MODULE *toinv*

EXTENDS *skeen*

- Total Order: There exists a total order $<$ on all messages that are multicast in an execution trace such that, if process p delivers message m, then for all messages $m' < m$ such that p is one of addresses of message m', p delivers m' before m.
- Total Order can be formalized as the following formula

$$
\begin{array}{l}
\mathit{GlobalTotalOrdering} \triangleq \\
\quad \exists\ \text{ordering} \in [\mathit{McastID} \to 1 \,..\, M] : \\
\qquad \land \forall\, p \in \mathit{Proc} : \forall\, id1,\, id2 \in \mathit{McastID} : \\
\qquad\quad (\ \land \mathit{globalTS}[p][id1] \neq \mathit{TimestampNull} \\
\qquad\quad\ \land \mathit{globalTS}[p][id2] \neq \mathit{TimestampNull} \\
\qquad\quad\ \land \mathit{ordering}[id1] < \mathit{ordering}[id2]\ ) \\
\qquad\qquad \Rightarrow \mathit{Less}(\mathit{globalTS}[p][id1],\, \mathit{globalTS}[p][id2])
\end{array}
$$

- However, *APALACHE* cannot verify *GlobalTotalOrdering* because the initialization of ordering and its corresponding quantifiers.

The conjunction of *ConsistentGlobalTS* and *AsymmetricOrdering* implies Total Order

$$
\begin{array}{l}
\mathit{AsymmetricOrdering} \triangleq \\
\quad \forall\, id1,\, id2 \in \mathit{McastID} : \forall\, p,\, q \in \mathit{Proc} : \\
\qquad (\ \land \mathit{globalTS}[p][id1] \neq \mathit{TimestampNull} \\
\qquad\ \land \mathit{globalTS}[p][id2] \neq \mathit{TimestampNull} \\
\qquad\ \land \mathit{globalTS}[q][id1] \neq \mathit{TimestampNull} \\
\qquad\ \land \mathit{globalTS}[q][id2] \neq \mathit{TimestampNull} \\
\qquad\ \land id1 \neq id2) \\
\qquad\quad \Rightarrow \neg(\mathit{Less}(\mathit{globalTS}[p][id1],\, \mathit{globalTS}[p][id2]) \land \mathit{Less}(\mathit{globalTS}[q][id2],\, \mathit{globalTS}[q][id1]))
\end{array}
$$

$$
\begin{array}{ll}
\mathit{ConsistentGlobalTS} \triangleq & \\
\quad \land \forall\, id \in \mathit{McastID} : \forall\, p,\, q \in \mathit{Proc} : & \text{All addressees of message } id \text{ must agree on its global timestamp.} \\
\qquad (\ \land \mathit{globalTS}[p][id] \neq \mathit{TimestampNull} & \\
\qquad\ \land \mathit{globalTS}[q][id] \neq \mathit{TimestampNull}) & \\
\qquad\quad \Rightarrow \mathit{globalTS}[p][id] = \mathit{globalTS}[q][id] & \\
\quad \land \forall\, id1,\, id2 \in \mathit{McastID} : \forall\, p \in \mathit{Proc} : & \text{Every message has a unique global timestamp.} \\
\qquad (\ \land id1 \neq id2 & \\
\qquad\ \land \mathit{globalTS}[p][id1] \neq \mathit{TimestampNull} & \\
\qquad\ \land \mathit{globalTS}[p][id2] \neq \mathit{TimestampNull}) & \\
\qquad\quad \Rightarrow \mathit{globalTS}[p][id1] \neq \mathit{globalTS}[p][id2] &
\end{array}
$$

$$\mathit{Validity} \triangleq \forall\, p \in \mathit{Proc} : \forall\, id \in \mathit{McastID} : \mathit{delivered}[p][id] \Rightarrow id \in \mathit{mcastedID}$$

If process p is not an addressee of message id, p never issues a local timestamp for id.
Process p issues a local timestampe for message id if and only if it receive a multicast message for id.
The time in every local timestamp cannot greater than the current value of the local clock.
Never issues a local timestamp with *GroupNull* or *TimestampNull*.

The owner of the local timestamp $localTS[p][id]$ must be process p.
Never issues two local timestapms at one time point.

$$
\begin{array}{l}
ValidOwnedLocalTS \triangleq \\
\quad \land (\forall\, id \in McastID : \forall\, p \in Proc \setminus GroupDest[id] : localTS[p][id] = TimestampNull) \\
\quad \land (\forall\, id \in McastID : \forall\, p \in GroupDest[id] : \\
\qquad \land localTS[p][id] = TimestampNull \equiv id \notin rcvdMcastID[p] \\
\qquad \land localTS[p][id].t \leq clock[p] \\
\qquad \land (localTS[p][id].g \neq GroupNull \Rightarrow localTS[p][id].t \neq TimeNull) \\
\qquad \land (localTS[p][id] \neq TimestampNull \\
\qquad\qquad \Rightarrow (\land id \in mcastedID \\
\qquad\qquad\quad \land localTS[p][id].g = p))) \\
\quad \land \forall\, id1,\, id2 \in McastID : \forall\, p \in Proc : \\
\qquad ((\ \land p \in GroupDest[id1] \\
\qquad\quad \land p \in GroupDest[id2] \\
\qquad\quad \land id1 \neq id2 \\
\qquad\quad \land localTS[p][id1] \neq TimestampNull \\
\qquad\quad \land localTS[p][id2] \neq TimestampNull) \\
\qquad\qquad \Rightarrow localTS[p][id1].t \neq localTS[p][id2].t)
\end{array}
$$

Every in-transit message in $inTransit[snder][rcver]$ was sent by process $snder$.
The in-transit proposed message for message id must be sent after the multicast message for message id.

$$
\begin{array}{l}
ValidInTransitMsg \triangleq \\
\quad \land \forall\, snder,\, rcver \in Proc : \forall\, m \in inTransit[snder][rcver] : m.source = snder \\
\quad \land \forall\, snder,\, rcver \in Proc : \forall\, m1,\, m2 \in inTransit[snder][rcver] : \\
\qquad (\quad (\land m1.id = m2.id \\
\qquad\quad \land m1.type = MType \\
\qquad\quad \land m2.type = PType) \\
\qquad\quad \Rightarrow m1.t < m2.t)
\end{array}
$$

- If process p is not an addressee of message id, no processes send a proposal for message id to process p.
- If process p is not an addressee of message id, it never sends a proposal for message id.
- If there exists a proposal for message id from process $snder$, process $snder$ has issued a local timestamp for message m. These timestamps must be the same.
- If there exists a proposal for message id, message id must be multicast before.
- The time in an issued timestamp by process $snder$ cannot greater than the current value of the clock of process $snder$.
- If there exists an in-transit proposed message for message id that is sent to process $rcver$, process $rcver$ has not issued a global timestamp for message id.
- If there exists an in-transit proposed message for message id that is sent from process $snder$, process $snder$ has issued a local timestamp for message id.
- If there exists an in-transit proposed message for message id, message id must be multicast before.
- If there exists an in-transit proposed message for message id that is sent from process $snder$, there exists no in-transit multicast message to process $snder$ such that this multicast message is for message id.

$$
\begin{array}{l}
\mathit{ValidInTransitProposeTS} \triangleq \\
\quad \land (\forall\, id \in \mathit{McastID} : \forall\, rcver \in \mathit{Proc} \setminus \mathit{GroupDest}[id] : \forall\, snder \in \mathit{Proc} : \forall\, m \in \mathit{inTransit}[snder][rcver] : m.id \\
\quad \land (\forall\, id \in \mathit{McastID} : \forall\, snder \in \mathit{Proc} \setminus \mathit{GroupDest}[id] : \forall\, rcver \in \mathit{Proc} : \forall\, m \in \mathit{inTransit}[snder][rcver] : m.id \\
\quad \land (\forall\, id \in \mathit{McastID} : \forall\, rcver \in \mathit{GroupDest}[id] : \forall\, snder \in \mathit{Proc} : \forall\, m \in \mathit{inTransit}[snder][rcver] : \\
\qquad\quad (\land\, m.id = id \\
\qquad\qquad \land\, m.source = snder \\
\qquad\qquad \land\, m.type = \mathit{PType}) \\
\qquad\quad \Rightarrow\ (\land\, m.t = \mathit{localTS}[snder][id].t \\
\qquad\qquad \land\, id \quad \in \mathit{rcvdMcastID}[snder] \\
\qquad\qquad \land\, id \quad \in \mathit{mcastedID})) \\
\quad \land (\forall\, snder,\, rcver \in \mathit{Proc} : \forall\, m \in \mathit{inTransit}[snder][rcver] : m.t \leq \mathit{clock}[m.source] \land m.t > \mathit{TimeNull}) \\
\quad \land (\forall\, snder,\, rcver \in \mathit{Proc} : \forall\, m \in \mathit{inTransit}[snder][rcver] : \\
\qquad m.t = \mathit{PType} \Rightarrow (\land\, \mathit{globalTS}[rcver][m.id] = \mathit{TimestampNull} \\
\qquad\qquad\qquad\qquad \land \neg(\exists\, m1 \in \mathit{inTransit}[\mathit{McastID}[m.id]][snder] : m1.id = m.id \land m1.type = \mathit{MType}) \\
\qquad\qquad\qquad\qquad \land \neg(\exists\, p \in \mathit{Proc} : \exists\, m1 \in \mathit{inTransit}[p][snder] : p = \mathit{McastID}[m.id] \land m1.id = m.id \land m1.type = M \\
\qquad\qquad\qquad\qquad \land\, \mathit{localTS}[m.source][m.id] \neq \mathit{TimestampNull} \\
\qquad\qquad\qquad\qquad \land\, m.id \in \mathit{rcvdMcastID}[m.source])) \\
\quad \land (\forall\, snder,\, rcver \in \mathit{Proc} : \forall\, m \in \mathit{inTransit}[snder][rcver] : \\
\qquad m.t = \mathit{PType} \Rightarrow (\land\, \mathit{localTS}[m.source][m.id].g = m.source \\
\qquad\qquad\qquad\qquad \land\, \mathit{localTS}[m.source][m.id].t = m.t)) \\
\quad \land (\forall\, id \in \mathit{McastID} : \forall\, snder,\, rcver \in \mathit{GroupDest}[id] : \forall\, m \in \mathit{inTransit}[snder][rcver] : \\
\qquad ((m.t = \mathit{PType} \land m.id = id) \\
\qquad\qquad \Rightarrow\ (\forall\, m1 \in \mathit{inTransit}[\mathit{Mcaster}[id]][snder] : m1.type = \mathit{MType} \Rightarrow m1.id \neq id)))
\end{array}
$$

- If process p is not an addressee of message id, it never receives a proposal for message id.
- Every received proposed message for message id must be grouped correctly.
- Every received proposed message for message id from process $snder$ must propose the local timestamp that is issued by process $snder$ for message id.

$$
\begin{array}{l}
\mathit{ValidRcvdProposeTS} \triangleq \\
\quad \land (\forall\, id \in \mathit{McastID} : \forall\, rcver \in \mathit{Proc} \setminus \mathit{GroupDest}[id] : \forall\, snder \in \mathit{Proc} : \\
\qquad \mathit{proposeTS}[rcver][id] = (\{\} <: \{[type \mapsto \mathit{Int},\, t \mapsto \mathit{Int},\, id \mapsto \mathit{Int},\, source \mapsto \mathit{Int}]\})) \\
\quad \land (\forall\, id \in \mathit{McastID} : \forall\, rcver \in \mathit{GroupDest}[id] : \forall\, msg \in \mathit{proposeTS}[rcver][id] : \\
\qquad \land\, msg.t = \mathit{localTS}[msg.source][msg.id].t \\
\qquad \land\, msg.id = id \\
\qquad \land (\forall\, m \in \mathit{inTransit}[msg.source][rcver] : m.id \neq id)) \\
\quad \land (\forall\, id \in \mathit{McastID} : \forall\, rcver \in \mathit{GroupDest}[id] : \forall\, msg \in \mathit{proposeTS}[rcver][id] : \\
\qquad \land\, msg.t = \mathit{localTS}[msg.source][msg.id].t \\
\qquad \land\, msg.source = \mathit{localTS}[msg.source][msg.id].g \\
\qquad \land\, msg.id = id)
\end{array}
$$

Every local clock is bounded with *MaxClock*

$$\mathit{BoundedClock} \triangleq \forall\, p \in \mathit{Proc} : \mathit{clock}[p] \leq \mathit{MaxClock}$$

- The global timestamp for message m cannot be less than any proposed local timestamp for message m.
- The global timestamp for message m must equal some local timestamp for message m.
- If process p is not an addressee of message id, it never issues a global timestamp for message id.
- There exists no global timestamp with *GroupNull* or *TimeNull*.
- The time in every global timestamp cannot greater than the current value of the clock.
- The global timestamp for message id is issued if and only if message id is committed.

$$
\begin{array}{l}
ValidGlobalTS \triangleq \\
\quad \land \forall\, id \in McastID : \forall\, rcver \in GroupDest[id] : \\
\qquad (globalTS[rcver][id] \neq TimestampNull \\
\qquad\quad \equiv (\land \forall\, snder \quad \in GroupDest[id] : \exists\, m \in proposeTS[rcver][id] : \\
\qquad\qquad\quad (\land m.source = snder \\
\qquad\qquad\qquad \land \lor m.t < globalTS[rcver][id].t \\
\qquad\qquad\qquad\quad \lor \land m.t = globalTS[rcver][id].t \\
\qquad\qquad\qquad\qquad \land m.source \leq globalTS[rcver][id].g)) \\
\qquad\qquad \land \exists\, snder \in GroupDest[id] : \exists\, m \in proposeTS[rcver][id] : \\
\qquad\qquad\quad (\land globalTS[rcver][id].t = m.t \\
\qquad\qquad\qquad \land globalTS[rcver][id].g = m.source)) \\
\quad \land \forall\, id \in McastID : \forall\, rcver \in Proc \setminus GroupDest[id] : globalTS[rcver][id] = TimestampNull \\
\quad \land \forall\, id \in McastID : \forall\, rcver \in GroupDest[id] : globalTS[rcver][id].g \neq GroupNull \Rightarrow globalTS[rcver][id].t \neq T \\
\quad \land \forall\, id \in McastID : \forall\, p \in Proc \setminus GroupDest[id] : globalTS[p][id] = TimestampNull \\
\quad \land \forall\, id \in McastID : \forall\, p \in GroupDest[id] : \\
\qquad globalTS[p][id] \neq TimestampNull \Rightarrow \exists\, q \in GroupDest[id] : localTS[q][id] = globalTS[p][id] \\
\quad \land \forall\, id \in McastID : \forall\, rcver \in GroupDest[id] : globalTS[rcver][id].g \neq GroupNull \Rightarrow globalTS[rcver][id].t \leq c \\
\quad \land \forall\, id \in McastID : \forall\, p \in Proc : globalTS[p][id] \neq TimestampNull \equiv phase[p][id] = Committed
\end{array}
$$

Process p sets the status of message id to *Start* iff it has not issued a local timestamp for message id.
If process p commits message id, it has received at least one proposal for message id.
If process p commits message id, it has not issued any global timestamp for message id.

$$
\begin{array}{l}
ValidPhase \triangleq \\
\quad \forall\, p \in Proc : \forall\, id \in McastID : \\
\qquad (\land p \notin GroupDest[id] \Rightarrow phase[p][id] = Start \\
\qquad\quad \land phase[p][id] = Start \equiv localTS[p][id] = TimestampNull \\
\qquad\quad \land phase[p][id] = Proposed \Rightarrow (localTS[p][id] \neq TimestampNull \land id \in rcvdMcastID[p]) \\
\qquad\quad \land phase[p][id] = Committed \equiv (\forall\, q \in GroupDest[id] : \exists\, m \in proposeTS[p][id] : m.source = q) \\
\qquad\quad \land (localTS[p][id] \neq TimestampNull \land id \in rcvdMcastID[p]) \Rightarrow phase[p][id] \in \{Proposed,\ Committed\})
\end{array}
$$

Message id can be delivered to process p if and only if process p has issued a global timestamp for message id
and the local timestamps of all proposed message at process p must be greater than the global timestamp of message id.

$$
\begin{array}{l}
ValidDelivery \triangleq \\
\quad \forall\, p \in Proc : \forall\, id \in McastID : \\
\qquad delivered[p][id] \\
\qquad\quad \Rightarrow (\land globalTS[p][id] \neq TimestampNull \\
\qquad\qquad \land phase[p][id] = Committed
\end{array}
$$

$$\land \forall\, mid \in rcvdMcastID[p] :$$
$$\quad phase[p][mid] = Proposed \Rightarrow Less(globalTS[p][id],\, localTS[p][mid]))$$

Every in-transit message has an unique timestamp.
If process *snder* has sent a proposal for message *id*, no in-transit message to process *p* is a multicast message for message *id*.

$$UniqueMsg \triangleq$$
$$\land (\forall\, snder,\, rcver \in Proc : \forall\, m1,\, m2 \in inTransit[snder][rcver] :$$
$$\quad \land (m1.type = m2.type \land m1.id = m2.id) \Rightarrow m1.t = m2.t$$
$$\quad \land (m1.type = m2.type \land m1.t = m2.t) \Rightarrow m1.id = m2.id$$
$$\quad \land (m1.id = m2.id \land m1.t = m2.t) \Rightarrow m1.type = m2.type)$$
$$\land (\forall\, snder,\, rcver \in Proc : \forall\, m \in inTransit[snder][rcver] :$$
$$\quad m.type = PType \Rightarrow \neg(\exists\, m1 \in inTransit[Mcaster[m.id]][snder] :$$
$$\quad\quad \land m1.id = m.id$$
$$\quad\quad \land m1.source = Mcaster[m.id]$$
$$\quad\quad \land m1.type = MType))$$

- If process *p* is not an addressee of message *id*, it never receives a multicast message for message *id*.
- Every multicast message for message *id* must be multicast by its multicaster.
- If there exists a multicast message for message *id*, message *id* must be multicast before.
- The timestamp of every proposed message from process *snder* cannot be greater the local clock of process *snder*, and must be not 0.
- If there exists an in-transit multicast message for message *id* to process *rcver*, process *rcver* has not issued neither local timestamp nor global timestamp for message *id*.

$$ValidInTransitMcast \triangleq$$
$$\land \forall\, snder,\, rcver \in Proc : \forall\, id \in McastID : \forall\, m \in inTransit[snder][rcver] :$$
$$\quad (m.type = MType \land m.id = id) \Rightarrow (snder = Mcaster[id] \land id \in mcastedID)$$
$$\land \forall\, snder,\, rcver \in Proc : \forall\, m \in inTransit[snder][rcver] :$$
$$\quad m.type = MType \Rightarrow m.source = Mcaster[m.id]$$
$$\land \forall\, snder,\, rcver \in Proc : \forall\, m \in inTransit[snder][rcver] :$$
$$\quad m.type = MType \Rightarrow m.id \in mcastedID$$
$$\land \forall\, snder,\, rcver \in Proc : \forall\, m \in inTransit[snder][rcver] :$$
$$\quad m.t \leq clock[m.source] \land m.t > TimeNull$$
$$\land \forall\, mcaster,\, rcver \in Proc : \forall\, m \in inTransit[mcaster][rcver] :$$
$$\quad m.type = MType \Rightarrow (\land \neg(\exists\, q \in Proc : \exists\, m1 \in inTransit[rcver][q] : m1.source = rcver \land m1.id = m.id \land$$
$$\quad\quad \land localTS[rcver][m.id] = TimestampNull$$
$$\quad\quad \land \forall\, p \in GroupDest[m.id] : globalTS[p][m.id] = TimestampNull)$$

$$TypeOK \triangleq$$
$$\land clock \in [Proc \to Time \cup \{TimeNull\}]$$
$$\land localTS \in [Proc \to [McastID \to TimestampSet]]$$

$\land\ globalTS \in [Proc \to [McastID \to TimestampSet]]$
$\land\ phase \in [Proc \to [McastID \to \{Start,\ Proposed,\ Committed\}]]$
$\land\ rcvdMcastID \in [Proc \to \text{SUBSET } McastID]$
$\land\ mcastedID \in \text{SUBSET } McastID$
$\land\ inTransit \in [Proc \to [Proc \to \text{SUBSET } InTransitMsgSet]]$
$\land\ delivered \in [Proc \to [McastID \to \text{BOOLEAN }]]$
$\land\ proposeTS \in [Proc \to [McastID \to \text{SUBSET } ProposeMsgSet]]$
$\land\ dCntr \in [Proc \to [McastID \to \{0, 1\}]]$

$Integrity \triangleq$
$\forall\, id \in McastID : \forall\, p \in Proc :$
$\land\ delivered[p][id] \equiv dCntr[p][id] = 1$
$\land\ \neg delivered[p][id] \equiv dCntr[p][id] = 0$

$IndInv \triangleq$
$\land\ TypeOK$
$\land\ Integrity$
$\land\ Validity$
$\land\ ValidInTransitMsg$
$\land\ AsymmetricOrdering$
$\land\ ConsistentGlobalTS$
$\land\ ValidOwnedLocalTS$
$\land\ ValidInTransitProposeTS$
$\land\ ValidRcvdProposeTS$
$\land\ BoundedClock$
$\land\ ValidGlobalTS$
$\land\ ValidDelivery$
$\land\ ValidPhase$
$\land\ ValidInTransitMcast$
$\land\ UniqueMsg$

* Modification History
* Last modified *Mon Mar* 22 12:06:38 *CET* 2021 by *tran*
* Created *Tue Mar* 16 08:59:43 *CET* 2021 by *tran*