PODC 2000 TLA$^+$ Tutorial

Leslie Lamport

16 July 2000

MODULE *Mutex*

CONSTANT *Proc*

VARIABLE *state*

$Init \triangleq state = [p \in Proc \mapsto \text{"idle"}]$

$$
\begin{aligned}
Request(p) \triangleq\ & \land\ state[p] = \text{"idle"} \\
& \land\ state' = [state \text{ EXCEPT } ![p] = \text{"waiting"}]
\end{aligned}
$$

$$
\begin{aligned}
Acquire(p) \triangleq\ & \land\ state[p] = \text{"waiting"} \\
& \land\ \forall q \in Proc : state[q] \neq \text{"owner"} \\
& \land\ state' = [state \text{ EXCEPT } ![p] = \text{"owner"}]
\end{aligned}
$$

$$
\begin{aligned}
Release(p) \triangleq\ & \land\ state[p] = \text{"owner"} \\
& \land\ state' = [state \text{ EXCEPT } ![p] = \text{"idle"}]
\end{aligned}
$$

$Next \triangleq \exists p \in Proc : Request(p) \lor Acquire(p) \lor Release(p)$

$StarvationFree \triangleq \forall p \in Proc : \mathrm{SF}_{state}(Acquire(p))$

$Spec \triangleq Init \land \Box[Next]_{state} \land StarvationFree$

```
------------------------ MODULE Mutex ------------------------
CONSTANT Proc
VARIABLE state
--------------------------------------------------------------
Init == state = [p \in Proc |-> "idle"]

Request(p) == /\ state[p] = "idle"
              /\ state' = [state EXCEPT ![p] = "waiting"]

Acquire(p) == /\ state[p] = "waiting"
              /\ \A q \in Proc : state[q] # "owner"
              /\ state' = [state EXCEPT ![p] = "owner"]

Release(p) == /\ state[p] = "owner"
              /\ state' = [state EXCEPT ![p] = "idle"]

Next == \E p \in Proc : Request(p) \/ Acquire(p) \/ Release(p)
--------------------------------------------------------------
StarvationFree == \A p \in Proc : SF_state(Acquire(p))

Spec == Init /\ [][Next]_state /\ StarvationFree
==============================================================
```

MODULE *TimeClocks*

$\text{EXTENDS } Naturals, Sequences$

$\text{CONSTANT } Proc, \_ \ll \_$

$$\text{ASSUME } \forall p \in Proc : \begin{array}{l} \land \neg p \ll p \\ \land \forall q \in Proc \setminus \{p\} : (p \ll q) \lor (q \ll p) \\ \land \forall q, r \in Proc : (p \ll q) \land (q \ll r) \Rightarrow (p \ll r) \end{array}$$

$$a \prec b \triangleq \begin{array}{l} \lor\ a.TS < b.TS \\ \lor\ (a.TS = b.TS) \land (a.proc \ll b.proc) \end{array}$$

$\text{VARIABLES } state, msgQ, reqSet\ clock, lastTSent, lastTRcvd$

$vars \triangleq \langle state, msgQ, reqSet, clock, lastTSent, lastTRcvd \rangle$

$$Init \triangleq \begin{array}{lcl} \land\ state & = & [p \in Proc \mapsto \text{“idle”}] \\ \land\ msgQ & = & [p \in Proc \mapsto [q \in Proc \setminus \{p\} \mapsto \langle \rangle]] \\ \land\ reqSet & = & [p \in Proc \mapsto \{\}] \\ \land\ clock & \in & [Proc \to Nat] \\ \land\ lastTSent & = & [p \in Proc \mapsto [q \in Proc \setminus \{p\} \mapsto 0]] \\ \land\ lastTRcvd & = & [p \in Proc \mapsto [q \in Proc \setminus \{p\} \mapsto 0]] \end{array}$$

$$\begin{array}{l} Request(p) \triangleq \\ \quad \land\ state[p] = \text{“idle”} \\ \quad \land\ state' = [state \text{ EXCEPT } ![p] = \text{“waiting”}] \\ \quad \land\ \exists\, n \in Nat : \\ \qquad \land\ clock' = [clock \text{ EXCEPT } ![p] = n] \\ \qquad \land\ n > clock[p] \\ \qquad \land\ \text{LET } msg \triangleq [TS \mapsto n, proc \mapsto p, cmd \mapsto \text{“acquire”}] \\ \qquad\quad \text{IN } \land\ msgQ' = [msgQ \text{ EXCEPT } ![p] = \\ \qquad\qquad\qquad\qquad [q \in Proc \setminus \{p\} \mapsto Append(@[q], msg)]] \\ \qquad\qquad\ \land\ reqSet' = [reqSet \text{ EXCEPT } ![p] = @ \cup \{msg\}] \\ \qquad \land\ lastTSent' = [lastTSent \text{ EXCEPT } ![p] = [q \in Proc \setminus \{p\} \mapsto n]] \\ \quad \land\ \text{UNCHANGED } lastTRcvd \end{array}$$

$$\begin{array}{l} Acquire(p) \triangleq \\ \quad \text{LET } pReq \triangleq \text{CHOOSE } req \in reqSet[p] : req.proc = p \\ \quad \text{IN } \land\ state[p] = \text{“waiting”} \\ \qquad\ \land\ \forall\, req \in reqSet[p] \setminus \{pReq\} : pReq \prec req \\ \qquad\ \land\ \forall\, q \in Proc \setminus \{p\} : pReq \prec [TS \mapsto lastTRcvd[p][q] + 1, proc \mapsto q] \\ \qquad\ \land\ state' = [state \text{ EXCEPT } ![p] = \text{“owner”}] \\ \qquad\ \land\ reqSet' = [reqSet \text{ EXCEPT } ![p] = @ \setminus \{pReq\}] \\ \qquad\ \land\ \text{UNCHANGED } \langle msgQ, clock, lastTSent, lastTRcvd \rangle \end{array}$$

$$
\begin{array}{l}
Release(p) \;\triangleq \\
\quad \land\; state[p] = \text{"owner"} \\
\quad \land\; state' = [state \;\textsc{except}\; ![p] = \text{"idle"}] \\
\quad \land\; \textsc{let}\;\; msg \;\triangleq\; [TS \mapsto clock[p],\; proc \mapsto p,\; cmd \mapsto \text{"release"}] \\
\quad\quad\;\; \textsc{in}\;\;\; msgQ' = [msgQ \;\textsc{except}\; ![p] = \\
\qquad\qquad\qquad\qquad\qquad\qquad [q \in Proc \setminus \{p\} \mapsto Append(@[q],\; msg)]\,] \\
\quad \land\; lastTSent' = [lastTSent \;\textsc{except}\; ![p] = [q \in Proc \setminus \{p\} \mapsto clock[p]\,]\,] \\
\quad \land\; \textsc{unchanged}\; \langle clock,\; lastTRcvd,\; reqSet \rangle
\end{array}
$$

$$
\begin{array}{l}
RcvMsg(p,\; q) \;\triangleq \\
\quad \textsc{let}\;\; msg \qquad\;\; \triangleq\; Head(msgQ[q][p]) \\
\qquad\quad\; msgQTail \;\triangleq\; [msgQ \;\textsc{except}\; ![q][p] = Tail(@)] \\
\qquad\quad\; ack \qquad\;\;\; \triangleq\; [TS \mapsto clock'[p],\; proc \mapsto p,\; cmd \mapsto \text{"ack"}] \\
\quad \textsc{in}\quad\; \land\; msgQ[q][p] \neq \langle\,\rangle \\
\qquad\qquad \land\; clock' = [clock \;\textsc{except}\; ![p] = \textsc{if}\;\; msg.TS > @ \;\;\textsc{then}\;\; msg.TS \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad\;\; \textsc{else}\;\;\; @\,] \\
\qquad\qquad \land\; \textsc{if}\;\; \land\; msg.cmd = \text{"acquire"} \\
\qquad\qquad\qquad\;\;\; \land\; [TS \mapsto lastTSent[p][q] + 1,\; proc \mapsto p] \prec msg \\
\qquad\qquad\quad\;\; \textsc{then}\;\; \land\; msgQ' = [msgQTail \;\textsc{except}\; ![p][q] = Append(@,\; ack)] \\
\qquad\qquad\qquad\qquad\;\; \land\; lastTSent' = [lastTSent \;\textsc{except}\; ![p][q] = clock'[p]\,] \\
\qquad\qquad\quad\;\; \textsc{else}\;\;\; \land\; msgQ' = msgQTail \\
\qquad\qquad\qquad\qquad\;\; \land\; \textsc{unchanged}\; lastTSent \\
\qquad\qquad \land\; lastTRcvd' = [lastTRcvd \;\textsc{except}\; ![p][q] = msg.TS] \\
\qquad\qquad \land\; reqSet' = [reqSet \;\textsc{except}\; ![p] = \\
\qquad\qquad\qquad\qquad\qquad \textsc{case}\;\; msg.cmd = \text{"acquire"} \;\rightarrow\; @ \cup \{msg\} \\
\qquad\qquad\qquad\qquad\qquad\quad\;\; \Box\;\; msg.cmd = \text{"release"} \;\rightarrow\; \{m \in @ \;:\; m.proc \neq q\} \\
\qquad\qquad\qquad\qquad\qquad\quad\;\; \Box\;\; msg.cmd = \text{"ack"} \qquad\rightarrow\; @\,] \\
\qquad\qquad \land\; \textsc{unchanged}\; state
\end{array}
$$

$$
\begin{array}{l}
Tick(p) \;\triangleq\; \land\; \exists\, n \in Nat \;:\; \land\; n > clock[p] \\
\qquad\qquad\qquad\qquad\qquad\qquad\;\; \land\; clock' = [clock \;\textsc{except}\; ![p] = n] \\
\qquad\qquad\quad\; \land\; \textsc{unchanged}\; \langle state,\; msgQ,\; reqSet,\; lastTSent,\; lastTRcvd \rangle
\end{array}
$$

$$
\begin{array}{l}
Next \;\triangleq\; \exists\, p \in Proc \;:\; \lor\; Request(p) \lor Acquire(p) \lor Release(p) \\
\qquad\qquad\qquad\qquad\quad\;\; \lor\; \exists\, q \in Proc \setminus \{p\} \;:\; RcvMsg(p,\; q) \\
\qquad\qquad\qquad\qquad\quad\;\; \lor\; Tick(p)
\end{array}
$$

$$
\begin{array}{l}
Liveness \;\triangleq\; \forall\, p \in Proc \;:\; \land\; \mathrm{WF}_{vars}(Acquire(p)) \\
\qquad\qquad\qquad\qquad\qquad\quad\;\; \land\; \forall\, q \in Proc \setminus \{p\} \;:\; \mathrm{WF}_{vars}(RcvMsg(p,\; q))
\end{array}
$$

$$
Spec \;\triangleq\; Init \land \Box[Next]_{vars} \land Liveness
$$

MODULE $SMParameters$

CONSTANTS

$Proc$,	The set of processes.
Cmd,	The set of commands.
$InitState$,	The initial state-machine state.
$NewState(\_, \_)$,	$NewState(s, c)$ is the state obtained by executing command c in state s.
$Output(\_, \_)$	$Output(s, c)$ is the output generated by executing command c in state s.

VARIABLE int Represents the abstract interface.

CONSTANTS

$Request(\_, \_, \_, \_)$,	A $Request(int, int', c, p)$ step represents process p issuing command c to the system.
$Reply(\_, \_, \_, \_)$	A $Reply(int, int', r, p)$ step represents the system issuing reply r to process p.

MODULE $StateMachine$

EXTENDS $SMParams$

$Inner(state, pending, rcvd) \triangleq$ INSTANCE $InnerSM$

$Spec \triangleq \exists\, state, pending, rcvd : Inner(state, pending, rcvd)!ISpec$

$$
\begin{array}{l}
\textsc{module } \mathit{InnerSM} \\
\textsc{extends } \mathit{SMParams},\ \mathit{Sequences} \\
\hline
\textsc{variables } \mathit{state},\ \mathit{pending},\ \mathit{rcvd} \\
\mathit{vars} \triangleq \langle \mathit{int},\ \mathit{state},\ \mathit{pending},\ \mathit{rcvd} \rangle \\
\mathit{Init} \triangleq \begin{array}[t]{l} \land\ \mathit{state} = [p \in \mathit{Proc} \mapsto \mathit{InitState}] \\ \land\ \mathit{pending} = [p \in \mathit{Proc} \mapsto \langle\,\rangle] \\ \land\ \mathit{rcvd} = [p \in \mathit{Proc} \mapsto \langle\,\rangle] \end{array} \\
\hline
\mathit{IssueCmd}(p,\ c) \triangleq \\
\quad \land\ \mathit{Request}(\mathit{int},\ \mathit{int}',\ c,\ p) \\
\quad \land\ \mathit{rcvd}' = [\mathit{rcvd} \textsc{ except } ![p] = \mathit{Append}(@,\ c)] \\
\quad \land\ \textsc{unchanged } \langle \mathit{state},\ \mathit{pending} \rangle \\
\mathit{SequenceCmd}(p) \triangleq \\
\quad \land\ \mathit{rcvd}[p] \neq \langle\,\rangle \\
\quad \land\ \mathit{rcvd}' = [\mathit{rcvd} \textsc{ except } ![p] = \mathit{Tail}(@)] \\
\quad \land\ \mathit{pending}' = [q \in \mathit{Proc} \mapsto \mathit{Append}(\mathit{pending}[q],\ \mathit{Head}(\mathit{rcvd}[p]))] \\
\quad \land\ \textsc{unchanged } \langle \mathit{int},\ \mathit{state} \rangle \\
\mathit{IssueReply}(p) \triangleq \\
\quad \land\ \mathit{pending}[p] \neq \langle\,\rangle \\
\quad \land\ \mathit{Reply}(\mathit{int},\ \mathit{int}',\ \mathit{Output}(\mathit{state}[p],\ \mathit{Head}(\mathit{pending}[p])),\ p) \\
\quad \land\ \mathit{state}' = [\mathit{state} \textsc{ except } ![p] = \mathit{NewState}(@,\ \mathit{Head}(\mathit{pending}[p]))] \\
\quad \land\ \mathit{pending}' = [\mathit{pending} \textsc{ except } ![p] = \mathit{Tail}(@)] \\
\quad \land\ \textsc{unchanged } \mathit{rcvd} \\
\hline
\mathit{Next} \triangleq \exists\, p \in \mathit{Proc} : \begin{array}[t]{l} \lor\ \exists\, c \in \mathit{Cmd} : \mathit{IssueCmd}(p,\ c) \\ \lor\ \mathit{SequenceCmd}(p) \\ \lor\ \mathit{IssueReply}(p) \end{array} \\
\mathit{ISpec} \triangleq \begin{array}[t]{l} \land\ \mathit{Init} \\ \land\ \Box[\mathit{Next}]_{\mathit{vars}} \\ \land\ \forall\, p \in \mathit{Proc} : \begin{array}[t]{l} \land\ \mathit{WF}_{\mathit{vars}}(\mathit{SequenceCmd}(p)) \\ \land\ \mathit{WF}_{\mathit{vars}}(\mathit{IssueReply}(p)) \end{array} \end{array}
\end{array}
$$

$$
\begin{array}{l}
\textsc{module}\ \mathit{MutexSM} \\
\hline
\textsc{variable}\ \mathit{int} \\
\textsc{constants}\ \mathit{Proc},\ \mathit{Request}(\_,\ \_,\ \_,\ \_),\ \mathit{Reply}(\_,\ \_,\ \_,\ \_) \\
\hline
\mathit{InitState}\ \triangleq\ [\mathit{free} \mapsto \text{TRUE}, \mathit{waiting} \mapsto \langle\,\rangle] \\
\mathit{NewState}(s, c)\ \triangleq \\
\quad \textsc{if}\ \ c.\mathit{cmd} = \text{“acquire”} \\
\quad\quad \textsc{then}\ \ \textsc{if}\ \ s.\mathit{free}\ \ \textsc{then}\ \ [s\ \textsc{except}\ !.\mathit{free} = \text{FALSE}] \\
\quad\quad\quad\quad\quad\quad\quad\quad \textsc{else}\ \ [s\ \textsc{except}\ !.\mathit{waiting} = \mathit{Append}(@,\ c.\mathit{proc})] \\
\quad\quad \textsc{else}\ \ \textsc{if}\ \ s.\mathit{waiting} = \langle\,\rangle\ \ \textsc{then}\ \ [s\ \textsc{except}\ !.\mathit{free} = \text{TRUE}] \\
\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad \textsc{else}\ \ [s\ \textsc{except}\ !.\mathit{waiting} = \mathit{Tail}(@)] \\
\mathit{None}\ \triangleq\ \textsc{choose}\ v : v \notin \mathit{Proc} \\
\mathit{Output}(s, c)\ \triangleq \\
\quad \textsc{if}\ \ c.\mathit{cmd} = \text{“acquire”}\ \ \textsc{then}\ \ \textsc{if}\ \ s.\mathit{free}\ \ \textsc{then}\ \ c.\mathit{proc} \\
\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad \textsc{else}\ \ \mathit{None} \\
\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad \textsc{else}\ \ \textsc{if}\ \ s.\mathit{waiting} \neq \langle\,\rangle\ \ \textsc{then}\ \ \mathit{Head}(s.\mathit{waiting}) \\
\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad \textsc{else}\ \ \mathit{None} \\
\mathit{Cmd}\ \triangleq\ [\mathit{cmd} : \{\text{“acquire”},\ \text{“release”}\},\ \mathit{proc} : \mathit{Proc}] \\
\textsc{instance}\ \mathit{StateMachine} \\
\hline
\end{array}
$$

```
---------------------------- MODULE MutexUser ----------------------------
VARIABLE int
CONSTANTS Request(_, _, _, _), Reply(_, _, _, _), Proc
  ------------------------ MODULE Inner ------------------------
  VARIABLE state
  None ≜ CHOOSE v : v ∉ Proc
  Init ≜ state = [p ∈ Proc ↦ "idle"]
  IssueAcq(p) ≜ ∧ state[p] = "idle"
                ∧ Request(int, int', [cmd ↦ "acquire", proc ↦ p], p)
                ∧ state' = [state EXCEPT ![p] = "waiting"]
  IssueRel(p) ≜ ∧ state[p] = "owner"
                ∧ Request(int, int' [cmd ↦ "release", proc ↦ p], p)
                ∧ state' = [state EXCEPT ![p] = "idle"]
  RcvReply(p) ≜ ∃ r ∈ Proc ∪ {None} :
                    ∧ Reply(int, int', r, p)
                    ∧ state' = IF r = p
                                 THEN [state EXCEPT ![p] = "owner"]
                                 ELSE state
  Next ≜ ∃ p ∈ Proc : IssueAcq(p) ∨ IssueRel(p) ∨ RcvReply(p)
  ISpec ≜ Init ∧ □[Next]_⟨int, state⟩
  ==============================================================
I(state) ≜ INSTANCE Inner
Spec ≜ 𝐄 state : I(state)!ISpec
==========================================================================
```

$\textsc{module}\ \mathit{MutexSystem}$

$\textsc{variable}\ \mathit{int}$

$\textsc{constants}\ \mathit{Request}(\_, \_, \_, \_),\ \mathit{Reply}(\_, \_, \_, \_),\ \mathit{Proc}$

$\mathit{Env}\ \triangleq\ \textsc{instance}\ \mathit{MutexUser}$

$\mathit{Sys}\ \triangleq\ \textsc{instance}\ \mathit{MutexSM}$

$\mathit{ClosedSystem}\ \triangleq\ \mathit{Env}!\mathit{Spec} \land \mathit{Sys}!\mathit{Spec}$

$\mathit{OpenSystem}\ \triangleq\ \mathit{Env}!\mathit{Spec} \overset{+}{\rhd} \mathit{Sys}!\mathit{Spec}$

TLA home page

`http://www.research.digital.com/SRC/tla/`

Handout

`http://www.research.digital.com/SRC/tla/podc00-handout.ps`