MVSI – Modelling and Verification of Computer Systems

TLA$^+$ modelling (1)

Tutorial – with solutions

Dominique Méry – 20th November 2026

✎ Exercise 1

`appex1_1.tla, appex1_1.cfg`

Access to a room is controlled by a system which observes the people entering or leaving the room. This system is a set of sensors identifying the passage of a person from the outside to the inside and from the inside to the outside. The system must guarantee that at most `max` people are in the room.
Write a TLA$^+$ module modelling such a system and satisfying the expected property.

✔ Solution of exercise 1

We give three possible solutions, according to our analysis:

- the first solution proposes two actions *entrer* (enter) and *sortir* (leave), and defines a transition relation *next*. A model is defined by instantiating *max* and by testing the two safety properties *question1* and *safety1*. Both questions fail, hence the safety property does not hold.
- the second solution proposes two actions *entrer2* and *sortir*, and defines a transition relation *next2*. A model is defined by instantiating *max* and by testing the two safety properties *question1* and *safety1*. No error is reported for *safety1*.
- the third solution proposes two actions *entrer2* and *sortir2*, and defines a transition relation *next3*. A model is defined by instantiating *max* and by testing the two safety properties *question1* and *safety1*. The tool checks the property *safety1*.

Listing 1: TLA$^+$ module `appex1_1.tla`

```
------------------------------- MODULE appex1_1 ---------------------
EXTENDS Naturals, Integers, TLC
-------------------------------------------------------------------------
CONSTANTS max
-------------------------------------------------------------------------
VARIABLES  np
---------------------------------
(*  tentative 1 *)
entrer ==
    /\ TRUE
    /\ np'=np +1
sortir == np'=np-1
next1 == entrer \/ sortir
Init == np=0
-------------------------------------------------------------------------
(* tentative 2 *)
entrer2 == np<max /\ np'=np+1
next2 ==  entrer2  \/  sortir
-------------------------------------------------------------------------
(* tentative 3 *)
sortir2 == np>0 /\ np'=np-1
next3 ==  entrer2  \/  sortir2
------------------
Safety1 == np \in Int
Safety2 == np \leq max
Safety3 == np \geq 0
qq ==  np # 89

Safety ==  Safety1 /\ Safety2 /\ Safety3
Test == Safety

=========================================================================
```

Listing 2: TLC configuration `appex1_1.cfg`

```
\* SPECIFICATION
\* Uncomment the previous line and provide the specification name if it's declared
\* in the specification file. Comment INIT / NEXT parameters if you use SPECIFICATION.

CONSTANTS
    greeting = "Hello"
    max = 19
    minint = 1000
    maxint =  1000

INIT Init
NEXT next3


\* PROPERTY
\* Uncomment the previous line and add property names

 INVARIANT
\* Uncomment the previous line and add invariant names
\* SPECIFICATION
\*    Safety2
 \*   Safety1
 \*   Safety3
  \*  qq
    Test
```

✎ Exercise 2

The GCD of two numbers satisfies the following properties:

- $\forall a,b \in \mathbb{N}.gcd(a,b) = gcd(b,a)$
- $\forall a,b \in \mathbb{N}.gcd(a,a+b) = gcd(a,b)$
- Write a TLA$^+$ specification computing the GCD of two given numbers.
- Give an explanation or a justification of the correctness of this solution.

✔ Solution of exercise 2

Listing 3: TLA$^+$ module `appex1_2.tla`

```
------------------- MODULE appex1_2 -----------------
(* Computing  GCD of two numbers a and b *)
EXTENDS Naturals,TLC
CONSTANTS a,b
VARIABLES  x,y
-----------------------------------
ASSUME a \in Nat /\ b \in Nat
Init == x=a /\ y=b
-----------------------------------
a1 ==
    /\ x > y
    /\ x'=x-y
    /\ y'=y
a2 ==
    /\ x < y
    /\ y'=y-x
    /\ x'=x
over == x=y /\ x'=x /\ y'=y
-----------------------------------
Next ==
    \/ a1
    \/ a2
    \/ over

```

```
------------------------------------
(* Safety properties to be verified *)
question == x # y
prop1 == [](x \geq 0 /\ y \geq 0)
prop2 == x+y \leq a+b
prop == question
Check == prop2
=============================================
```

Listing 4: TLC configuration `appex1_2.cfg`

```
\* SPECIFICATION
\* Uncomment the previous line and provide the specification name if it's declared
\* in the specification file. Comment INIT / NEXT parameters if you use SPECIFICATION.

CONSTANTS
    greeting = "Hello"
    a = 99
    b = 56

INIT Init
NEXT Next

PROPERTY
\* Uncomment the previous line and add property names
    prop1
INVARIANT
\* Uncomment the previous line and add invariant names
Check

CHECK_DEADLOCK FALSE
```

✎ Exercise 3 `appex1_3.tla,appex1_3.cfg, petri10`

A Petri net is a tuple $R = (S, T, F, K, M, W)$ where

- S is the (finite) set of places.
- T is the (finite) set of transitions.
- $S \cap T = \emptyset$
- F is the flow relation: $F \subseteq S \times T \cup T \times S$
- K is the capacity of each place: $K \in S \mapsto Nat$.

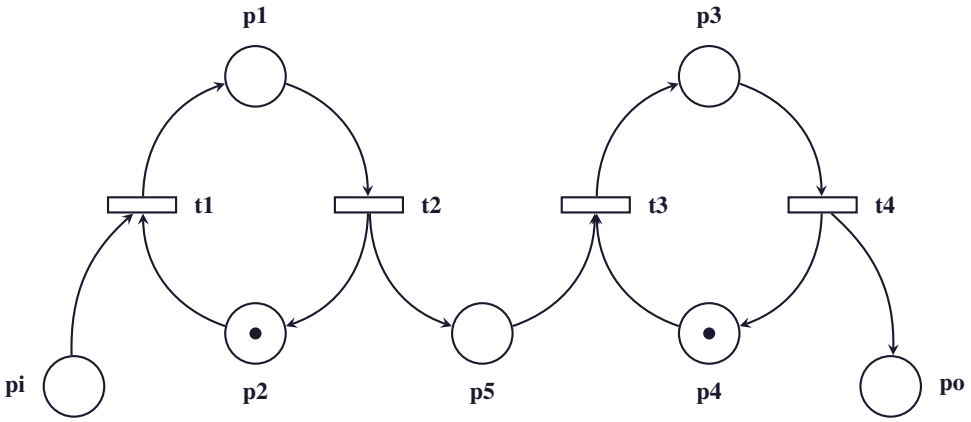
The function K assigns to every place s of S the maximum number of tokens that it may contain. If this function is not defined, then the capacity is infinite.

- M is the initial marking of each place:
  $M \in S \to Nat$ and satisfies the condition $\forall s \in S : M(s) \leq K(s)$.
- W is the weight of each arc: $W \in F \to Nat$
- a marking M for R is a function from S to Nat:
  $M \in S \to Nat$, satisfying the condition $\forall s \in dom(K) : M(s) \leq K(s)$.
- a transition t of T is enabled from a marking M of R if
  1. $\forall s \in \{ s' \in S \mid (s',t) \in F \} : M(s) \geq W(s,t)$.
  2. $\forall s \in \{ s' \in S \mid (t,s') \in F \} : M(s) \leq K(s) - W(s,t)$.
- For each transition t of T, Pre(t) is the set of places leading to t, and Post(t) is the set of places pointed to by an arc from t:
  $Pre(t) = \{ s' \in S : (s',t) \in F \}$ and $Post(t) = \{ s' \in S: (t,s') \in F \}$

- Let t be a transition of T enabled from a marking M of R:
  1. $\forall$ s $\in$ { s' $\in$ S | (s',t) $\in$ F }: M(s) $\geq$ W(s,t).
  2. $\forall$ s $\in$ { s' $\in$ S | (t,s') $\in$ F }: M(s) $\leq$ K(s) - W(s,t).
- a new marking M' is defined from M by: $\forall$ s $\in$ S,

$$M'(s) = \begin{cases} M(s) - W(s,t), & \text{IF } s \in \text{PRE}(t) - \text{POST}(t) \\ M(s) + W(t,s), & \text{IF } s \in \text{POST}(t) - \text{PRE}(t) \\ M(s) - W(s,t) + W(t,s), & \text{IF } s \in \text{PRE}(t) \cap \text{POST}(t) \\ M(s), & \text{OTHERWISE} \end{cases}$$

We consider the following net:



Skeleton of the module:

```
---------------------- MODULE petri10 ----------------------
EXTENDS Naturals, TLC
CONSTANTS Places, N, Q, B
VARIABLES M
Init == M = [p \in Places |-> IF p \in {"p4","p2"} THEN 1 ELSE
                              IF p = "pi" THEN N ELSE 0]
t1 == ...
t2 == ...
t3 == ...
t4 == ...
Next == t1 \/ t2 \/ t3 \/ t4
=============================================================
```

Question 3.1 Translate this net into a TLA$^+$ module whose skeleton is given in the text. To do so, give the definition of the four transitions $t1$, $t2$, $t3$, $t4$. The capacity of the places is not taken into account: the places have a capacity of at most one token, except the place pi which may contain N tokens; the place p5 may contain at most B tokens and the place po may contain at most Q tokens.

Question 3.2
Give a relation linking the places po, p1, p3, p5, pi and the value N. Justify your answer.

Question 3.3
Assuming that the place po may contain at most Q tokens, give a condition on Q such that all the tokens of pi are eventually consumed. Justify your answer.

Question 3.4 Explain what this Petri net models.

✔ **Solution of exercise 3**

Listing 5: TLA$^+$ module `appex1_3.tla`

```
1 -------------------------------- MODULE appex1_3 --------------------------------
2 (* petri10 *)
3 EXTENDS Naturals,TLC
4 CONSTANTS Places,N,Q,B
5 VARIABLES  M
```

```
------------------------------

t1 ==
        /\ M["p2"]  \geq 1  /\  M["pi"]  \geq 1
        /\ M'=  [[[M EXCEPT!["p1"]= 1]
                    EXCEPT!["pi"]= @-1]
                    EXCEPT!["p2"]=@-1]

t2 ==
        /\ M["p1"] \geq 1 /\ M["p5"] <  B
        /\ M'=  [[[M EXCEPT!["p1"]=@-1]
                    EXCEPT!["p5"]=M["p5"]+1]
                    EXCEPT!["p2"]=1]

t3 ==
        /\ M["p5"] \geq 1 /\ M["p4"] \geq 1
        /\ M'=  [[[M EXCEPT!["p3"]=@+1]
                    EXCEPT!["p5"]=@-1]
                    EXCEPT!["p4"]=@-1]



t4 ==
        /\ M["p3"] \geq 1 /\ M["po"]<Q
        /\ M'=  [[[M EXCEPT!["p3"]= M["p3"]-1]
                 EXCEPT!["po"]=M["po"]+1]
                 EXCEPT!["p4"]=M["p4"]+1]

------------------------------

Init1 ==  M = [p \in Places |-> IF p \in {"p4","p2"} THEN 1 ELSE
                IF p = "pi" THEN N ELSE 0 ]
Init == Init1
Next == t1 \/ t2 \/ t3 \/ t4 \/ M'=M


------------------------------

TypeInvariant ==  \A p \in Places :  M[p] \geq 0

Inv1 ==  M["pi"]+M["p5"]+M["po"]+M["p1"]+ M["p3"]  = N

Inv2 ==  M["po"] \leq  Q

QInv4 ==  M["pi"]+M["p5"]+M["po"]+M["p2"]+M["p4"] = N+2
Inv5 ==  M["p3"]+M["p4"]+M["p1"]+M["p2"] = 2

Inv3 ==  M["p3"] = 0

Inv6 == M["p1"] + M["p2"] = 1

Question == M["po"] #  N
Safety1 ==  M["p2"] \leq  1  /\ M["p2"] \geq  0
Check == Inv6
safe == Inv3
=============================================
```

Listing 6: TLC configuration appex1_3.cfg

```
\* SPECIFICATION
\* Uncomment the previous line and provide the specification name if it's declared
\* in the specification file. Comment INIT / NEXT parameters if you use SPECIFICATION.

CONSTANTS
    greeting = "Hello"
    B = 4
    N = 9
    Q = 10
    Places = {"pi","po","p1","p2","p3","p4","p5"}

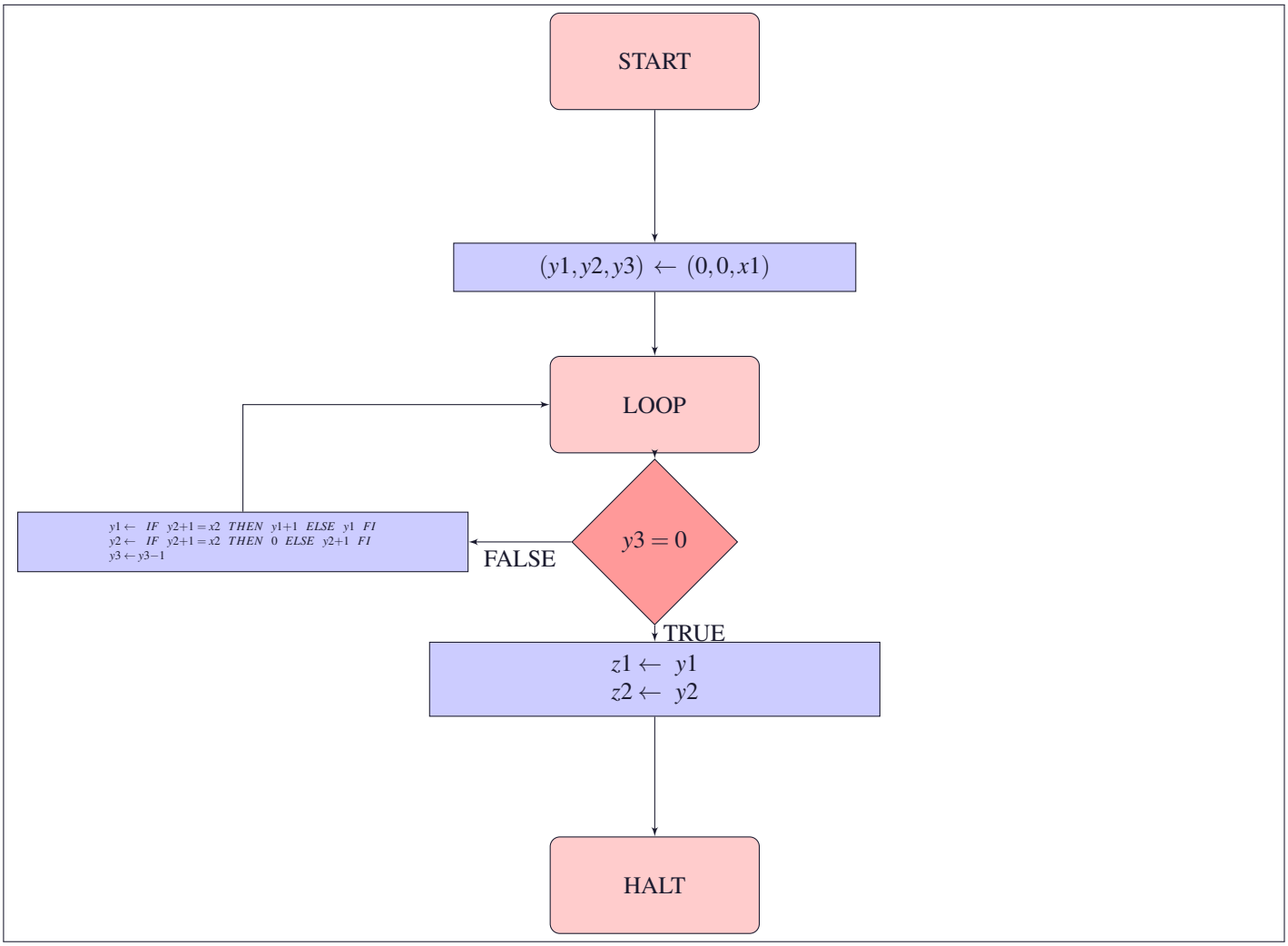
INIT Init
NEXT Next
```

```
\* PROPERTY
\* Uncomment the previous line and add property names

INVARIANT
\* Uncomment the previous line and add invariant names
    safe
```

✎ Exercise 4 `appex1_4`

We consider the following algorithm, described by a flowchart:



Question 4.1 Translate this algorithm into a TLA$^+$ module.

Question 4.2 Write a contract for this algorithm.

Question 4.3 Test the values of the variables at run time.

Question 4.4 Show that this algorithm is partially correct with respect to its precondition and its postcondition.

✔ Solution of exercise 4

Listing 7: TLA$^+$ module `appex1_4.tla`

```
---------------------------------- MODULE appex1_4 ------------------------------
(* modules de base importables *)
EXTENDS Naturals, Integers, TLC
```

```
----------------------------------------------------------------------------
CONSTANTS  x10,x20,U,MAX

MIN == -MAX
----------------------------------------------------------------------------
VARIABLES  x1,x2,y1,y2,y3,z1,z2,pc
----------------------------------------------------------------------------
locs == {"START","HALT","LOOP"}
----------------------------------------------------------------------------
BF(X) == X#U => X \in MIN..MAX

ASSUME  BF(x10) /\ BF(x20)
----------------------------------------------------------------------------


Init == pc="START" /\ x1=x10 /\ x2=x20 /\ y1=U /\ y2 = U /\ y3=U /\ z1 = U /\ z2 = U
----------------------------------------------------------------------------
actionSTART_LOOP ==
              /\ pc = "START"
              /\ pc'="LOOP"
              /\ y1'=0
              /\ y2'=0
              /\ y3'=x1
              /\ UNCHANGED <<x1,x2,z1,z2>>


actionLOOP_HALT ==
             /\ pc="LOOP"
             /\ y3=0
             /\ pc'="HALT"
              /\ y1'=y1
              /\ y2'=y2
              /\ y3'=y3
              /\ z1'=y1
              /\ z2'=y2
              /\ UNCHANGED <<x1,x2>>



actionLOOP_LOOP ==
             /\ pc="LOOP"
             /\ y3#0
             /\ pc'=pc
             /\ y1'= IF y2+1=x2 THEN y1+1 ELSE y1
             /\ y2'= IF y2+1 = x2 THEN 0 ELSE y2+1
             /\ y3'=y3-1
             /\ z1'=z1
             /\ z2'=z2
              /\ UNCHANGED <<x1,x2>>


skip == UNCHANGED <<x1,x2, y1,y2,y3,z1,z2,pc>>

----------------------------------------------------------------------------


Next ==
    \/ actionSTART_LOOP
    \/  actionLOOP_HALT
    \/ actionLOOP_LOOP
    \/ skip


----------------------------------------------------------------------------

(* vérification du contrôle *)
safety1 == pc \in locs
(* correction partielle *)
safety2 == pc="HALT" => z1 = x1 \div x2 /\ z2= x1 % x2 /\ PrintT(z1) /\ PrintT(z2)
safety3 == pc="HALT" =>     x1 = z1*x2 +z2  /\ 0 \leq z2 /\ z2 < x2
(* vérification de l'absence d'erreurs à l'exé&cution ou RTE *)
safety4 == BF(z1) /\BF(z2) /\BF(y1) /\BF(y2) /\BF(y3)
```

```
prop1 == [] (x1=x10 /\ x2=x20)


Safety == safety1 /\ safety2  /\ safety3  /\ safety4
=============================================================================
```

Listing 8: TLC configuration appex1_4.cfg

```
\* SPECIFICATION
\* Uncomment the previous line and provide the specification name if it's declared
\* in the specification file. Comment INIT / NEXT parameters if you use SPECIFICATION.

CONSTANTS
    x10 = 46
    x20 = 45

    MAX = 1000
    U = 1001

INIT Init
NEXT Next

\* PROPERTY
\* Uncomment the previous line and add property names

INVARIANT
\* Uncomment the previous line and add invariant names
Safety
```