
    Send(new1b)
   }
 }

macro FakeSend2a( ) {
  with ( fin ∈ FINSUBSET(msgs, RefCardinality),
         LL ∈ SUBSET Learner,
         new2a = [type ↦ "2a", lrn ↦ LL, acc ↦ self, ref ↦ fin] )
  {
    when WellFormed(new2a) ;
    Send(new2a)
   }
 }

macro LearnerRecv( m ) {
  when WellFormed(m) ;
  Recv(m)
 }

macro LearnerDecide( b, v ) {
  when ChosenIn(self, b, v) ;
  decision[⟨self, b⟩] := decision[self, b] ∪ {v}
 }

process ( proposer ∈ Proposer ) {
  propose:   while ( TRUE ) {
    with ( b ∈ Ballot ) { ProposerSendAction(b) }
   }
 }

process ( safe_acceptor ∈ SafeAcceptor ) {
  safe:   while ( TRUE ) {
    with ( m ∈ msgs ) {
```

```
        Recv(m);
        when WellFormed(m);
        either Process1a(m)
        or     Process1b(m)
        or     Process2a(m)
      }
    }
  }

  process ( learner ∈ Learner ) {
    learn:   while ( TRUE ) {
      either with ( m ∈ msgs ) LearnerRecv(m)
      or     with ( b ∈ Ballot, v ∈ Value ) LearnerDecide(b, v)
    }
  }

  process ( fake_acceptor ∈ FakeAcceptor ) {
    fake:   while ( TRUE ) {
      either FakeSend1b()
      or     FakeSend2a()
    }
  }
}
```

****

BEGIN TRANSLATION ($chksum(pcal) =$ "e34bd140" $\land\ chksum(tla) =$ "89878241")

VARIABLES $msgs$, $known\_msgs$, $recent\_msgs$, $prev\_msg$, $decision$, $BVal$

define statement

$$Get1a(m) \triangleq \{x \in Tran(m) : \land\, x.type = \text{“1a”} \land\, \forall\, y \in Tran(m) : y.type = \text{“1a”} \Rightarrow y.bal \leq x.bal\}$$

$$B(m,\, bal) \triangleq \exists\, x \in Get1a(m) : bal = x.bal$$

$$V(m,\, val) \triangleq \exists\, x \in Get1a(m) : val = BVal[x.bal]$$

$$SameBallot(x,\, y) \triangleq \forall\, b \in Ballot : B(x,\, b) \equiv B(y,\, b)$$

$$MaxBal(a,\, mbal) \triangleq \land\, \exists\, m \in known\_msgs[a] : B(m,\, mbal) \land\, \forall\, x \in known\_msgs[a] : \forall\, b \in Ballot : B(x,\, b) \Rightarrow b \leq mbal$$

$$KnownRefs(a,\, m) \triangleq \forall\, r \in m.ref : r \in known\_msgs[a]$$

$$
\begin{array}{l}
CaughtMsg(x) \triangleq \\
\quad \{m \in Tran(x) : \\
\qquad \land\, m.type \neq \text{“1a”} \\
\qquad \land\, \exists\, m1 \in Tran(x) : \\
\qquad\quad \land\, m1.type \neq \text{“1a”} \\
\qquad\quad \land\, m.acc = m1.acc \\
\qquad\quad \land\, m \neq m1 \\
\qquad\quad \land\, m \notin PrevTran(m1) \\
\qquad\quad \land\, m1 \notin PrevTran(m)\}
\end{array}
$$

$$
Caught(x) \triangleq \{m.acc : m \in CaughtMsg(x)\}
$$

$$
\begin{array}{l}
CaughtMsg0(x) \triangleq \\
\quad \{m \in Tran(x) : \\
\qquad \land\, m.type \neq \text{“1a”} \\
\qquad \land\, \exists\, m1 \in Tran(x) : \\
\qquad\quad \land\, m1.type \neq \text{“1a”} \\
\qquad\quad \land\, m.acc = m1.acc \\
\qquad\quad \land\, m \neq m1 \\
\qquad\quad \land\, m.prev = m1.prev\}
\end{array}
$$

$$
Caught0(x) \triangleq \{m.acc : m \in CaughtMsg0(x)\}
$$

$$
\begin{array}{l}
ConByQuorum(a,\, b,\, x,\, S) \triangleq \\
\quad \land\, [from \mapsto a,\, to \mapsto b,\, q \mapsto S] \in TrustSafe \\
\quad \land\, S \cap Caught(x) = \{\}
\end{array}
$$

$$
\begin{array}{l}
Con(a,\, x) \triangleq \\
\quad \{b \in Learner : \\
\qquad \exists\, S \in ByzQuorum : ConByQuorum(a,\, b,\, x,\, S)\}
\end{array}
$$

$$
\begin{array}{l}
Buried(l,\, x,\, y) \triangleq \\
\quad \text{LET } Q \triangleq \{m \in Tran(y) : \\
\qquad\qquad\qquad \exists\, z \in Tran(m) : \\
\qquad\qquad\qquad\quad \land\, z.type = \text{“2a”} \\
\qquad\qquad\qquad\quad \land\, l \in x.lrn \\
\qquad\qquad\qquad\quad \land\, l \in z.lrn \\
\qquad\qquad\qquad\quad \land\, \forall\, bx,\, bz \in Ballot : \\
\qquad\qquad\qquad\qquad B(x,\, bx) \land B(z,\, bz) \Rightarrow bx < bz \\
\qquad\qquad\qquad\quad \land\, \forall\, vx,\, vz \in Value : \\
\qquad\qquad\qquad\qquad V(x,\, vx) \land V(z,\, vz) \Rightarrow vx \neq vz\} \\
\quad \text{IN } \quad [lr \mapsto l,\, q \mapsto \{m.acc : m \in Q\}] \in TrustLive
\end{array}
$$

$$
Con2as(l,\, x) \triangleq
$$

$\{m \in \mathit{Tran}(x) :$
$\quad \land\, m.\mathit{type} = \text{“2a”}$
$\quad \land\, m.\mathit{acc} = x.\mathit{acc}$
$\quad \land\, \exists\, \mathit{beta} \in \mathit{Con}(l,\, x) : \neg \mathit{Buried}(\mathit{beta},\, m,\, x)\}$

$\mathit{Fresh}(l,\, x) \triangleq$
$\quad \forall\, m \in \mathit{Con2as}(l,\, x) : \forall\, v \in \mathit{Value} : V(x,\, v) \equiv V(m,\, v)$

$q(l,\, x) \triangleq$
$\quad \text{LET } Q \triangleq \{m \in \mathit{Tran}(x) :$
$\qquad\qquad \land\, m.\mathit{type} = \text{“1b”}$
$\qquad\qquad \land\, \mathit{Fresh}(l,\, m)$
$\qquad\qquad \land\, \forall\, b \in \mathit{Ballot} : B(m,\, b) \equiv B(x,\, b)\}$
$\quad \text{IN } \{m.\mathit{acc} : m \in Q\}$

$\mathit{ChainRef}(m) \triangleq$
$\quad \lor\, m.\mathit{prev} = \mathit{NoMessage}$
$\quad \lor\, m.\mathit{prev} \in m.\mathit{ref} \land m.\mathit{prev}.\mathit{acc} = m.\mathit{acc}$

$\mathit{WellFormed1b}(m) \triangleq$
$\quad \forall\, y \in \mathit{Tran}(m) :$
$\qquad m \neq y \land \mathit{SameBallot}(m,\, y) \Rightarrow y.\mathit{type} = \text{“1a”}$

$\mathit{WellFormed2a}(m) \triangleq$
$\quad m.\mathit{lrn} = \{l \in \mathit{Learner} : [\mathit{lr} \mapsto l,\, q \mapsto q(l,\, m)] \in \mathit{TrustLive}\}$

$\mathit{WellFormed}(m) \triangleq$
$\quad \land\, m \in \mathit{Message}$
$\quad \land\, \exists\, b \in \mathit{Ballot} : B(m,\, b)$
$\quad \land\, \mathit{ChainRef}(m)$
$\quad \land\, m.\mathit{type} = \text{“1b”} \Rightarrow$
$\qquad \land\, (\exists\, r \in m.\mathit{refs} : r.\mathit{type} = \text{“1a”})$
$\qquad \land\, \mathit{WellFormed1b}(m)$
$\quad \land\, m.\mathit{type} = \text{“2a”} \Rightarrow$
$\qquad \land\, m.\mathit{ref} \neq \{\}$
$\qquad \land\, \mathit{WellFormed2a}(m)$

$\mathit{Known2a}(l,\, b,\, v) \triangleq$
$\quad \{x \in \mathit{known\_msgs}[l] :$
$\qquad \land\, x.\mathit{type} = \text{“2a”}$
$\qquad \land\, l \in x.\mathit{lrn}$
$\qquad \land\, B(x,\, b)$
$\qquad \land\, V(x,\, v)\}$

$\mathit{ChosenIn}(l,\, b,\, v) \triangleq$
$\quad \exists\, S \in \text{SUBSET } \mathit{Known2a}(l,\, b,\, v) :$

$[lr \mapsto l,\ q \mapsto \{m.acc : m \in S\}] \in TrustLive$

$vars \triangleq \langle msgs,\ known\_msgs,\ recent\_msgs,\ prev\_msg,\ decision,\ BVal \rangle$

$ProcSet \triangleq (Proposer) \cup (SafeAcceptor) \cup (Learner) \cup (FakeAcceptor)$

$Init \triangleq$ Global variables
$\land\ msgs = \{\}$
$\land\ known\_msgs = [x \in Acceptor \cup Learner \mapsto \{\}]$
$\land\ recent\_msgs = [a \in Acceptor \mapsto \{\}]$
$\land\ prev\_msg = [a \in Acceptor \mapsto NoMessage]$
$\land\ decision\ \ = [lb \in Learner \times Ballot \mapsto \{\}]$
$\land\ BVal \in [Ballot \to Value]$

$proposer(self) \triangleq \land \exists\, b \in Ballot :$
$msgs' = (msgs \cup \{([type \mapsto \text{“1a”},\ bal \mapsto b,\ prev \mapsto NoMessage,\ ref \mapsto \{\}])\})$
$\land$ UNCHANGED $\langle known\_msgs,\ recent\_msgs,\ prev\_msg,\ decision,$
$BVal \rangle$

$safe\_acceptor(self) \triangleq \land \exists\, m \in msgs :$
$\land\ \land\ m \notin known\_msgs[self]$
$\land\ KnownRefs(self,\ m)$
$\land\ known\_msgs' = [known\_msgs$ EXCEPT $![self] = known\_msgs[self] \cup \{m\}]$
$\land\ WellFormed(m)$
$\land\ \lor\ \land\ m.type = \text{“1a”}$
$\land$ LET $new1b \triangleq [type \mapsto \text{“1b”},$
$acc \mapsto self,$
$prev \mapsto prev\_msg[self],$
$ref \mapsto recent_msgs[self] \cup \{m\}]$IN
$\land\ Assert(new1b \in Message,$
“Failure of assertion at line 164, column 7 of macro called at line
$\land\ \lor\ \land\ WellFormed1b(new1b)$
$\land\ prev\_msg' = [prev\_msg$ EXCEPT $![self] = new1b]$
$\land\ recent\_msgs' = [recent\_msgs$ EXCEPT $![self] = \{new1b\}]$
$\land\ msgs' = (msgs \cup \{new1b\})$
$\lor\ \land\ \neg WellFormed1b(new1b)$
$\land$ TRUE
$\land$ UNCHANGED $\langle msgs,\ recent\_msgs,\ prev\_msg \rangle$
$\lor\ \land\ m.type = \text{“1b”}$
$\land\ \exists\, LL \in$ SUBSET $Learner :$
LET $new2a \triangleq [type \mapsto \text{“2a”},$
$lrn \mapsto LL,$
$acc \mapsto self,$
$prev \mapsto prev\_msg[self],$
$ref \mapsto recent_msgs[self] \cup \{m\}]$IN
$\land\ Assert(new2a \in Message,$

"Failure of assertion at line 187, column 7 of macro called at li

$\land\ \mathit{WellFormed2a}(\mathit{new2a})$
$\land\ \mathit{prev\_msg}' = [\mathit{prev\_msg}\ \text{EXCEPT}\ ![\mathit{self}] = \mathit{new2a}]$
$\land\ \mathit{recent\_msgs}' = [\mathit{recent\_msgs}\ \text{EXCEPT}\ ![\mathit{self}] = \{\mathit{new2a}\}]$
$\land\ \mathit{msgs}' = (\mathit{msgs} \cup \{\mathit{new2a}\})$
$\lor\ \land\ m.\mathit{type} = \text{"2a"}$
$\quad \land\ \mathit{recent\_msgs}' = [\mathit{recent\_msgs}\ \text{EXCEPT}\ ![\mathit{self}] = \mathit{recent\_msgs}[\mathit{self}] \cup \{m\}]$
$\quad \land\ \text{UNCHANGED}\ \langle \mathit{msgs}, \mathit{prev\_msg} \rangle$
$\land\ \text{UNCHANGED}\ \langle \mathit{decision}, \mathit{BVal} \rangle$

$\mathit{learner}(\mathit{self}) \triangleq \land \lor \land \exists m \in \mathit{msgs} :$
$\qquad \land\ \mathit{WellFormed}(m)$
$\qquad \land\ \land\ m \notin \mathit{known\_msgs}[\mathit{self}]$
$\qquad \quad \land\ \mathit{KnownRefs}(\mathit{self}, m)$
$\qquad \land\ \mathit{known\_msgs}' = [\mathit{known\_msgs}\ \text{EXCEPT}\ ![\mathit{self}] = \mathit{known\_msgs}[\mathit{self}] \cup \{m\}]$
$\quad \land\ \text{UNCHANGED}\ \mathit{decision}$
$\lor\ \land\ \exists b \in \mathit{Ballot} :$
$\qquad \exists v \in \mathit{Value} :$
$\qquad \quad \land\ \mathit{ChosenIn}(\mathit{self}, b, v)$
$\qquad \quad \land\ \mathit{decision}' = [\mathit{decision}\ \text{EXCEPT}\ ![\langle \mathit{self}, b \rangle] = \mathit{decision}[\mathit{self}, b] \cup \{v\}]$
$\quad \land\ \text{UNCHANGED}\ \mathit{known\_msgs}$
$\land\ \text{UNCHANGED}\ \langle \mathit{msgs}, \mathit{recent\_msgs}, \mathit{prev\_msg}, \mathit{BVal} \rangle$

$\mathit{fake\_acceptor}(\mathit{self}) \triangleq \land \lor \land \exists \mathit{fin} \in \mathit{FINSUBSET}(\mathit{msgs}, \mathit{RefCardinality}) :$
$\qquad \text{LET}\ \mathit{new1b} \triangleq [\mathit{type} \mapsto \text{"1b"}, \mathit{acc} \mapsto \mathit{self}, \mathit{ref} \mapsto \mathit{fin}]\ \text{IN}$
$\qquad \quad \land\ \mathit{WellFormed}(\mathit{new1b})$
$\qquad \quad \land\ \mathit{msgs}' = (\mathit{msgs} \cup \{\mathit{new1b}\})$
$\quad \lor\ \land\ \exists \mathit{fin} \in \mathit{FINSUBSET}(\mathit{msgs}, \mathit{RefCardinality}) :$
$\qquad \exists \mathit{LL} \in \text{SUBSET}\ \mathit{Learner} :$
$\qquad \quad \text{LET}\ \mathit{new2a} \triangleq [\mathit{type} \mapsto \text{"2a"}, \mathit{lrn} \mapsto \mathit{LL}, \mathit{acc} \mapsto \mathit{self}, \mathit{ref} \mapsto \mathit{fin}]\ \text{IN}$
$\qquad \qquad \land\ \mathit{WellFormed}(\mathit{new2a})$
$\qquad \qquad \land\ \mathit{msgs}' = (\mathit{msgs} \cup \{\mathit{new2a}\})$
$\land\ \text{UNCHANGED}\ \langle \mathit{known\_msgs}, \mathit{recent\_msgs}, \mathit{prev\_msg},$
$\qquad \mathit{decision}, \mathit{BVal} \rangle$

$\mathit{Next} \triangleq (\exists \mathit{self} \in \mathit{Proposer} : \mathit{proposer}(\mathit{self}))$
$\qquad \lor (\exists \mathit{self} \in \mathit{SafeAcceptor} : \mathit{safe\_acceptor}(\mathit{self}))$
$\qquad \lor (\exists \mathit{self} \in \mathit{Learner} : \mathit{learner}(\mathit{self}))$
$\qquad \lor (\exists \mathit{self} \in \mathit{FakeAcceptor} : \mathit{fake\_acceptor}(\mathit{self}))$

$\mathit{Spec} \triangleq \mathit{Init} \land \Box[\mathit{Next}]_{\mathit{vars}}$

END TRANSLATION

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

$Recv(a,\ m) \triangleq$
$\quad \land\ m \notin known\_msgs[a]$ ignore known messages
$\quad \land\ KnownRefs(a,\ m)$
$\quad \land\ known\_msgs' = [known\_msgs \text{ EXCEPT } ![a] = known\_msgs[a] \cup \{m\}]$

$Send1a(b) \triangleq$
$\quad \land\ Send([type \mapsto \text{“1a”},\ bal \mapsto b,\ prev \mapsto NoMessage,\ ref \mapsto \{\}])$
$\quad \land\ \text{UNCHANGED } \langle known\_msgs,\ recent\_msgs,\ prev\_msg\rangle$
$\quad \land\ \text{UNCHANGED } decision$
$\quad \land\ \text{UNCHANGED } BVal$

$Process1a(a,\ m) \triangleq$
$\quad \text{LET } new1b \triangleq [type \mapsto \text{“1b”},\ acc \mapsto a,$
$\qquad\qquad\qquad prev \mapsto prev\_msg[a],$
$\qquad\qquad\qquad ref \mapsto recent\_msgs[a] \cup \{m\}] \text{IN}$
$\quad \land\ m.type = \text{“1a”}$
$\quad \land\ Recv(a,\ m)$
$\quad \land\ WellFormed(m)$
$\quad \land\ new1b \in Message$
$\quad \land\ \lor\ \land\ WellFormed1b(new1b)$
$\qquad\quad \land\ Send(new1b)$
$\qquad\quad \land\ recent\_msgs' = [recent\_msgs \text{ EXCEPT } ![a] = \{new1b\}]$
$\qquad\quad \land\ prev\_msg' = [prev\_msg \text{ EXCEPT } ![a] = new1b]$
$\qquad \lor\ \land\ \neg WellFormed1b(new1b)$
$\qquad\quad \land\ \text{UNCHANGED } \langle msgs,\ prev\_msg,\ recent\_msgs\rangle$
$\quad \land\ \text{UNCHANGED } decision$
$\quad \land\ \text{UNCHANGED } BVal$

$Process1b(a,\ m) \triangleq$
$\quad \land\ m.type = \text{“1b”}$
$\quad \land\ Recv(a,\ m)$
$\quad \land\ WellFormed(m)$
$\quad \land\ \exists\, LL \in \text{SUBSET } Learner :$
$\qquad \text{LET } new2a \triangleq [type \mapsto \text{“2a”},\ lrn \mapsto LL,\ acc \mapsto a,$
$\qquad\qquad\qquad\qquad prev \mapsto prev\_msg[a],$
$\qquad\qquad\qquad\qquad ref \mapsto recent\_msgs[a] \cup \{m\}] \text{IN}$
$\qquad\quad \land\ new2a \in Message$
$\qquad\quad \land\ WellFormed2a(new2a)$
$\qquad\quad \land\ Send(new2a)$
$\qquad\quad \land\ recent\_msgs' = [recent\_msgs \text{ EXCEPT } ![a] = \{new2a\}]$
$\qquad\quad \land\ prev\_msg' = [prev\_msg \text{ EXCEPT } ![a] = new2a]$
$\quad \land\ \text{UNCHANGED } decision$
$\quad \land\ \text{UNCHANGED } BVal$

$Process2a(a,\ m) \triangleq$
$\quad \land\ m.type = \text{“2a”}$

$\land\ Recv(a,\ m)$
$\land\ WellFormed(m)$
$\land\ recent\_msgs' = [recent\_msgs \text{ EXCEPT } ![a] = recent\_msgs[a] \cup \{m\}]$
$\land$ UNCHANGED $\langle msgs,\ prev\_msg\rangle$
$\land$ UNCHANGED $decision$
$\land$ UNCHANGED $BVal$

$ProposerSendAction(p) \triangleq$
$\exists\, bal \in Ballot : Send1a(bal)$

$AcceptorProcessAction(a) \triangleq$
$\exists\, m \in msgs :$
$\lor\ Process1a(a,\ m)$
$\lor\ Process1b(a,\ m)$
$\lor\ Process2a(a,\ m)$

$FakeSend1b(a) \triangleq$
$\land \exists\, fin \in FINSUBSET(msgs,\ RefCardinality) :$
LET $new1b \triangleq [type \mapsto$ "1b", $acc \mapsto a,\ ref \mapsto fin]$IN
$\land\ WellFormed(new1b)$
$\land\ Send(new1b)$
$\land$ UNCHANGED $\langle known\_msgs,\ recent\_msgs,\ prev\_msg\rangle$
$\land$ UNCHANGED $decision$
$\land$ UNCHANGED $BVal$

$FakeSend2a(a) \triangleq$
$\land \exists\, fin \in FINSUBSET(msgs,\ RefCardinality) :$
$\exists\, LL \in$ SUBSET $Learner :$
LET $new2a \triangleq [type \mapsto$ "2a", $lrn \mapsto LL,\ acc \mapsto a,\ ref \mapsto fin]$IN
$\land\ WellFormed(new2a)$
$\land\ Send(new2a)$
$\land$ UNCHANGED $\langle known\_msgs,\ recent\_msgs,\ prev\_msg\ \rangle$
$\land$ UNCHANGED $decision$
$\land$ UNCHANGED $BVal$

$LearnerRecv(l,\ m) \triangleq$
$\land\ Recv(l,\ m)$
$\land\ WellFormed(m)$
$\land$ UNCHANGED $\langle msgs,\ recent\_msgs,\ prev\_msg\rangle$
$\land$ UNCHANGED $decision$
$\land$ UNCHANGED $BVal$

$LearnerDecide(l,\ b,\ v) \triangleq$
$\land\ ChosenIn(l,\ b,\ v)$
$\land\ decision' = [decision \text{ EXCEPT } ![\langle l,\ b\rangle] = decision[l,\ b] \cup \{v\}]$
$\land$ UNCHANGED $\langle msgs,\ known\_msgs,\ recent\_msgs,\ prev\_msg\rangle$
$\land$ UNCHANGED $BVal$

$LearnerAction(lrn) \triangleq$
$\lor \exists\, m \in msgs :$
$LearnerRecv(lrn,\ m)$
$\lor \exists\, bal \in Ballot\ :$
$\exists\, val \in Value :$
$LearnerDecide(lrn,\ bal,\ val)$

$FakeAcceptorAction(a) \triangleq$
$\lor FakeSend1b(a)$
$\lor FakeSend2a(a)$

$NextTLA \triangleq$
$\lor \exists\, p \in Proposer :$
$ProposerSendAction(p)$
$\lor \exists\, acc \in SafeAcceptor :$
$AcceptorProcessAction(acc)$
$\lor \exists\, lrn \in Learner :$
$LearnerAction(lrn)$
$\lor \exists\, acc \in FakeAcceptor :$
$FakeAcceptorAction(acc)$

THEOREM $NextDef \triangleq Next \equiv NextTLA$
$\langle 1\rangle 1$. ASSUME NEW $self \in Proposer$
PROVE $proposer(self) \equiv ProposerSendAction(self)$
BY DEF $proposer$, $ProposerSendAction$, $Send1a$, $Send$
$\langle 1\rangle 2$. ASSUME NEW $self \in SafeAcceptor$
PROVE $safe\_acceptor(self) \equiv AcceptorProcessAction(self)$
BY $Zenon$ DEF $safe\_acceptor$, $AcceptorProcessAction$, $Process1a$, $Process1b$, $Process2a$, $Recv$, $Send$, Ass
$\langle 1\rangle 3$. ASSUME NEW $self \in Learner$
PROVE $learner(self) \equiv LearnerAction(self)$
BY $Zenon$ DEF $learner$, $LearnerAction$, $LearnerRecv$, $LearnerDecide$, $Recv$
$\langle 1\rangle 4$. ASSUME NEW $self \in FakeAcceptor$
PROVE $fake\_acceptor(self) \equiv FakeAcceptorAction(self)$
BY $Zenon$ DEF $fake\_acceptor$, $FakeAcceptorAction$, $FakeSend1b$, $FakeSend2a$, $Send$
$\langle 1\rangle 5$. QED BY $\langle 1\rangle 1$, $\langle 1\rangle 2$, $\langle 1\rangle 3$, $\langle 1\rangle 4$ DEF $Next$, $NextTLA$

$Safety \triangleq$
$\forall\, L1,\ L2 \in Learner : \forall\, B1,\ B2 \in Ballot : \forall\, V1,\ V2 \in Value :$
$\langle L1,\ L2\rangle \in Ent \land$
$V1 \in decision[L1,\ B1] \land V2 \in decision[L2,\ B2] \Rightarrow$
$V1 = V2$

THEOREM $SafetyResult \triangleq Spec \Rightarrow \Box Safety$

Sanity check propositions

$SanityCheck0 \triangleq$
$\forall L \in Learner : Cardinality(known\_msgs[L]) = 0$

$SanityCheck1 \triangleq$
$\forall L \in Learner : \forall m1,\ m2 \in known\_msgs[L] :$
$\forall b1,\ b2 \in Ballot :$
$B(m1,\ b1) \land B(m2,\ b2) \Rightarrow b1 = b2$

$2aNotSent \triangleq$
$\forall M \in msgs : M.type \neq$ "2a"

$2aNotSentBySafeAcceptor \triangleq$
$\forall M \in msgs : M.type =$ "2a" $\Rightarrow M.acc \notin SafeAcceptor$

$1bNotSentBySafeAcceptor \triangleq$
$\forall M \in msgs : M.type =$ "1b" $\Rightarrow M.acc \notin SafeAcceptor$

$NoDecision \triangleq$
$\forall L \in Learner : \forall BB \in Ballot : \forall VV \in Value :$
$VV \notin decision[L,\ BB]$

$UniqueDecision \triangleq$
$\forall L1,\ L2 \in Learner : \forall B1,\ B2 \in Ballot : \forall V1,\ V2 \in Value :$
$V1 \in decision[L1,\ B1] \land V2 \in decision[L2,\ B2] \Rightarrow$
$V1 = V2$

* Modification History
* Last modified *Mon* May 20 16:48:25 *CEST* 2024 by *karbyshev*
* Created *Mon Jun* 19 12:24:03 *CEST* 2022 by *karbyshev*