Modelling and validating distributed systems with TLA+

Carla Ferreira
Universidade NOVA de Lisboa
30th April 2019
Best effort broadcast

- For any two correct processes $i$ and $j$, every message broadcast by $i$ is eventually delivered by $j$.
- No message is delivered more than once.
- If a correct process $j$ delivers a message $m$, then $m$ was broadcast to $j$ by some process $i$. 
Best effort broadcast

MODULE BestEffortBroadcast
EXTENDS Naturals, FiniteSets

CONSTANTS
Process, (* set of processes *)
MaxBroadcasts (* maximum number of broadcast messages *)

ASSUME
\( \forall \text{Process} \ # \ {}\)
\( \forall \text{MaxBroadcasts} > 0 \)

VARIABLES
comm, (* point-to-point communication between processes *)
broadcasted, (* global set of broadcasted messages *)
delivered, (* global set of delivered messages *)
pstate, (* process local state *)
alive, (* set of alive/active processes *)
correct (* set of correct processes *)
Best effort broadcast

MODULE BestEffortBroadcast

VARIABLES
  comm, (* point-to-point communication between processes *)
  broadcasted, (* global set of broadcasted messages *)
  delivered, (* global set of delivered messages *)
  pstate, (* process local state *)
  alive, (* set of alive/active processes *)
  correct (* set of correct processes *)

Message == [sdr : Process, mid : 0..MaxBroadcasts] (* broadcast message *)


TypeInv ==
  \ comm \in [Process -> SUBSET Message]
  \ broadcasted \in SUBSET Message
  \ delivered \in SUBSET MessageDel
  \ pstate \in [Process -> Nat] (* local state stores the message count *)
  \ alive \in SUBSET Process
  \ correct \in SUBSET Process

Init ==
  \ comm = [p \in Process |-> {}]
  \ broadcasted = {}
  \ delivered = {}
  \ pstate = [p \in Process |-> 0]
  \ alive = Process
  \ correct \in SUBSET Process
broadcasted: \{ \} 

delivered: \{ \} 

alive: \{ P1, P2, P3 \} 

correct: \{ P2 \} 

\[
pstate[P1] = 0 
\]

\[
pstate[P2] = 0 
\]

\[
pstate[P3] = 0 
\]

\[
\text{P1} \quad \text{P2} \quad \text{P3} 
\]

\[
\text{comm}[P1] \quad \text{comm}[P2] \quad \text{comm}[P3] 
\]
Best effort broadcast

MODULE BestEffortBroadcast

TypeInv ==
∀ comm \in [Process -> SUBSET Message] ∧ broadcasted \in SUBSET Message
∀ delivered \in SUBSET MessageDel ∧ pstate \in [Process -> Nat]
∀ alive \in SUBSET Process. ∧ correct \in SUBSET Process

broadcast(p) ==
∀ p \in alive
∀ Cardinality(broadcasted) < MaxBroadcasts
∀ LET msg == [sdr |-> p, mid |-> pstate[p]]
    IN
    ∀ broadcasted' = broadcasted \union { msg }
    ∀ comm' = [q \in Process |-> comm[q] \union { msg }]
    ∀ pstate' = [pstate EXCEPT ![p] = pstate[p] + 1]
    ∀ UNCHANGED<<delivered,alive,correct>>
broadcasted \{ \} 
delivered \{ \} 
alive \{ P1, P2, P3 \} 
correct \{ P2 \} 

broadcast(P2)

pstate[P1] = 0

pstate[P2] = 0

pstate[P3] = 0
broadcasted \{ m1 \}
delivered \{ \}
alive \{ P1, P2, P3 \}
correct \{ P2 \}

broadcast(P2)

\begin{align*}
\text{pstate}[P1] &= 0 \\
\text{pstate}[P2] &= 1 \\
\text{pstate}[P3] &= 0
\end{align*}
broadcasted { m1 }
delivered { }
alive { P1, P2, P3 }
correct { P2 }

broadcast(P1)

pstate[P1] = 0

comm[P2] {m1} comm[P1] {m1} comm[P3] {m1}

pstate[P2] = 1

pstate[P3] = 0
broadcasted: \{ m1, m2 \}
delivered: \{ \}
alive: \{ P1, P2, P3 \}
correct: \{ P2 \}

broadcast(P1)

\text{pstate}[P1] = 1

\text{comm}[P1] = \{ m1, m2 \}
\text{comm}[P2] = \{ m2, m1 \}
\text{comm}[P3] = \{ m1, m2 \}
\text{pstate}[P2] = 1
\text{pstate}[P3] = 0
Best effort broadcast

---------------------------------------- MODULE BestEffortBroadcast ----------------------------------------

TypeInv ==

\[\land \text{comm} \in [\text{Process} \rightarrow \text{SUBSET Message}] \land \text{broadcasted} \in \text{SUBSET Message}\]
\[\land \text{delivered} \in \text{SUBSET MessageDel} \land \text{pstate} \in [\text{Process} \rightarrow \text{Nat}]\]
\[\land \text{alive} \in \text{SUBSET Process.} \land \text{correct} \in \text{SUBSET Process}\]

receive(p) ==

\[\land p \in \text{alive}\]
\[\land \exists m \in \text{comm}[p]: \]
\[\land \text{comm'} = [\text{comm} \ \text{EXCEPT} \ ![p] = \text{comm}[p] \ \{m\}]\]
\[\land \text{delivered'} = \text{delivered} \ \text{union} \ \{[\text{rcv} \rightarrow p, \text{msg} \rightarrow m]\}\]
\[\land \text{UNCHANGED}<<\text{broadcasted,alive,pstate,correct>>}\]
receive(P3)

\[ \text{comm}[P2] = \{m2, m1\} \]

\[ \text{pstate}[P1] = 1 \]

\[ \text{comm}[P1] = \{m1, m2\} \]

\[ \text{comm}[P3] = \{m1, m2\} \]

\[ \text{pstate}[P3] = 0 \]

\[ \text{broadcasted} \{m1, m2\} \]

\[ \text{delivered} \{\} \]

\[ \text{alive} \{P1, P2, P3\} \]

\[ \text{correct} \{P2\} \]
receive(P3)

| broadcasted  | \{ m1, m2 \} |
| delievered   | \{ P3_m2 \} |
| alive        | \{ P1, P2, P3 \} |
| correct      | \{ P2 \} |

pstate[P1] = 1
pstate[P2] = 1
pstate[P3] = 0

comm[P1] \{m1,m2\}
comm[P2] \{m2,m1\}
comm[P3] \{m1\}
receive(P2)

P1

P2

P3

pstate[P1] = 1

pstate[P2] = 1

pstate[P3] = 0

comm[P2] = \{m2, m1\}

comm[P1] = \{m1, m2\}

comm[P3] = \{m1\}

broadcasted \{ m1, m2 \}
delivered \{ P3_m2 \}
alive \{ P1, P2, P3 \}
correct \{ P2 \}
receive(P2)

pstate[P1] = 1

comm[P2] = \{ m2 \}

pstate[P2] = 1

comm[P1] = \{ m1, m2 \}

comm[P3] = \{ m1 \}

pstate[P3] = 0

broadcasted = \{ m1, m2 \}
delivered = \{ P2_m1, P3_m2 \}
alive = \{ P1, P2, P3 \}
correct = \{ P2 \}
Best effort broadcast

----------------------------- MODULE BestEffortBroadcast -----------------------------

TypeInv ==
\[
\forall \text{comm} \in \text{SUBSET Process} \rightarrow \text{Message} \quad \forall \text{broadcasted} \in \text{SUBSET Message} \\
\forall \text{delivered} \in \text{SUBSET MessageDel} \quad \forall \text{pstate} \in \text{SUBSET Process} \\
\forall \text{alive} \in \text{SUBSET Process} \quad \forall \text{correct} \in \text{SUBSET Process}
\]

fail(p) ==
\[
\forall p \in \text{alive} \\
\forall p \notin \text{correct} \\
\forall \text{alive}' = \text{alive} \setminus \{p\} \\
\forall \text{UNCHANGED} \ll<\text{comm, broadcasted, delivered, pstate, correct}>\]

16
fail(P3)
fail(P3)

<table>
<thead>
<tr>
<th>State</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>broadcasted</td>
<td>{ m1, m2 }</td>
</tr>
<tr>
<td>delivered</td>
<td>{ P2_m1, P3_m2 }</td>
</tr>
<tr>
<td>alive</td>
<td>{ P1, P2 }</td>
</tr>
<tr>
<td>correct</td>
<td>{ P2 }</td>
</tr>
</tbody>
</table>

Pstate\[P1\] = 1
Pstate\[P2\] = 1
Pstate\[P3\] = 0
Best effort broadcast

--- MODULE BestEffortBroadcast ---

Next == \E p \in Process : broadcast(p) \lor receive(p) \lor fail(p)

Spec == \land Init
  \land [] [Next]_<<comm, broadcasted, delivered, pstate, alive, correct>>
broadcast\(p\) ==
\[\begin{align*}
&\land p \in \text{alive} \\
&\land \text{Cardinality(broadcasted)} < \text{MaxBroadcasts} \\
&\land \text{LET } \text{msg} == \left[ sdr \mapsto p, \text{mid} \mapsto \text{pstate}[p] \right] \\
&\quad \text{IN} \\
&\quad \land \text{broadcasted}' = \text{broadcasted} \ union \{ \text{msg} \} \\
&\quad \land \text{comm}' = \left[ q \in \text{Process} \mapsto \text{comm}[q] \ union \{ \text{msg} \} \right] \\
&\quad \land \text{pstate}' = \left[ \text{pstate}_{\text{EXCEPT}} ![p] = \text{pstate}[p] + 1 \right] \\
&\land \text{UNCHANGED}<<\text{delivered,alive,correct}}\]
\]
receive\(p\) ==
\[\begin{align*}
&\land p \in \text{alive} \\
&\land \exists m \in \text{comm}[p] : \\
&\quad \land \text{comm}' = \left[ \text{comm}_{\text{EXCEPT}} ![p] = \text{comm}[p] \setminus \{ m \} \right] \\
&\quad \land \text{delivered}' = \text{delivered} \ union \{ [\text{rcv} \mapsto p, \text{msg} \mapsto m] \} \\
&\land \text{UNCHANGED}<<\text{broadcasted,alive,correct}}\]
\]
fail\(p\) ==
\[\begin{align*}
&\land p \in \text{alive} \\
&\land p \notin \text{correct} \\
&\land \text{alive}' = \text{alive} \ setminus \{ p \} \\
&\land \text{UNCHANGED}<<\text{comm,broadcasted,delivered,pstate,correct}}\]
\]

Is this a “good” specification of the protocol?
Best effort broadcast (2\textsuperscript{nd} try)

\begin{verbatim}
MODULE BestEffortBroadcast
EXTENDS Naturals, FiniteSets

CONSTANTS
  Process, (* set of processes *)
  MaxBroadcasts (* maximum number of broadcast messages *)

ASSUME
  \land Process \# {}
  \land MaxBroadcasts > 0

VARIABLES
  comm, (* point-to-point communication between processes *)
  broadcasted, (* global set of broadcasted messages *)
  delivered, (* global set of delivered messages *)
  pstate, (* process local state *)
  alive, (* set of alive/active processes *)
  correct (* set of correct processes *)
\end{verbatim}

Exactly the same as before!
Best effort broadcast (2\textsuperscript{nd} try)

--- MODULE BestEffortBroadcast ---

Message == [sdr : Process, mid : 0..MaxBroadcasts] (* broadcast message *)


(* messages broadcasted *)

(* messages sent *)

(* broadcast messages count *)

VARIABLES

comm, broadcasted, delivered, pstate, alive, correct

TypeInv ==

\( \forall \) comm \in [Process \to SUBSET Message]
\( \forall \) broadcasted \in SUBSET Message
\( \forall \) delivered \in SUBSET MessageDel
\( \forall \) pstate \in [Process \to LocalState]
\( \forall \) alive \in SUBSET Process
\( \forall \) correct \in SUBSET Process

Init ==

\( \forall \) comm = [p \in Process |-> {}]
\( \forall \) broadcasted = {}
\( \forall \) delivered = {}

\( \forall \) pstate = [p \in Process |-> [bcast |-> {},
msent |-> [q \in Process |-> {}],
mcount |-> 0]]

\( \forall \) alive = Process
\( \forall \) correct \in SUBSET Process
broadcasted: \{
\}
delivered: \{
\}
alive: \{P1, P2, P3\}
correct: \{P2\}

\begin{align*}
\text{pstate}[P1].bcast &= \{\}\n\text{pstate}[P1].msent &= \{P1\mapsto\{\}, P2\mapsto\{\}, P3\mapsto\{\}\}\n\text{pstate}[P1].mcount &= 0
\end{align*}

\begin{align*}
\text{pstate}[P2].bcast &= \{\}\n\text{pstate}[P2].msent &= \{P1\mapsto\{\}, P2\mapsto\{\}, P3\mapsto\{\}\}\n\text{pstate}[P2].mcount &= 0
\end{align*}

\begin{align*}
\text{pstate}[P3].bcast &= \{\}\n\text{pstate}[P3].msent &= \{P1\mapsto\{\}, P2\mapsto\{\}, P3\mapsto\{\}\}\n\text{pstate}[P3].mcount &= 0
\end{align*}
Best effort broadcast (2\textsuperscript{nd} try)

\[\text{broadcast}(p) ==
\begin{align*}
\text{// } p \in \text{alive} \\
\text{// } \text{Cardinality}(pstate[p].bcast) < \text{MaxBroadcasts} \\
\text{// } \text{LET } \text{msg} == [\text{sdr} \mapsto p, \text{mid} \mapsto pstate[p].mcount] \\
\quad \text{IN} \\
\quad \text{// } \text{broadcasted' } = \text{broadcasted } \cup \{ \text{msg} \} \\
\quad \text{// } \text{pstate' } = [pstate \ \text{EXCEPT} \ !p] = [\text{bcast} \mapsto pstate[p].bcast \cup \{\text{msg}\}, \\
\quad \text{msent} \mapsto pstate[p].msent, \\
\quad \text{mcount} \mapsto pstate[p].mcount + 1] \\
\text{// UNCHANGED} \langle \text{comm,delivered,alive,correct} \rangle
\]
broadcasted: \{ \}  
delivered: \{ \}  
alive: \{ P1, P2, P3 \}  
correct: \{ P2 \}

pstate[P1].bcast = \{ \}  
pstate[P1].msent = \{P1\mapsto\{\}, P2\mapsto\{\}, P3\mapsto\{\}\}  
pstate[P1].mcount = 0

correct: \{ P2 \}

broadcast(P1)

pstate[P2].bcast = \{ \}  
pstate[P2].msent = \{P1\mapsto\{\}, P2\mapsto\{\}, P3\mapsto\{\}\}  
pstate[P2].mcount = 0

pstate[P2].bcast = \{ \}  
pstate[P2].msent = \{P1\mapsto\{\}, P2\mapsto\{\}, P3\mapsto\{\}\}  
pstate[P2].mcount = 0

P1

P2

P3
broadcasted
{ m1 }
delivered
{ }
alive
{ P1, P2, P3 }
correct
{ P2 }

pstate\[P1\].bcast = { m1 }
pstate\[P1\].msent = { P1→{}, P2→{}, P3→{} }
pstate\[P1\].mcount = 1

broadcast(P1)

pstate\[P2\].bcast = { } pstate\[P2\].msent = { P1→{}, P2→{}, P3→{} }
pstate\[P2\].mcount = 0

pstate\[P2\].bcast = { } pstate\[P2\].msent = { P1→{}, P2→{}, P3→{} }
pstate\[P2\].mcount = 0
Best effort broadcast (2\textsuperscript{nd} try)

--------- MODULE BestEffortBroadcast ---------

TypeInv ==
\( \land \text{comm} \in \{\text{Process} \rightarrow \text{SUBSET Message}\} \) \( \land \) \( \text{broadcasted} \in \text{SUBSET Message} \)
\( \land \text{delivered} \in \text{SUBSET MessageDel} \) \( \land \) \( \text{pstate} \in \{\text{Process} \rightarrow \text{LocalState}\} \)
\( \land \text{alive} \in \text{SUBSET Process} \) \( \land \) \( \text{correct} \in \text{SUBSET Process} \)

send(p) ==
\( \land \text{p} \in \text{alive} \)
\( \land \ \exists \text{m} \in \text{pstate[p].bcast}, \text{q} \in \text{Process} : \)
\( \land \text{m} \notin \text{pstate[p].msent[q]} \) (* m was not sent to q *)
\( \land \text{comm'} = \text{comm EXCEPT ![q]} = \text{comm}[q] \cup \{\text{m}\} \)
\( \land \text{pstate'} = \text{pstate EXCEPT ![p]} = \)
\( \text{[bcast} \rightarrow \text{pstate[p].bcast}, \)
\( \text{msent} \rightarrow \text{pstate[p].msent EXCEPT ![q]} = \text{pstate[p].msent[q]} \cup \{\text{m}\}], \)
\( \text{mcount} \rightarrow \text{pstate[p].mcount}] \)
\( \land \) \( \text{UNCHANGED}<<\text{broadcasted,delivered,alive,correct}}>> \)
broadcasted: \{ m1 \}
delivered: \{ \}
alive: \{ P1, P2, P3 \}
correct: \{ P2 \}

\[\text{pstate}[P1].bcast = \{ m1 \} \]
\[\text{pstate}[P1].msent = \{P1 \mapsto \{\}, P2 \mapsto \{\}, P3 \mapsto \{\}\} \]
\[\text{pstate}[P1].mcount = 1\]

\[\text{sent}(P2)\]

\[\text{pstate}[P2].bcast = \{\} \]
\[\text{pstate}[P2].msent = \{P1 \mapsto \{\}, P2 \mapsto \{\}, P3 \mapsto \{\}\} \]
\[\text{pstate}[P2].mcount = 0\]

\[\text{pstate}[P3].bcast = \{\} \]
\[\text{pstate}[P3].msent = \{P1 \mapsto \{\}, P2 \mapsto \{\}, P3 \mapsto \{\}\} \]
\[\text{pstate}[P3].mcount = 0\]
broadcasted  \{ \{ m1 \} \}
delivered  \{ \} 
alive  \{ P1, P2, P3 \} 
correct  \{ P2 \} 

\begin{align*}
pstate[P1].bcast &= \{ m1 \} 
pstate[P1].msent &= \{ P1 \mapsto \{ \}, P2 \mapsto \{ m1 \}, P3 \mapsto \{ \} \} 
pstate[P1].mcount &= 1 
\end{align*}

sent(P2)
broadcasted \{ m1 \}
delivered \{ \}
avive \{ P1, P2, P3 \}
correct \{ P2 \}

\[
pstate[P1].bcast = \{ m1 \} \\
pstate[P1].msent = \{P1\to\{\},P2\to\{m1\},P3\to\{\}\} \\
pstate[P1].mcount = 1
\]

sent(P3)

\[
pstate[P2].bcast = \{ \} \\
pstate[P2].msent = \{P1\to\{\},P2\to\{\},P3\to\{\}\} \\
pstate[P2].mcount = 0
\]
broadcasted: \{ \text{m1} \} \\
delivered: \{ \} \\
alive: \{ \text{P1, P2, P3} \} \\
correct: \{ \text{P2} \}

\begin{align*}
pstate[\text{P1}].\text{bcast} &= \{ \text{m1} \} \\
pstate[\text{P1}].\text{msent} &= \{ \text{P1}\mapsto \{ \}, \text{P2}\mapsto \{ \text{m1} \}, \text{P3}\mapsto \{ \text{m3} \} \} \\
pstate[\text{P1}].\text{mcount} &= 1 \\
pstate[\text{P2}].\text{bcast} &= \{ \} \\
pstate[\text{P2}].\text{msent} &= \{ \text{P1}\mapsto \{ \}, \text{P2}\mapsto \{ \}, \text{P3}\mapsto \{ \} \} \\
pstate[\text{P2}].\text{mcount} &= 0 \\
pstate[\text{P3}].\text{bcast} &= \{ \} \\
pstate[\text{P3}].\text{msent} &= \{ \text{P1}\mapsto \{ \}, \text{P2}\mapsto \{ \}, \text{P3}\mapsto \{ \} \} \\
pstate[\text{P3}].\text{mcount} &= 0
\end{align*}
broadcasted \{ m1 \}  
delivered \{ \}  
alive \{ P1, P2, P3 \}  
correct \{ P2 \}  

\begin{align*}
\text{pstate}[P1].bcast &= \{ m1 \} \\
\text{pstate}[P1].msent &= \{ P1\mapsto\{\}, P2\mapsto\{m1\}, P3\mapsto\{m3\} \} \\
\text{pstate}[P1].mcount &= 1 \\
\text{receive}(P3) 
\end{align*}

\begin{align*}
\text{pstate}[P2].bcast &= \{ \} \\
\text{pstate}[P2].msent &= \{ P1\mapsto\{\}, P2\mapsto\{\}, P3\mapsto\{\} \} \\
\text{pstate}[P2].mcount &= 0 \\
\text{pstate}[P2].bcast &= \{ \} \\
\text{pstate}[P2].msent &= \{ P1\mapsto\{\}, P2\mapsto\{\}, P3\mapsto\{\} \} \\
\text{pstate}[P2].mcount &= 0
\end{align*}
broadcasted \{ m1 \}
delivered \{ P3_m1 \}
alive \{ P1, P2, P3 \}
correct \{ P2 \}

\begin{align*}
\text{pstate}[P1].bcast &= \{ m1 \} \\
\text{pstate}[P1].msent &= \{ P1\rightarrow\{\}, P2\rightarrow\{m1\}, P3\rightarrow\{m3\} \} \\
\text{pstate}[P1].mcount &= 1
\end{align*}

\textbf{receive(P3)}

\begin{align*}
\text{pstate}[P2].bcast &= \{ \} \\
\text{pstate}[P2].msent &= \{ P1\rightarrow\{\}, P2\rightarrow\{\}, P3\rightarrow\{} \} \\
\text{pstate}[P2].mcount &= 0
\end{align*}

\begin{align*}
\text{pstate}[P2].bcast &= \{ \} \\
\text{pstate}[P2].msent &= \{ P1\rightarrow\{\}, P2\rightarrow\{\}, P3\rightarrow\{} \} \\
\text{pstate}[P2].mcount &= 0
\end{align*}
fail(P1)

\[
pstate[P1].bcast = \{ m1 \} \\
pstate[P1].msent = \{ P1 \mapsto \{ \}, P2 \mapsto \{ m1 \}, P3 \mapsto \{ m3 \} \} \\
pstate[P1].mcount = 1
\]

\[
pstate[P2].bcast = \{ \} \\
pstate[P2].msent = \{ P1 \mapsto \{ \}, P2 \mapsto \{ \}, P3 \mapsto \{ \} \} \\
pstate[P2].mcount = 0
\]
fail(P1)

pstate[P1].bcast = { m1 }
pstate[P1].msent = {P1↦{}, P2↦{m1}, P3↦{m3}}
pstate[P1].mcount = 1

comm[P2] {m1} P1

pstate[P2].bcast = { }
pstate[P2].msent = {P1↦{}, P2↦{}, P3↦{}}
pstate[P2].mcount = 0

comm[P1] {} P3

pstate[P2].bcast = { }
pstate[P2].msent = {P1↦{}, P2↦{}, P3↦{}}
pstate[P2].mcount = 0
Best effort broadcast (2\textsuperscript{nd} try)

---

\textbf{MODULE} BestEffortBroadcast

Next == \( \forall p \in \text{Process} : \text{broadcast}(p) \lor \text{receive}(p) \lor \text{fail}(p) \)

---

Spec == \( \forall \text{Init} \)

\( \forall \, \square[\text{Next}]_{<<\text{comm, broadcasted, delivered, pstate, alive, correct>>>}} \)

---

\textbf{Just checking type safety!}

What about specific protocol properties?
System’s properties

• Safety
  • Something bad never happens
  • No two processes can access the critical section at the same time

• Liveness
  • Something good eventually happens
  • If a client wants to access a resource, it will eventually get access to that resource
• Temporal Logic of Actions:

  • Action formulas: describe state and state transitions

  • Temporal formulas: describe state sequences (traces)
Action Formulas

- $[A]<<e>> == A \lor e' = e$
- $<A><<e>> == A \land \neg(e' = e)$

MODULE OneBitClock

VARIABLE b

Init == (b=0) \lor (b=1)

TypeInv == b \in \{0,1\}

Next == IF b = 0 THEN b' = 1
ELSE b' = 0

Spec == Init \land \Box[Next]_<<b>>
Temporal Formulas

• Temporal operators

  • ⊨ F F is always true
  • <> F F is eventually true
  • F ~> G F leads to G

  • WF_<e> (A) Weak fairness for action A
  • SF_<e> (A) Strong fairness for action A
Temporal Formulas

• [] F  \text{F is always true}

• Formula [] P, where P is a state predicate, is true iff P is true in every state

\begin{center}
\begin{tikzpicture}
    \node at (0,0) [circle,fill,inner sep=1.5pt] (s1) [label=left:s1] {P};
    \node at (1,0) [circle,fill,inner sep=1.5pt] (s2) [label=left:s2] {P};
    \node at (2,0) [circle,fill,inner sep=1.5pt] (s3) [label=left:s3] {P};
    \draw (s1) -- (s2);
    \draw (s2) -- (s3);
    \end{tikzpicture}
\end{center}

• Formula [] [A]_<<e>>, where A is an action and e a state function, is true iff every successive step is an [A]_<<e>> step

\begin{center}
\begin{tikzpicture}
    \node at (0,0) [circle,fill,inner sep=1.5pt] (s1) [label=left:s1] {A};
    \node at (1,0) [circle,fill,inner sep=1.5pt] (s2) [label=left:s2] {A};
    \node at (2,0) [circle,fill,inner sep=1.5pt] (s3) [label=left:s3] {A};
    \draw (s1) -- (s2);
    \draw (s2) -- (s3);
    \end{tikzpicture}
\end{center}
Temporal Formulas

- $\langle\rangle F$: F is eventually true

- $\langle\rangle F = \sim[] (~F)$

  F is not always false
Temporal Formulas

- $F \leadsto G$  \quad F \text{ leads to } G

- $F \leadsto G = [\square](F \Rightarrow \langle\rangle G)$

Whenever $F$ is true, then $G$ is eventually true

- Every request leads to a response

request $\leadsto$ response
Temporal Formulas

- $\langle\rangle \langle\rangle F$ \textbf{Infinitely often} (Progress)
  - E.g. the traffic light is green infinitely often.

- $\langle\rangle \langle\rangle F$ \textbf{Eventually always} (Stability)
  - E.g. eventually all messages are delivered.
Liveness

• To prove liveness properties it is necessary to make some assumptions about the system environment.

• TLA has two forms of fairness:
  • Strong fairness
  • Weak fairness
Temporal Formulas

• Temporal operators

  • \([\square]\) F  
    F is always true
  • \(<\rangle\) F  
    F is eventually true
  • F \(\leadsto\) G  
    F leads to G
  • WF_<<e>> (A)  
    Weak fairness for action A
  • SF_<<e>> (A)  
    Strong fairness for action A
Weak fairness

- **WF_<<e>> (A)**  Weak fairness for action A

- \((<>[] \text{ ENABLED } <A>_{-<<e>>}) \Rightarrow ([]<> <A>_{-<<e>>})\)

  If \(A\) ever becomes forever enabled, then an \(A\) step must eventually occur
Strong fairness

- SF_<e> (A)  **Strong fairness for action A**

- ([<>] ENABLED <A>_<<e>>) => ([<>] <A>_<<e>>)

  If A is infinitely often enabled, then infinitely many A steps occur
Fairness in TLA

• **Weak fairness** of $A$ asserts that an $A$ step must eventually occur if $A$ is continuously enabled
  
  • continuously -> without interruption

• **Strong fairness** of $A$ asserts that an $A$ step must eventually occur if $A$ is continually enabled
  
  • continually -> repeatedly, possible with interruptions
Liveness properties

• Validity
  • For any two correct processes i and j, every message broadcast by i is eventually delivered by j.

• Agreement
  • If a message m is delivered by some correct process i, then m is eventually delivered by every correct process j.

Express this properties in TLA.