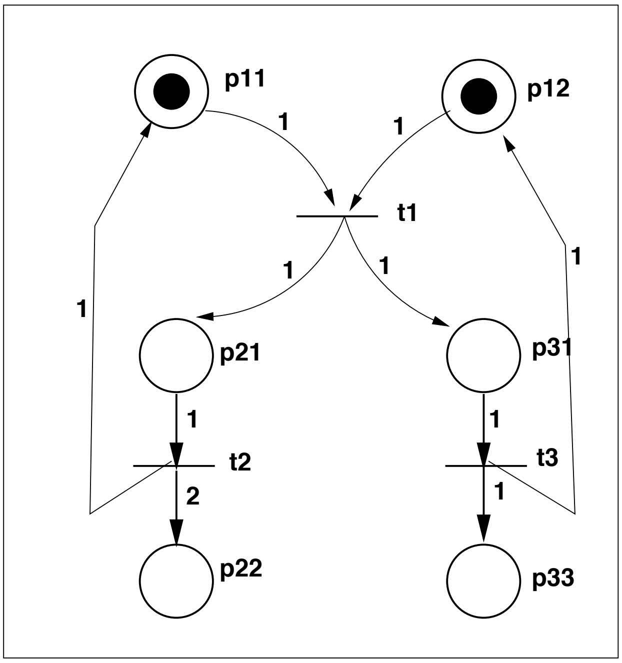


.

<

Cours Modélisation et vérification des systèmes informatiques
Exercices
Modélisation TLA^{+} (2)
par Dominique Méry
1er décembre 2025

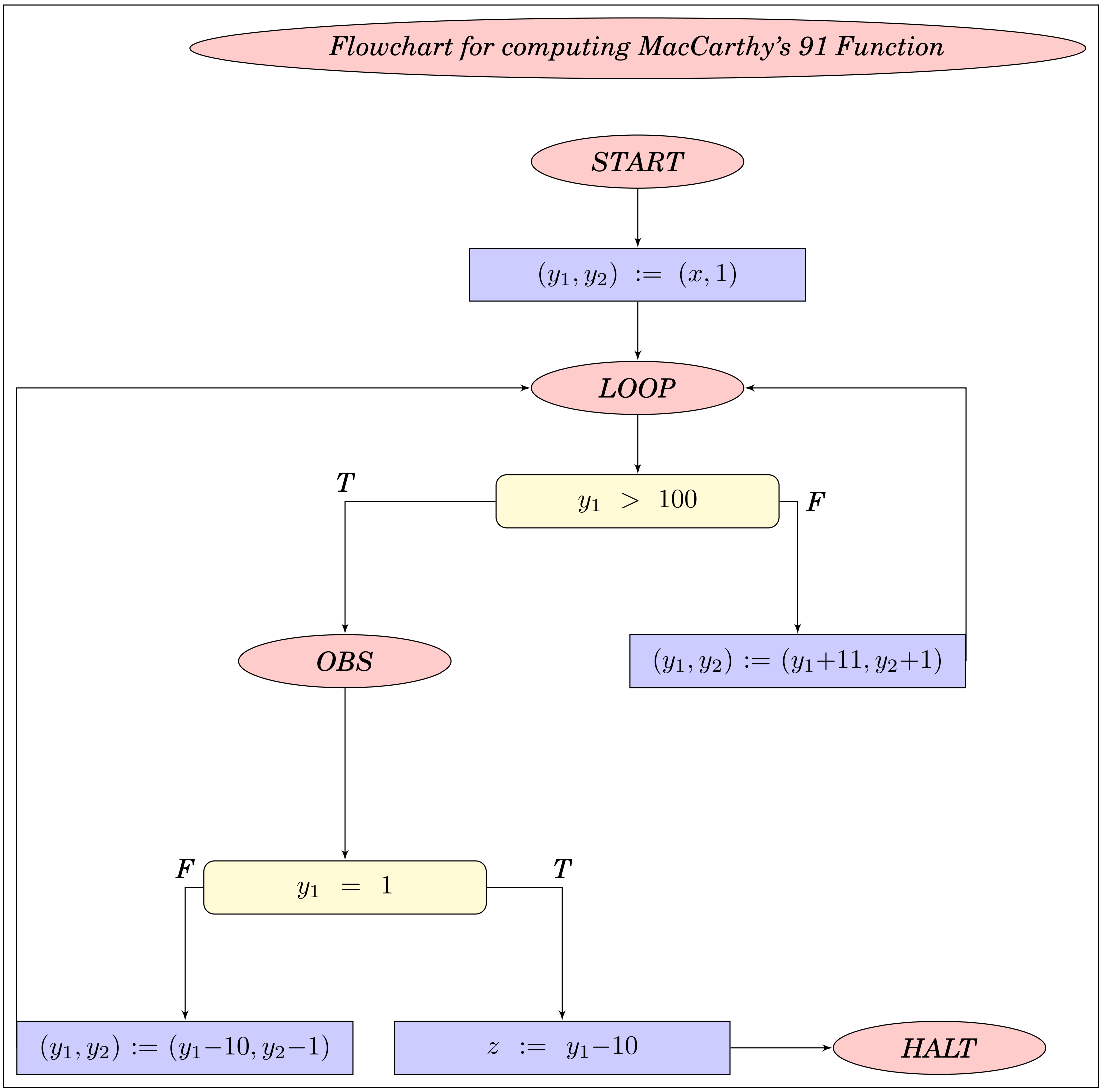
Exercice 1 *Soit le réseau de Petri de la figure 1.*

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.*

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.*

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

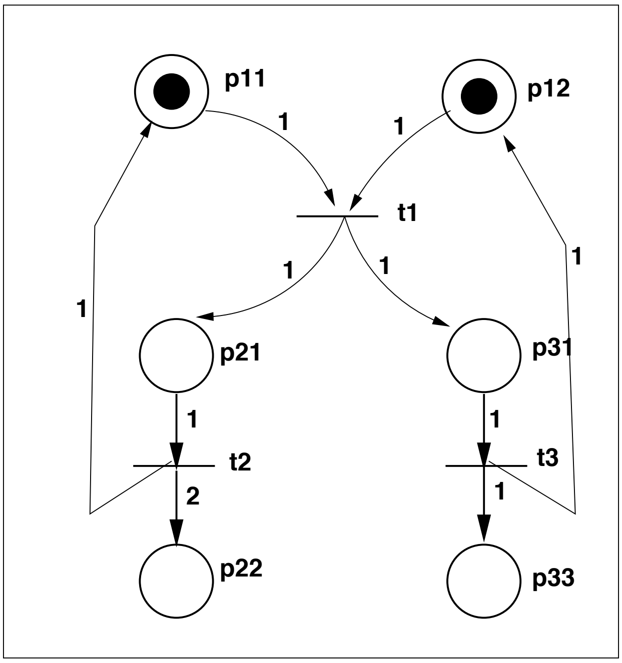
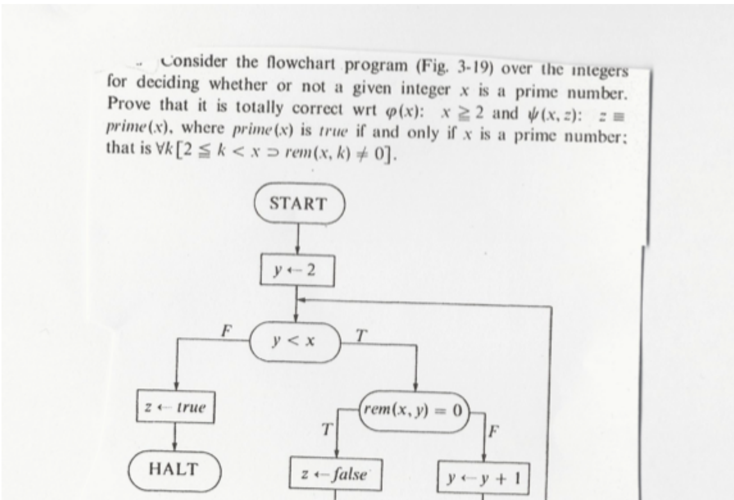


FIGURE 1 – Réseau de Petri



Consider the flowchart program (Fig. 3-19) over the integers for deciding whether or not a given integer x is a prime number. Prove that it is totally correct wrt $\varphi(x)$: $x \geq 2$ and $\psi(x, z)$: $z \equiv prime(x)$, where $prime(x)$ is *true* if and only if x is a prime number; that is $\forall k[2 \leq k < x \supset rem(x, k) \neq 0]$.

Figure 3-19 Flowchart program for deciding whether or not x is a prime number.

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*

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

MODULE *Naturals*

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

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

MODULE *TLC*

LOCAL INSTANCE *Naturals*
LOCAL INSTANCE *Sequences*

$$
\begin{array}{l}
\mathit{Print}(\mathit{out},\ \mathit{val}) \triangleq \mathit{val} \\
\mathit{PrintT}(\mathit{out}) \triangleq \mathit{TRUE} \\
\mathit{Assert}(\mathit{val},\ \mathit{out}) \triangleq \text{IF } \mathit{val} = \mathit{TRUE} \text{ THEN } \mathit{TRUE} \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{ELSE CHOOSE } v : \mathit{TRUE} \\
\mathit{JavaTime} \triangleq \text{CHOOSE } n : n \in \mathit{Nat} \\
\mathit{TLCGet}(i) \triangleq \text{CHOOSE } n : \mathit{TRUE} \\
\mathit{TLCSet}(i,\ v) \triangleq \mathit{TRUE}
\end{array}
$$

$$
\begin{array}{l}
d :> e \triangleq [x \in \{d\} \mapsto e] \\
f \ @@\ g \triangleq [x \in (\text{DOMAIN } f) \cup (\text{DOMAIN } g) \mapsto \\
\qquad\qquad\quad \text{IF } x \in \text{DOMAIN } f \text{ THEN } f[x] \text{ ELSE } g[x]] \\
\mathit{Permutations}(S) \triangleq \\
\quad \{f \in [S \rightarrow S] : \forall\, w \in S : \exists\, v \in S : f[v] = w\}
\end{array}
$$

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

$$
\begin{array}{l}
\mathit{SortSeq}(s,\ \mathit{Op}(\_,\ \_)) \triangleq \\
\quad \text{LET } \mathit{Perm} \triangleq \text{CHOOSE } p \in \mathit{Permutations}(1 \,..\, \mathit{Len}(s)) : \\
\qquad\qquad\qquad\qquad \forall\, i,\ j \in 1..\mathit{Len}(s) : \\
\qquad\qquad\qquad\qquad\quad (i < j) \Rightarrow \mathit{Op}(s[p[i]],\ s[p[j]]) \lor (s[p[i]] = s[p[j]]) \\
\quad \text{IN } \ [i \in 1..\mathit{Len}(s) \mapsto s[\mathit{Perm}[i]]]
\end{array}
$$

$$\mathit{RandomElement}(s) \triangleq \text{CHOOSE } x \in s : \mathit{TRUE}$$

$$\mathit{Any} \triangleq \text{CHOOSE } x : \mathit{TRUE}$$

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

$$\mathit{TLCEval}(v) \triangleq v$$

FIGURE 2 – Modules Naturals.tla et TLC.tla

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*

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

$$
\begin{array}{l}
n \triangleq 10 \\
nodes \triangleq 1..n \\
l \triangleq [i \in 1..n \mapsto \text{IF } i = 1 \text{ THEN } \{4, 5\} \text{ ELSE} \\
\qquad\qquad\qquad \text{IF } i = 2 \text{ THEN } \{6, 7, 10\} \text{ ELSE} \\
\qquad\qquad\qquad \text{IF } i = 4 \text{ THEN } \{7, 8\} \text{ ELSE} \\
\qquad\qquad\qquad \text{IF } i = 5 \text{ THEN } \{\} \text{ ELSE} \\
\qquad\qquad\qquad \text{IF } i = 6 \text{ THEN } \{4\} \text{ ELSE} \\
\qquad\qquad\qquad \text{IF } i = 7 \text{ THEN } \{5\} \text{ ELSE} \\
\qquad\qquad\qquad \text{IF } i = 8 \text{ THEN } \{5, 2\} \text{ ELSE} \\
\qquad\qquad\qquad \{\} \\
\qquad\qquad\qquad]
\end{array}
$$

$$
\begin{array}{l}
Init \triangleq p = 1 \\
M(i) \triangleq \land\, i \in l[p] \\
\qquad\quad\; \land\, p' = i \\
Next \triangleq \exists\, i \in 1..n : M(i)
\end{array}
$$

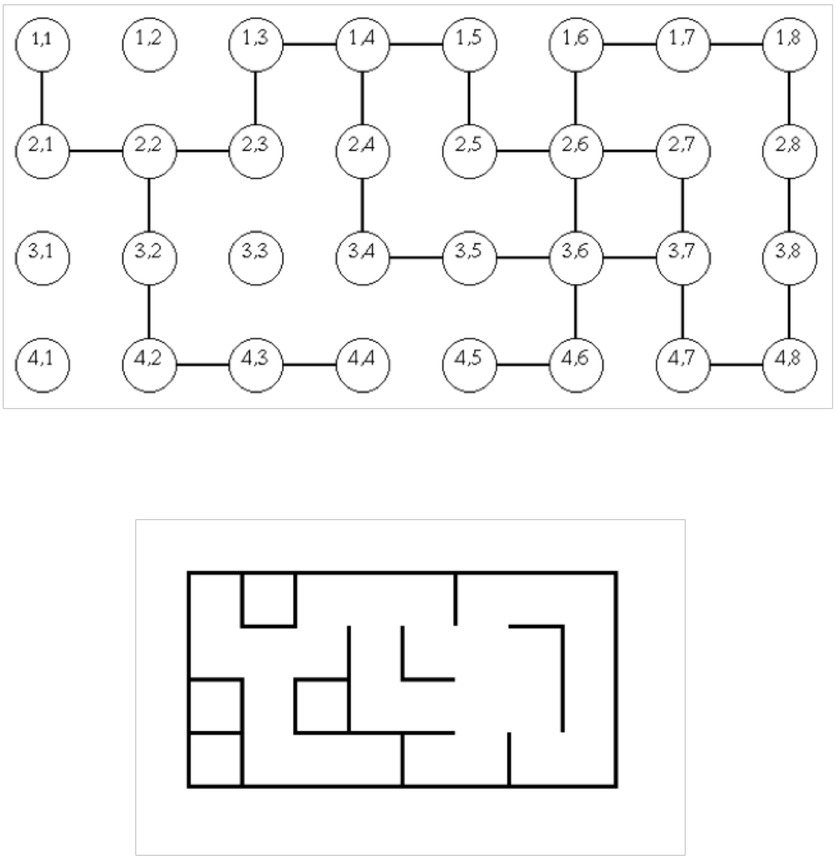


FIGURE 3 – Labyrinthe