MODULE *TwoStepOptimiticBroadcast*

EXTENDS *FiniteSets*, *Integers*, *TLC*

CONSTANTS
  P the set of parties
, $Faulty$ the set of faulty parties
, V the set of value that may be broadcast

$N \triangleq Cardinality(P)$
$F \triangleq Cardinality(Faulty)$

ASSUME $Faulty \subseteq P \land \ N > 3 * F$

Integer division, rounded up:
$CeilDiv(a, b) \triangleq$ IF $a\%b = 0$ THEN $a \div b$ ELSE $(a \div b) + 1$

The set of possible messages in the network:
$Message \triangleq$
  $[src : P,\ dst : P,\ type : \{\text{“proposal”}, \text{“echo”}, \text{“vote”}, \text{“ready”}\},\ val : V]$

```
--algorithm Broadcast{
  variables
    broadcaster ∈ P,   the distinguised broadcaster party; could be faulty or not
    msgs = {} ;   the set of sent messages
  define {
    Msgs(self, v, type) ≜ {m ∈ msgs : m.type = type ∧ m.val = v ∧ m.dst = self}
    Echos(self, v) ≜ Msgs(self, v, "echo")
    Votes(self, v) ≜ Msgs(self, v, "vote")
    Readys(self, v) ≜ Msgs(self, v, "ready")
   }
  macro SendAll( type, value ) {
    msgs := msgs ∪ {[src ↦ self, dst ↦ d, type ↦ type, val ↦ value] : d ∈ P}
   }
  fair process ( correctParty ∈ P \ Faulty )
    variable delivered = ⟨⟩ ;   the delivered value
  {
l0:   while ( TRUE ) with ( v ∈ V ) {
        either {   send proposal
          when self = broadcaster ;
          when ∀ m ∈ msgs : ¬(m.src = self ∧ m.type = "proposal") ;
          with ( proposal ∈ V )
          SendAll("proposal", proposal)
         }
        or {   send echo
          when ∀ m ∈ msgs : ¬(m.src = self ∧ m.type = "echo") ;
          await [src ↦ broadcaster, dst ↦ self, type ↦ "proposal", val ↦ v] ∈ msgs ;
```

```
            SendAll("echo", v)
        }
        or {  fast delivery
            await Cardinality({m ∈ Echos(self, v) : m.src ≠ broadcaster}) ≥ CeilDiv(N + 2 * F − 2,
            delivered := v
        }
        or {  send vote
            when ∀ m ∈ msgs : ¬(m.src = self ∧ m.type = "vote");
            await Cardinality({m ∈ Echos(self, v) : m.src ≠ broadcaster}) ≥ CeilDiv(N, 2);
            SendAll("vote", v)
        }
        or {  send ready
            when ∀ m ∈ msgs : ¬(m.src = self ∧ m.type = "ready");
            await
                ∨ Cardinality({m ∈ Echos(self, v) : m.src ≠ broadcaster}) ≥ CeilDiv(N + F − 1, 2)
                ∨ Cardinality({m ∈ Votes(self, v) : m.src ≠ broadcaster}) ≥ CeilDiv(N + F − 1, 2)
                ∨ Cardinality(Readys(self, v)) ≥ F + 1;
            SendAll("ready", v)
        }
        or {  slow delivery
            await Cardinality(Readys(self, v)) ≥ 2 * F + 1;
            delivered := v
        }
    }
}
process ( faultyParty ∈ Faulty ) {
    faulty parties may send arbitrary messages:
l1:  while ( TRUE )
    with ( v ∈ V, t ∈ {"proposal", "echo", "vote", "ready"}, d ∈ P \ Faulty ) {
        msgs := msgs ∪ {[src ↦ self, dst ↦ d, type ↦ t, val ↦ v]}
    }
}
}
```

Correctness properties:

$$
\begin{aligned}
&\mathit{TypeOK} \triangleq \\
&\quad \land\ \forall\, m \in \mathit{msgs} : m \in \mathit{Message} \\
&\quad \land\ \forall\, p \in P \setminus \mathit{Faulty} : \mathit{delivered}[p] \in \{\langle\rangle\} \cup V
\end{aligned}
$$

$$
\begin{aligned}
&\mathit{ReadySame} \triangleq \forall\, m1, m2 \in \mathit{msgs} : \\
&\quad \land\ m1.\mathit{src} \notin \mathit{Faulty} \land m2.\mathit{src} \notin \mathit{Faulty} \\
&\quad \land\ m1.\mathit{type} = \text{“ready”} \land m2.\mathit{type} = \text{“ready”} \\
&\quad \Rightarrow \\
&\qquad m1.\mathit{val} = m2.\mathit{val}
\end{aligned}
$$

to find an execution in which all correct parties deliver:

$$
\begin{array}{l}
\mathit{Falsy} \triangleq \neg(\\
\quad \forall\, p \in P \setminus \mathit{Faulty} : \mathit{delivered}[p] \neq \langle\rangle \\
)
\end{array}
$$

$$
\begin{array}{l}
\mathit{Agreement} \triangleq \forall\, p1,\, p2 \in P \setminus \mathit{Faulty} : \\
\quad \mathit{delivered}[p1] \neq \langle\rangle \land \mathit{delivered}[p2] \neq \langle\rangle \Rightarrow \mathit{delivered}[p1] = \mathit{delivered}[p2]
\end{array}
$$

$$
\begin{array}{l}
\mathit{Liveness} \triangleq \\
\quad \land\ (\mathit{broadcaster} \notin \mathit{Faulty} \Rightarrow \forall\, p \in P \setminus \mathit{Faulty} : \\
\qquad \Diamond(\exists\, v \in V : \\
\qquad\quad \land\ [\mathit{src} \mapsto \mathit{broadcaster},\, \mathit{dst} \mapsto p,\, \mathit{type} \mapsto \text{“proposal”},\, \mathit{val} \mapsto v] \in \mathit{msgs} \\
\qquad\quad \land\ \mathit{delivered}[p] = v)) \\
\quad \land\ \Box((\exists\, p \in P \setminus \mathit{Faulty} : \mathit{delivered}[p] \neq \langle\rangle) \Rightarrow \forall\, p \in P \setminus \mathit{Faulty} : \Diamond(\mathit{delivered}[p] \neq \langle\rangle))
\end{array}
$$