MODULE *syncCon*1

zheng kai 50247576
xuyin bo 50102922
Synchronized Consensus

EXTENDS *Integers*, *Sequences*, *FiniteSets*, *TLC*
CONSTANTS N, $FAILNUM$
ASSUME $N \leq 5 \land 0 \leq FAILNUM \land FAILNUM \leq 4$
$Nodes \triangleq 1 \,..\, N$

```
--algorithm syncCon1{
    variables FailNum = FAILNUM ;                 Initialization block
              up = [n  ∈ Nodes ↦ TRUE] ;          Nodes are up
              pt = [n  ∈ Nodes ↦ 0] ;             Nodes are at round 0
              t = [n ∈ Nodes ↦ FALSE] ;           Nodes are not terminated
              d = [n ∈ Nodes ↦ −1] ;              Nodes are not decided
              mb = [n ∈ Nodes ↦ {}] ;             Nodes have mailbox as emptyset

    define {
        SetMin(S) ≜ CHOOSE i ∈ S : ∀ j ∈ S : i ≤ j    choose the smallest value in S
     }

    macro MaybeFail( ) {
        if ( FailNum > 0 ∧ up[self] )
        {
            either
            { up[self] := FALSE ; FailNum := FailNum − 1 ; }
            or skip ;
         }
     }

    fair process ( n ∈ Nodes )
    variables v = 0, Q = {} ;
    {
            P: if ( up[self] ) {
                    v := self ;
                    Q := Nodes ;
            PS: while ( up[self] ∧ Q ≠ {} ) {
                    with ( p ∈ Q ) {
                            MaybeFail() ;          the node can crash when sending message
                            mb[p] := mb[p] ∪ {v} ;         put value v into the node's mailbox
                            Q :=  Q \ {p} ;          delete p out of the set Q
                     } ;
                } ;
                if ( up[self] ) pt[self] := pt[self] + 1 ;
           PR: await (up[self]) ∧ (∀ i ∈ Nodes : (pt[i] = 1 ∨ up[i] = FALSE)) ;   wait for all nodes receive others' me
                d[self] := SetMin(mb[self]) ;      use SetMin() function to decide the smallest value in collection
```

```
            t[self] := TRUE;                    when the node has decide the smallest value, it's done.
        } ;
    }
}
```

BEGIN TRANSLATION

VARIABLES $FailNum$, up, pt, t, d, mb, pc

define statement

$SetMin(S) \triangleq$ CHOOSE $i \in S : \forall j \in S : i \leq j$

VARIABLES v, Q

$vars \triangleq \langle FailNum, up, pt, t, d, mb, pc, v, Q \rangle$

$ProcSet \triangleq (Nodes)$

$Init \triangleq$ Global variables
$\land FailNum = FAILNUM$
$\land up = [n \in Nodes \mapsto \text{TRUE}]$
$\land pt = [n \in Nodes \mapsto 0]$
$\land t = [n \in Nodes \mapsto \text{FALSE}]$
$\land d = [n \in Nodes \mapsto -1]$
$\land mb = [n \in Nodes \mapsto \{\}]$
Process n
$\land v = [self \in Nodes \mapsto 0]$
$\land Q = [self \in Nodes \mapsto \{\}]$
$\land pc = [self \in ProcSet \mapsto \text{“P”}]$

$P(self) \triangleq \land pc[self] = \text{“P”}$
$\land$ IF $up[self]$
THEN $\land v' = [v \text{ EXCEPT } ![self] = self]$
$\land Q' = [Q \text{ EXCEPT } ![self] = Nodes]$
$\land pc' = [pc \text{ EXCEPT } ![self] = \text{“PS”}]$
ELSE $\land pc' = [pc \text{ EXCEPT } ![self] = \text{“Done”}]$
$\land$ UNCHANGED $\langle v, Q \rangle$
$\land$ UNCHANGED $\langle FailNum, up, pt, t, d, mb \rangle$

$PS(self) \triangleq \land pc[self] = \text{“PS”}$
$\land$ IF $up[self] \land Q[self] \neq \{\}$
THEN $\land \exists p \in Q[self] :$
$\land$ IF $FailNum > 0 \land up[self]$
THEN $\land \lor \land up' = [up \text{ EXCEPT } ![self] = \text{FALSE}]$
$\land FailNum' = FailNum - 1$
$\lor \land \text{TRUE}$
$\land$ UNCHANGED $\langle FailNum, up \rangle$
ELSE $\land$ TRUE

$$
\begin{array}{l}
\qquad\qquad\qquad\qquad\qquad\qquad \land \text{ UNCHANGED } \langle FailNum,\ up\rangle \\
\qquad\qquad\qquad\qquad\qquad \land\ mb' = [mb \text{ EXCEPT } ![p] = mb[p] \cup \{v[self]\}] \\
\qquad\qquad\qquad\qquad\qquad \land\ Q' = [Q \text{ EXCEPT } ![self] = Q[self] \setminus \{p\}] \\
\qquad\qquad\qquad\qquad \land\ pc' = [pc \text{ EXCEPT } ![self] = \text{“PS”}] \\
\qquad\qquad\qquad\qquad \land\ pt' = pt \\
\qquad\qquad\qquad \text{ELSE } \land \text{ IF } up[self] \\
\qquad\qquad\qquad\qquad\qquad \text{THEN } \land\ pt' = [pt \text{ EXCEPT } ![self] = pt[self] + 1] \\
\qquad\qquad\qquad\qquad\qquad \text{ELSE } \land \text{ TRUE} \\
\qquad\qquad\qquad\qquad\qquad\qquad\quad \land\ pt' = pt \\
\qquad\qquad\qquad\qquad \land\ pc' = [pc \text{ EXCEPT } ![self] = \text{“PR”}] \\
\qquad\qquad\qquad\qquad \land \text{ UNCHANGED } \langle FailNum,\ up,\ mb,\ Q\rangle \\
\qquad\qquad\qquad \land \text{ UNCHANGED } \langle t,\ d,\ v\rangle
\end{array}
$$

$$
\begin{array}{lll}
PR(self) & \triangleq & \land\ pc[self] = \text{“PR”} \\
& & \land\ (up[self]) \land (\forall\, i \in Nodes : (pt[i] = 1 \lor up[i] = \text{FALSE})) \\
& & \land\ d' = [d \text{ EXCEPT } ![self] = SetMin(mb[self])] \\
& & \land\ t' = [t \text{ EXCEPT } ![self] = \text{TRUE}] \\
& & \land\ pc' = [pc \text{ EXCEPT } ![self] = \text{“Done”}] \\
& & \land \text{ UNCHANGED } \langle FailNum,\ up,\ pt,\ mb,\ v,\ Q\rangle
\end{array}
$$

$$
n(self) \triangleq P(self) \lor PS(self) \lor PR(self)
$$

$$
\begin{array}{lll}
Next & \triangleq & (\exists\, self \in Nodes : n(self)) \\
& & \quad \lor \text{ Disjunct to prevent deadlock on termination} \\
& & \quad ((\forall\, self \in ProcSet : pc[self] = \text{“Done”}) \land \text{ UNCHANGED } vars)
\end{array}
$$

$$
\begin{array}{lll}
Spec & \triangleq & \land\ Init \land \Box[Next]_{vars} \\
& & \land\ \forall\, self \in Nodes : \mathrm{WF}_{vars}(n(self))
\end{array}
$$

$$
Termination \triangleq \Diamond(\forall\, self \in ProcSet : pc[self] = \text{“Done”})
$$

END TRANSLATION

$$
Inv \triangleq (\exists\, i \in Nodes : \neg t[i]) \lor (\forall\, l,\ m \in Nodes : \neg up[l] \lor \neg up[m] \lor d[l] = d[m])
$$

---

* 1.2 Model-check safety properties with TLA+

* First, we assume no crash. It means after this round, every node sends it value to others and decide the smallest value.

* So at the model check, we set $FailNum = 0$, $Nodes = 5$, choose the termination property and there is no error.

* Then change the $FailNum$ to 1,2,3,4, Node still equals 5, choose the termination property, then error happens.

* For examle, $node1$ crash when only $node2$ receive it's value. Assume others node don't crash and program still run,

* Then $node2$ set the min value $= 1$, but $node3$,$node4$,$node5$ can not set min value $= 1$. So this consensus protocol algorithm does't work.

* And at this algorithm, when $node1$ crash, $up[1]$ will be False. So the termination property will be violated.

* So the agreement property satisfied when $FailNum = 0$, unsatisfied when $FailNum > 0$
* Then test an invariant property $Inv \triangleq (\exists\, i \in Nodes : \neg t[i]) \lor (\forall\, l,m \in Nodes : \neg up[l] \lor \neg up[m] \lor d[l] = d[m])$
* An invariant property should be satisfied at every situations. So check the invariant property when $FailNum$ from 0 to 4. It maintains right.

* Modification History
* Last modified *Tue Oct* 24 15:24:46 *EDT* 2017 by *xinboyu*
* Last modified *Tue Oct* 24 11:58:03 *EDT* 2017 by kz-*pc*
* Created Sat *Oct* 21 11:02:18 *EDT* 2017 by kz-*pc*