MODULE *Galene*

Galene: is a linearizable protocol used in *ccKVS* of Scale-out *ccNUMA* [*Eurosys'18*]

EXTENDS *Integers*, *FiniteSets*

CONSTANTS *G_NODES*, *G_MAX_VERSION*

VARIABLES *msgs*, *nodeTS*, *nodeState*, *nodeRcvedAcks*

all *Galene*(+ environment) variables

$gvars \triangleq \langle msgs,\ nodeTS,\ nodeState,\ nodeRcvedAcks \rangle$

A buffer maintaining all network messages. Messages are only appended to this variable (not removed once delivered) intentionally to check protocols tolerance in dublicates and reorderings

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

Check if all acknowledgments for a write have been received

$receivedAllAcks(n) \triangleq (G\_NODES \setminus \{n\}) \subseteq nodeRcvedAcks[n]$

$equalTS(v1,\ tb1,\ v2,\ tb2) \triangleq$ Timestamp equality

$\land\ v1 = v2$

$\land\ tb1 = tb2$

$greaterTS(v1,\ tb1,\ v2,\ tb2) \triangleq$ Timestamp comparison

$\lor\ v1 > v2$

$\lor\ \land\ v1 = v2$

$\land\ tb1 > tb2$

$GMessage \triangleq$ Messages exchanged by *Galene*

$[type : \{\text{“INV”},\ \text{“ACK”}\},\ sender : G\_NODES,\ version : 0\ ..\ G\_MAX\_VERSION,\ tieBreaker : G\_NODES]$

$\cup$

We do not send the Value w/ *VALs* (*TS* suffices to check consistency)

$[type : \{\text{“VAL”}\},\ version : 0\ ..\ G\_MAX\_VERSION,\ tieBreaker : G\_NODES]$

$GTypeOK \triangleq$ The type correctness invariant

$\land\ msgs \subseteq GMessage$

$\land\ \forall\, n \in G\_NODES : nodeRcvedAcks[n] \subseteq (G\_NODES \setminus \{n\})$

$\land\ nodeTS \in [G\_NODES \to [version : 0\ ..\ G\_MAX\_VERSION,$

$$\begin{array}{lll} & & tieBreaker : G\_NODES \quad]] \\ \land\ nodeState & \in [G\_NODES \to \{\text{“valid”}, \text{“invalid”}, \text{“write”}\}] & \end{array}$$

The consistent invariant: all alive nodes in valid state should have the same TS (value)

$$\begin{array}{l} GConsistent \triangleq \\ \quad \forall\, k,\, s \in G\_NODES : \lor\ nodeState[k] \neq \text{“valid”} \\ \qquad\qquad\qquad\qquad\ \lor\ nodeState[s] \neq \text{“valid”} \\ \qquad\qquad\qquad\qquad\ \lor\ nodeTS[s] \quad = nodeTS[k] \end{array}$$

$$\begin{array}{l} GSWMR \triangleq \quad \text{veryfying exactly one write is committed per version} \\ \quad \forall\, m,\, l \in msgs : \lor\ m.type \quad \neq \text{“VAL”} \\ \qquad\qquad\qquad\ \lor\ l.type \quad \neq \text{“VAL”} \\ \qquad\qquad\qquad\ \lor\ m.version \neq l.version \\ \qquad\qquad\qquad\ \lor\ m.tieBreaker = l.tieBreaker \end{array}$$

$$\begin{array}{lll} GInit \triangleq & \text{The initial predicate} & \\ \quad \land\ msgs & = \{\} & \\ \quad \land\ nodeRcvedAcks & = [n \in G\_NODES \mapsto \{\}] & \\ \quad \land\ nodeState & = [n \in G\_NODES \mapsto \text{“valid”}] & \\ \quad \land\ nodeTS & = [n \in G\_NODES \mapsto [version \quad \mapsto 0, & \\ & \qquad\qquad\qquad\qquad tieBreaker \mapsto & \\ & \qquad\qquad\qquad\qquad\ \text{CHOOSE } k \in G\_NODES : & \\ & \qquad\qquad\qquad\qquad\ \forall\, m \in G\_NODES : k \leq m]] & \end{array}$$

$$\begin{array}{lll} g\_actions\_for\_upd(n,\, newVersion,\, newTieBreaker,\, newState,\, newAcks) \triangleq & & \\ \quad \land\ nodeRcvedAcks' & = [nodeRcvedAcks \quad \text{EXCEPT } ![n] = newAcks] & \\ \quad \land\ nodeState' & = [nodeState \quad \text{EXCEPT } ![n] \quad = newState] & \\ \quad \land\ nodeTS' & = [nodeTS \quad \text{EXCEPT } ![n].version \quad = newVersion, & \\ & \qquad\qquad\qquad\qquad\qquad\ ![n].tieBreaker = newTieBreaker] & \\ \quad \land\ send([type & \mapsto \text{“INV”}, & \\ \qquad sender & \mapsto n, & \\ \qquad version & \mapsto newVersion, & \\ \qquad tieBreaker & \mapsto newTieBreaker]) & \end{array}$$

$$\begin{array}{l} GRead(n) \triangleq \quad \text{Execute a read} \\ \quad \land\ nodeState[n] = \text{“valid”} \\ \quad \land\ \text{UNCHANGED } gvars \end{array}$$

$$\begin{array}{l} GWrite(n) \triangleq \quad \text{Execute a write} \\ \quad \land\ \ nodeState[n] = \text{“valid”} \\ \quad \land\ \ nodeTS[n].version < G\_MAX\_VERSION \quad \text{to configurably terminate the model checking} \\ \quad \land\ \ g\_actions\_for\_upd(n,\, nodeTS[n].version + 1,\, n,\, \text{“write”},\, \{\}) \end{array}$$

$GRcvAck(n) \triangleq$ Process received *Ack*

$\exists\, m \in msgs :$
$\land\ m.type = \text{“ACK”}$
$\land\ nodeState[n] = \text{“write”}$
$\land\ m.sender \neq n$
$\land\ m.sender \notin nodeRcvedAcks[n]$
$\land\ equalTS(m.version,\ m.tieBreaker,$
$nodeTS[n].version,$
$nodeTS[n].tieBreaker)$
$\land\ nodeRcvedAcks' = [nodeRcvedAcks \text{ EXCEPT } ![n] =$
$nodeRcvedAcks[n] \cup \{m.sender\}]$
$\land\ \text{UNCHANGED } \langle msgs,\ nodeTS,\ nodeState\rangle$

$GSendVals(n) \triangleq$ Send validations once acknowledments from all nodes are received

$\land\ receivedAllAcks(n)$
$\land\ nodeState[n] = \text{“write”}$
$\land\ send([type \mapsto \text{“VAL”},$
$version \mapsto nodeTS[n].version,$
$tieBreaker \mapsto nodeTS[n].tieBreaker])$
$\land\ nodeState' = [nodeState \text{ EXCEPT } ![n] = \text{“valid”}]$
$\land\ \text{UNCHANGED } \langle nodeTS,\ nodeRcvedAcks\rangle$

$GCoordinatorActions(n) \triangleq$ Coordinator actions for reads or writes

$\lor\ GRead(n)$
$\lor\ GWrite(n)$
$\lor\ GRcvAck(n)$
$\lor\ GSendVals(n)$

$GRcvInv(n,\ m) \triangleq$ Process received invalidation iff greater ts

$\land\ m.type = \text{“INV”}$
$\land\ m.sender \neq n$
$\land\ greaterTS(m.version,\ m.tieBreaker,$
$nodeTS[n].version,$
$nodeTS[n].tieBreaker)$
$\land\ send([type \mapsto \text{“ACK”},$
$sender \mapsto n,$
$version \mapsto m.version,$
$tieBreaker \mapsto m.tieBreaker])$
$\land\ nodeState' = [nodeState \text{ EXCEPT } ![n] = \text{“invalid”}]$
$\land\ nodeTS' = [nodeTS \text{ EXCEPT } ![n].version = m.version,$
$![n].tieBreaker = m.tieBreaker]$
$\land\ \text{UNCHANGED } \langle nodeRcvedAcks\rangle$

$GRcvVal(n,\ m) \triangleq$ Process received validation iff same ts

$\land\ m.type \quad = \text{“VAL”}$
$\land\ nodeState[n] \neq \text{“valid”}$
$\land\ equalTS(m.version,\ m.tieBreaker,$
$\qquad nodeTS[n].version,$
$\qquad nodeTS[n].tieBreaker)$
$\land\ nodeState' = [nodeState \text{ EXCEPT } ![n] = \text{“valid”}]$
$\land \text{ UNCHANGED } \langle msgs,\ nodeTS,\ nodeRcvedAcks \rangle$

$GFollowerActions(n) \triangleq$ Follower actions for writes
$\quad \exists\, m \in msgs :$
$\qquad \lor\ GRcvInv(n,\ m)$
$\qquad \lor\ GRcvVal(n,\ m)$

$GNext \triangleq$
$\quad \exists\, n \in G\_NODES :$
$\qquad \lor\ GFollowerActions(n)$
$\qquad \lor\ GCoordinatorActions(n)$

$G\_Spec \triangleq GInit \land \Box[GNext]_{gvars}$

THEOREM $G\_Spec \Rightarrow (\Box GTypeOK) \land (\Box GConsistent) \land (\Box GSWMR)$
