MODULE *TPaxos*

EXTENDS *Integers*, *FiniteSets*

CONSTANTS

Participant, the set of partipants

Value, the set of possible input values for *Participant* to propose

Quorum

$None \triangleq$ CHOOSE $b : b \notin Value$

$NP \triangleq Cardinality(Participant)$ number of $p \in Participant$

ASSUME $QuorumAssumption \triangleq$

$\land \quad \forall\, Q \in Quorum : Q \subseteq Participant$

$\land \quad \forall\, Q1,\ Q2 \in Quorum : Q1 \cap Q2 \neq \{\}$

$Ballot \triangleq Nat$

$Max(m,\ n) \triangleq$ IF $m > n$ THEN m ELSE n

$Injective(f) \triangleq \forall\, a,\ b \in$ DOMAIN $f : (a \neq b) \Rightarrow (f[a] \neq f[b])$

$PIndex \triangleq$ CHOOSE $f \in [Participant \to 1\ ..\ NP] : Injective(f)$

$Bals(p) \triangleq \{b \in Ballot : b\%NP = PIndex[p] - 1\}$ allocate ballots for each $p \in Participant$

$State \triangleq [maxBal \ : Ballot \cup \{-1\},$
$\quad maxVBal : Ballot \cup \{-1\},\ maxVVal : Value \cup \{None\}]$

$InitState \triangleq [maxBal \mapsto -1,\ maxVBal \mapsto -1,\ maxVVal \mapsto None]$

$Message \triangleq [from : Participant,\ to :$ SUBSET $Participant,\ state : [Participant \to State]]$

VARIABLES

state, $state[p][q]$: the state of $q \in Participant$ from the view of $p \in Participant$

msgs the set of messages that have been sent

$vars \triangleq \langle state,\ msgs \rangle$

$TypeOK \triangleq$

$\land \quad state \in [Participant \to [Participant \to State]]$

$\land \quad msgs \subseteq Message$

$Init \triangleq$

$\land\ state = [p \in Participant \mapsto [q \in Participant \mapsto InitState]]$

$\land\ msgs = \{\}$

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

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

$Prepare(p,\ b) \triangleq$

$\land \quad b \in Bals(p)$

$\land \quad state[p][p].maxBal < b$

$\land \quad state' = [state$ EXCEPT $![p][p].maxBal = b]$

$\land \quad Send([from \mapsto p,\ to \mapsto Participant,\ state \mapsto state'[p]])$

$q \in Participant$ 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 $OnMessage(q)$ of $TPaxosAP$.

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

$$
\begin{array}{l}
UpdateState(q,\ p,\ pp) \ \triangleq \\
\quad state' = [state \ \text{EXCEPT} \\
\qquad\qquad ![q][p].maxBal = Max(@,\ pp.maxBal), \\
\qquad\qquad ![q][p].maxVBal = Max(@,\ pp.maxVBal), \\
\qquad\qquad ![q][p].maxVVal = \text{IF } state[q][p].maxVBal < pp.maxVBal \\
\qquad\qquad\qquad\qquad\qquad \text{THEN } pp.maxVVal \text{ ELSE } @, \\
\qquad\qquad ![q][q].maxBal = Max(@,\ pp.maxBal), \quad \text{make promise} \\
\qquad\qquad ![q][q].maxVBal = \text{IF } state[q][q].maxBal \leq pp.maxVBal \ \text{ accept} \\
\qquad\qquad\qquad\qquad\qquad \text{THEN } pp.maxVBal \text{ ELSE } @, \\
\qquad\qquad ![q][q].maxVVal = \text{IF } state[q][q].maxBal \leq pp.maxVBal \ \text{ accept} \\
\qquad\qquad\qquad\qquad\qquad \text{THEN } pp.maxVVal \text{ ELSE } @]
\end{array}
$$

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

$$
\begin{array}{l}
OnMessage(q) \ \triangleq \\
\quad \exists\, m \in msgs : \\
\qquad \land\ q \in m.to \\
\qquad \land \text{LET } p \triangleq m.from \\
\qquad\quad \text{IN} \quad UpdateState(q,\ p,\ m.state[p]) \\
\qquad \land \text{LET } qm \triangleq [from \mapsto m.from,\ to \mapsto m.to \setminus \{q\},\ state \mapsto m.state] \ \text{ remove } q \text{ from to} \\
\qquad\qquad\ nm \triangleq [from \mapsto q,\ to \mapsto \{m.from\},\ state \mapsto state'[q]] \ \text{ new message to reply} \\
\qquad\quad \text{IN} \quad \text{IF } \lor\ m.state[q].maxBal < state'[q][q].maxBal \\
\qquad\qquad\qquad\qquad \lor\ m.state[q].maxVBal < state'[q][q].maxVBal \\
\qquad\qquad\qquad \text{THEN } msgs' = (msgs \setminus \{m\}) \cup \{qm,\ nm\} \\
\qquad\qquad\qquad \text{ELSE } msgs' = (msgs \setminus \{m\}) \cup \{qm\}
\end{array}
$$

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

$$
\begin{array}{l}
Accept(p,\ b,\ v) \triangleq \\
\quad \land\ b \in Bals(p) \\
\quad \land\ state[p][p].maxBal \leq b \quad \text{corresponding to the first conjunction in } Voting \\
\quad \land\ state[p][p].maxVBal \neq b \quad \text{correspongding to the second conjunction in } Voting \\
\quad \text{pick } v \text{ based on a quorum of } state[p] \\
\quad \land\ \exists\, Q \in Quorum : \quad \text{collecting ``enough'' replies to } Prepare(p,\ b) \\
\qquad \land\ \forall\, q \in Q : state[p][q].maxBal = b \\
\qquad \land\ \lor\, \forall\, q \in Q : state[p][q].maxVBal = -1 \quad \text{free to pick its own value} \\
\qquad\quad \lor\, \exists\, q \in Q : \quad v \text{ is the value with the highest } maxVBal \\
\qquad\qquad \land\ state[p][q].maxVVal = v \\
\qquad\qquad \land\ \forall\, r \in Participant : state[p][q].maxVBal \geq state[p][r].maxVBal \\
\quad \land\ state' = [state \text{ EXCEPT } ![p][p].maxVBal = b, \\
\qquad\qquad\qquad\qquad\qquad\quad ![p][p].maxVVal = v] \\
\quad \land\ Send([from \mapsto p,\ to \mapsto Participant,\ state \mapsto state'[p]])
\end{array}
$$

$$
\begin{array}{l}
Next \triangleq \exists\, p \in Participant : \lor\, OnMessage(p) \\
\qquad\qquad\qquad\qquad\qquad\quad \lor\, \exists\, b \in Ballot : \lor\, Prepare(p,\ b) \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \lor\, \exists\, v \in Value : Accept(p,\ b,\ v) \\
Spec \triangleq Init \land \Box[Next]_{vars}
\end{array}
$$

$$
\begin{array}{l}
ChosenP(p) \triangleq \quad \text{the set of values chosen by } p \in Participant \\
\quad \{v \in Value : \exists\, b \in Ballot : \\
\qquad\qquad\qquad \exists\, Q \in Quorum : \forall\, q \in Q : \land\, state[p][q].maxVBal = b \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \land\, state[p][q].maxVVal = v\}
\end{array}
$$

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

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

THEOREM $Spec \Rightarrow \Box Consistency$

\ * Modification History
\ * Last modified Sun *Sep* 08 21:30:28 *CST* 2019 by *pure_*
\ * Last modified *Wed Jul* 31 15:00:12 *CST* 2019 by *hengxin*
\ * 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_*