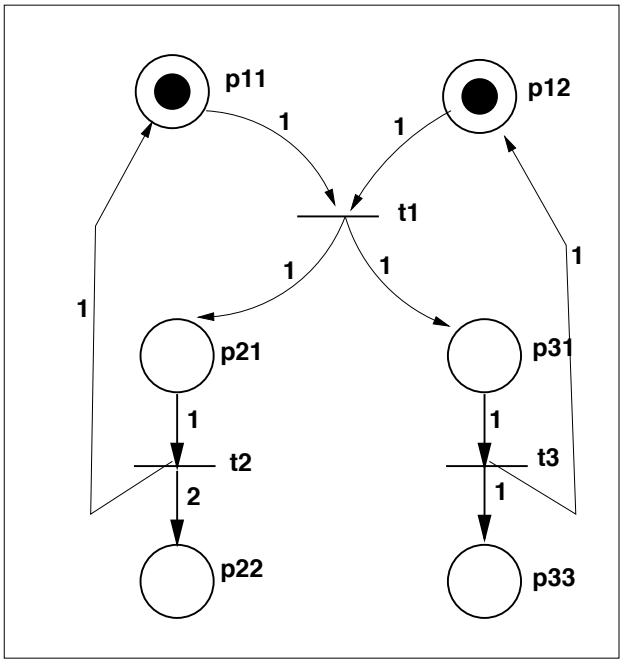


Cours Modélisation et vérification des systèmes informatiques
Exercices (avec les corrections)
Modélisation TLA$^+$ (2)
par Dominique Méry
9 octobre 2024

Exercice 1 *Soit le réseau de Petri de la figure* **??**.

Question 1.1 *Déterminer les conditions initiales.*

Question 1.2 *Déterminer les relations modélisant les transitions.*

Question 1.3 *Valider les propriétés et les hypothèses que vous pourrez faire sur ce réseau de Petri.*

◇– **Solution de l'exercice 1**

MODULE $petri8$

EXTENDS *Naturals*, *TLC*
VARIABLES M
CONSTANT *Places*

condition de t1

$$
\begin{array}{l}
t1 \triangleq \\
\quad \wedge M[\text{"p11"}] \geq 1 \wedge M[\text{"p12"}] \geq 1 \\
\quad \wedge M' = [[[[M \text{ EXCEPT! } [\text{"p11"}] = @-1] \\
\qquad\qquad \text{EXCEPT!}[\text{"p12"}] = @-1] \\
\qquad\qquad \text{EXCEPT!}[\text{"p21"}] = @+1] \\
\qquad\qquad \text{EXCEPT!}[\text{"p31"}] = @+1]
\end{array}
$$

$$
\begin{array}{l}
t2 \triangleq \\
\quad \wedge M[\text{"p21"}] \geq 1 \\
\quad \wedge M' = [[[M \text{ EXCEPT!}[\text{"p21"}] = @-1] \\
\qquad\qquad \text{EXCEPT!}[\text{"p22"}] = @+2]
\end{array}
$$

$$\text{EXCEPT}!["\mathsf{p11}"] \;=\; @{+}1]$$

$t3 \;\triangleq$
$$\begin{array}{l} \quad \wedge\, M["\mathsf{p31}"] \;\geq\; 1 \\ \quad \wedge\, M' \;=\; [[[M \text{ EXCEPT}! \,["\mathsf{p31}"] \;=\; @{-}1] \\ \qquad\qquad\qquad \text{EXCEPT}!["\mathsf{p33}"] \;=\; @{+}1] \\ \qquad\qquad\qquad \text{EXCEPT}!["\mathsf{p12}"] \;=\; @{+}1] \end{array}$$

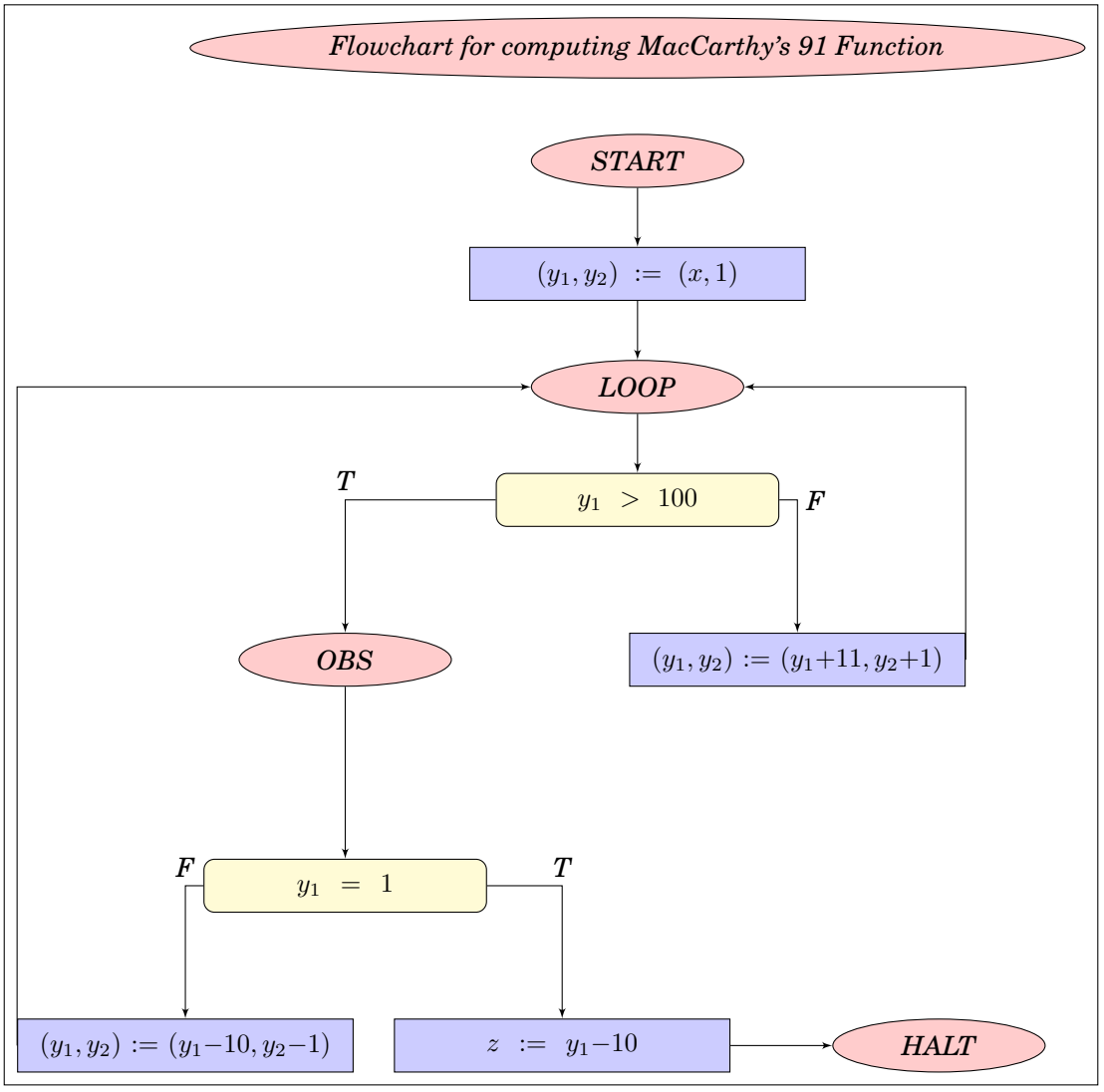
$\mathit{Init} \;\triangleq\; M \;=\; [p \in \mathit{Places} \mapsto \text{IF } p \in \{"\mathsf{p11}", "\mathsf{p12}"\} \text{ THEN } 1 \text{ ELSE } 0\,]$
$\mathit{Next} \;\triangleq\; t_1 \;\vee\; t_2 \;\vee\; t3$

$Q1 \;\triangleq\; M["\mathsf{p22}"] \neq 32$

$Q2 \;\triangleq\; M["\mathsf{p21}"] = 1 \;\wedge\; M["\mathsf{p31}"] = 1 \;\Rightarrow\; 2{\cdot}M["\mathsf{p22}"] = M["\mathsf{p33}"]$

Fin 1

Exercice 2 *Soit l'organigramme suivant proposé par Z. Manna dans son ouvrage Mathematical Theory of Computation.*



Question 2.1 *Construire un module TLA$^+$ modélisant les différents pas de calcul.*

Question 2.2 *Evaluer l'algorithme en posant des questions de sûreté suivantes :*

1. *l'algorithme est partiellement correct.*
2. *l'algorithme n'a pas d'erreurs à l'exécution.*

◇– **Solution de l'exercice 2**

MODULE *maccarthy*91

EXTENDS *Naturals*, *TLC*, *Integers*
CONSTANTS x, min, max
VARIABLES $y_1, y2, z, c$

$a \triangleq c = \text{"START"} \land y_1' = x \land y_2' = 1 \land c' = \text{"LOOP"} \land \text{UNCHANGED } \langle z \rangle$

$b \triangleq$
$\land c = \text{"LOOP"} \land y_1 \leq 100$
$\land y_1' = y_1+11 \land y_2' = y_2+1$

$\land$ UNCHANGED $\langle z, c\rangle$

$cc \triangleq$
$\land c =$ "LOOP" $\land\ y_1 > 100 \land\ y_2 \neq 1$
$\land\ y_1' = y_1 - 10 \land\ y_2' = y_2 - 1$
$\land$ UNCHANGED $\langle z, c\rangle$

$d \triangleq$
$\land c =$ "LOOP" $\land\ y_1 > 100 \land\ y_2 = 1$
$\land\ z' = y_1 - 10 \land\ c' =$ "HALT"
$\land$ UNCHANGED $\langle y_1, y2\rangle$

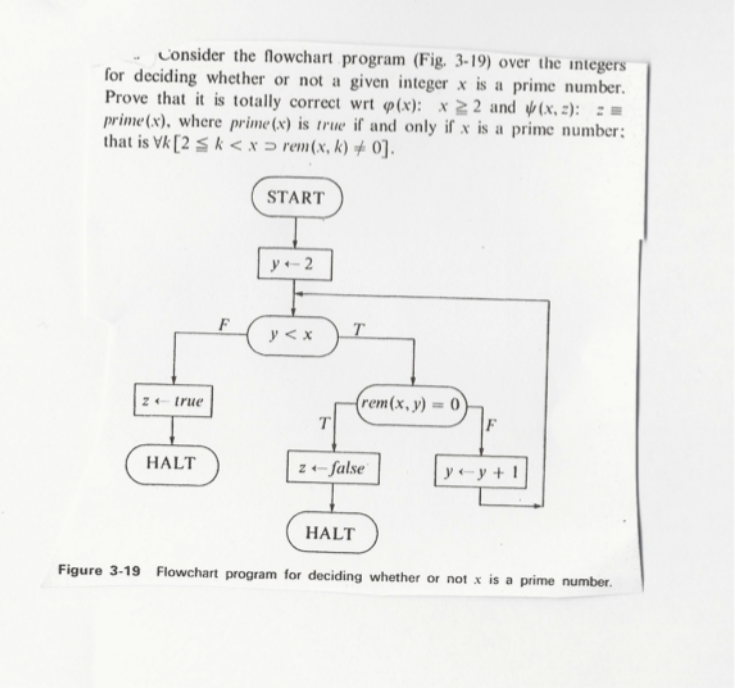
$next \triangleq a \lor b \lor cc \lor d \lor$ UNCHANGED $\langle y_1, y2, z, c\rangle$
$init_1 \triangleq y_1 \in Int \land y_2 \in Int \land z \in Int \land c =$ "START"
$init \triangleq y_1 = 0 \land y_2 = 0 \land z = 0 \land c =$ "START"

$Q1 \triangleq c \neq$ "HALT" c prned la valeur HALT
$Q2 \triangleq c =$ "HALT" $\Rightarrow z =$ IF $x > 100$ THEN $x - 10$ ELSE 91
$Qy_1 \triangleq min \leq y_1 \land y_1 \leq max$

$Question \triangleq Qy1$

Fin 2

Exercice 3 *Soit le schéma suivant définissant un calcul déterminant sir un nombre entier naturel est premier ou non.*



Question 3.1 *Ecrire un modèle TLA modélisant ce schéma de calcul.*

Question 3.2 *Ecrire un invariant à partir d'annotations que vous définirez après avoir défini des points de contrôle.*

Question 3.3 *Vérifier la correction partielle*

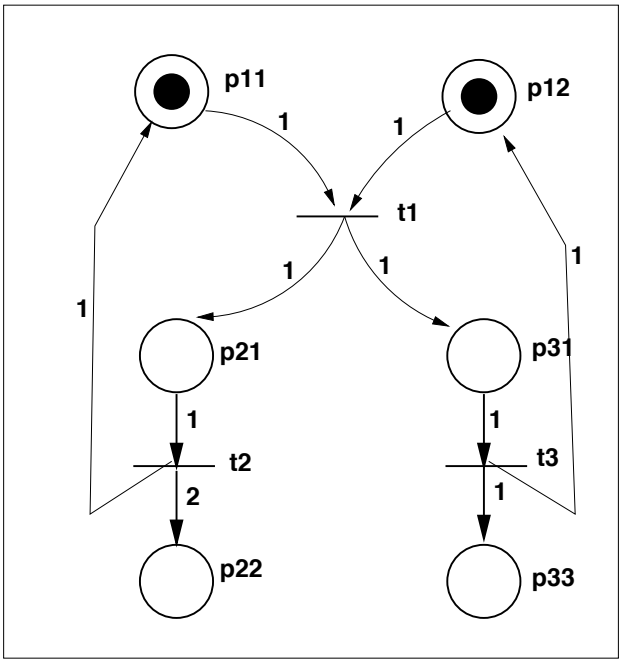


FIGURE 1 – Réseau de Petri

Question 3.4 *Vérifier l'absence d'erreurs à l'exécution.*

Exercice 4 *Le module truc permet de résoudre un problème très classique en informatique : trouver un chemin entre un sommet input et des sommets output supposés être des sommets de sortie.*

Question 4.1 *Pour trouver un chemin de input à l'un des sommets de output, il faut poser une question de sûreté à notre système de vérification. Donner une question de sûreté à poser permettant de trouver un chemin de input vers un sommet de output.*

Question 4.2 *On désire utiliser cette technique pour trouver un chemin dans un labyrinthe. Un labyrinthe est représenté par une matrice carrée de taille n. On définit ensuite pour chaque élément* $<< i, j >>$ *de la matrice les voisins communiquant à l'aide de la fonction lab qui associe à* $<< i, j >>$ *les éléments qui peuvent être atteints en un coup. Par exemple, le mouvement possible à partir de* $<< 1, 1 >>$ *est* $<< 2, 1 >>$*, ou le mouvement possible à partir de* $<< 2, 1 >>$ *est* $<< 2, 2 >>$ *ou* $<< 1, 1 >>$*, ou le mouvement possible à partir de* $<< 2, 2 >>$ *est* $<< 2, 3 >>$ *ou* $<< 3, 2 >>$ *ou* $<< 2, 1 >>$*, ...*

```
lab == [<<x,y>> \in (nodes \X nodes) |->
                      IF x=1 /\ y=1  THEN {<<2,1>>} ELSE
                      IF x=2 /\ y=1  THEN {<<2,2>>}
                      IF x=1 /\ y=2  THEN {} ELSE
                      IF x=2 /\ y=2  THEN {<<3,2>>,<<2,3>>} ELSE
                      ELSE {}

                      ]
```

Modifier le module truc pour traiter ce problème et donner la question à poser pour trouver une sortie.

MODULE *truc*

MODULE *Naturals*

A dummy module that defines the operators that are defined by the real Naturals module.

$Nat \triangleq \{\ \}$
$a+b \triangleq \{a,\ b\}$
$a-b \triangleq \{a,\ b\}$
$a \cdot b \triangleq \{a,\ b\}$
$a^b \triangleq \{a,\ b\}$
$a < b \triangleq a = b$
$a > b \triangleq a = b$
$a \leq b \triangleq a = b$
$a \geq b \triangleq a = b$
$a \,\%\, b \triangleq \{a,\ b\}$
$a \div b \triangleq \{a,\ b\}$
$a \,..\, b \triangleq \{a,\ b\}$

MODULE *TLC*

LOCAL INSTANCE *Naturals*
LOCAL INSTANCE *Sequences*

$Print(out,\ val) \triangleq val$
$PrintT(out) \triangleq TRUE$
$Assert(val,\ out) \triangleq$ IF $val = TRUE$ THEN $TRUE$
ELSE CHOOSE $v : TRUE$
$JavaTime \triangleq$ CHOOSE $n : n \in Nat$
$TLCGet(i) \triangleq$ CHOOSE $n : TRUE$
$TLCSet(i,\ v) \triangleq TRUE$

$d :> e \triangleq [x \in \{d\} \mapsto e]$
$f \ @@\ g \triangleq [x \in (\text{DOMAIN } f) \cup (\text{DOMAIN } g) \mapsto$
IF $x \in$ DOMAIN f THEN $f[x]$ ELSE $g[x]]$
$Permutations(S) \triangleq$
$\{f \in [S \to S] : \forall\, w \in S : \exists\, v \in S : f[v] = w\}$

In the following definition, we use Op as the formal parameter rather than \prec because TLC Version 1 can't handle infix formal parameters.

$SortSeq(s,\ Op(\_,\ \_)) \triangleq$
LET $Perm \triangleq$ CHOOSE $p \in Permutations(1\ ..\ Len(s)) :$
$\forall\, i,\ j \in 1..Len(s) :$
$(i < j) \Rightarrow Op(s[p[i]],\ s[p[j]]) \vee (s[p[i]] = s[p[j]])$
IN $[i \in 1..Len(s) \mapsto s[Perm[i]]]$

$RandomElement(s) \triangleq$ CHOOSE $x \in s : TRUE$

$Any \triangleq$ CHOOSE $x : TRUE$

$ToString(v) \triangleq (\text{CHOOSE } x \in [a : v,\ b : STRING] : TRUE).b$

$TLCEval(v) \triangleq v$

FIGURE 2 – Modules Naturals.tla et TLC.tla

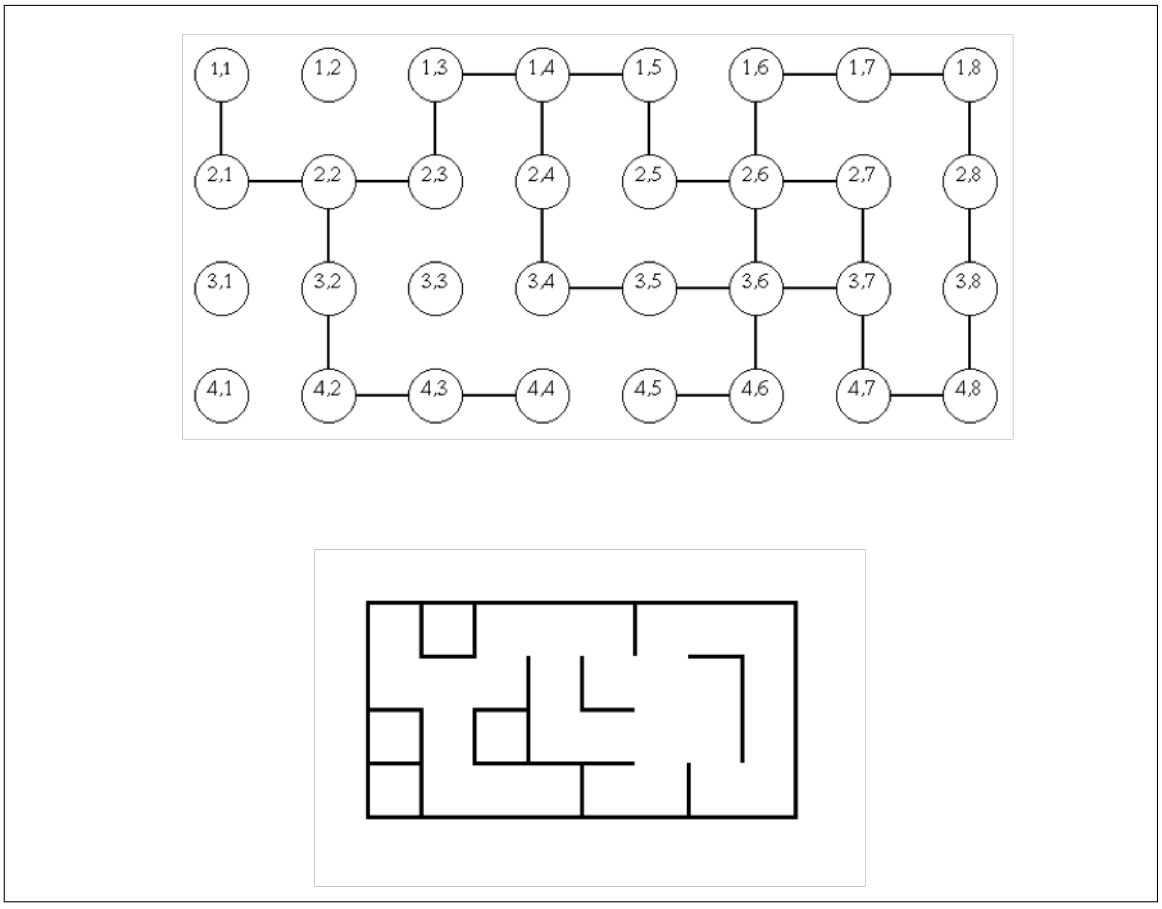


FIGURE 3 – Labyrinthe

```
EXTENDS Integers, TLC
VARIABLES p
CONSTANTS input, output

n ≜ 10
nodes ≜ 1..n
l ≜ [i ∈ 1..n ↦ IF i = 1 THEN {4, 5} ELSE
                IF i = 2 THEN {6, 7, 10} ELSE
                IF i = 4 THEN {7, 8} ELSE
                IF i = 5 THEN {} ELSE
                IF i = 6 THEN {4} ELSE
                IF i  = 7 THEN {5} ELSE
                IF i  = 8 THEN {5, 2} ELSE
                {}
                ]

Init ≜ p = 1
M(i) ≜ ∧ i ∈ l[p]
       ∧ p′ = i
Next ≜ ∃ i ∈ 1..n : M(i)
```

◇– **Solution de l'exercice 4**

MODULE *labyrinthe*

```
EXTENDS Integers, TLC
VARIABLES p
CONSTANTS input, output

n ≜ 10
nodes ≜ 1..n
l ≜ [i ∈ 1..n ↦ IF i = 1 THEN {4, 5} ELSE
                IF i = 2 THEN {6, 7, 10} ELSE
                IF i = 4 THEN {7, 8} ELSE
                IF i = 5 THEN {} ELSE
                IF i = 6 THEN {4} ELSE
                IF i  = 7 THEN {5} ELSE
                IF i  = 8 THEN {5, 2} ELSE
                {}
                ]

lab ≜ [⟨x, y⟩ ∈ (nodes × nodes) ↦
                IF x = 1  ∧ y = 1 THEN {⟨1, 2⟩} ELSE
                IF x = 1  ∧ y = 2 THEN {⟨1, 3⟩, ⟨2, 2⟩}
                ELSE {}

                ]

Init ≜ p = 1
M(i) ≜ ∧ i ∈ l[p]
       ∧ p′ = i
Next ≜ ∃ i ∈ 1..n : M(i)

Initlab ≜ p = ⟨1, 1⟩
```

$ML(q) \triangleq \wedge\, q \in lab[p]$
$\quad\quad\quad \wedge\, p' = q$
$Nextlab \triangleq \exists\, q \in nodes \times nodes : ML(q)$

$Sortie \triangleq p \notin output$

Fin 4