Cours Modélisation et vérification des systèmes informatiques
Exercices (avec les corrections)
Annotation, Contrat, modélisation et vérification
par Dominique Méry
9 octobre 2024

Exercice 1 *Soit le contrat suivant :*

variables $int\ X, int\ Y, int\ Z$
requires $P(x_0, y_0, z_0)$
ensures $Q(x_0, y_0, z_0, x_f, y_f, z_f)$
begin
$//\ 0 : R_0(x_0, y_0, z_0, x, y, z)$
$\mathtt{X} = \mathsf{g}(\mathtt{X}, \mathtt{Y}, \mathtt{Z})$
$//\ f : R_f(x_0, y_0, z_0, x, y, z)$
end

- g *est une fonction arithmétique définie sur le type des entiers* `int` *et conduit à un prédicat de typage* $x, y, y \in \mathbb{Z}$ *(ou* $\mathbb{Z}(x, y, z)$*).*
- $P(x_0, y_0, z_0)$ *définit la précondition c'est-à-dire les conditions que doivent satisfaire les valeurs initiales des variables* `X`*,* `Y`*,* `Z`*.*
- $Q(x_0, y_0, z_0, x_f, y_f, z_f)$ *définit la postcondition c'est-à-dire la relation que doit satisfaire les valeurs initiales et les valeurs finales des variables* `X`*,* `Y`*,* `Z`*.*
- $R_0(x_0, y_0, z_0, x, y, z)$ *et* $R_f(x_0, y_0, z_0, x, y, z)$ *définissent les conditions ou les assertions satisfaites par mes valeurs courantes des variables* `X`*,* `Y`*,* `Z`*.*

Question 1.1 *On définit*

- $P(x_0, y_0, z_0) \stackrel{def}{=} x_0, y_0, z_0 \in \mathbb{Z}$
- $g \stackrel{def}{=} \lambda u, v, w.u+v+w$
- $Q(x_0, y_0, z_0, x, y, z) \stackrel{def}{=} x_0, y_0, z0, x, y, z \in \mathbb{Z} \wedge x = x_0+y_0+z_0 \wedge y = y_0 \wedge z = z_0$
- $R_0(x_0, y_0, z_0, x, y, z) \stackrel{def}{=} x = x_0 \wedge y = y0 \wedge z = z_0 \wedge \mathbb{Z}(x_0, y_0, z0, x, y, z)$
- $R_f(x_0, y_0, z_0, x, y, z) \stackrel{def}{=} \mathbb{Z}(x_0, y_0, z0, x, y, z) \wedge x = x_0+y_0+z_0 \wedge y = y_0 \wedge z = z_0$

Ecrire les conditions de vérification de ce contrat et vérifier leur correction.

Question 1.2 *On définit*

- $P(x_0, y_0, z_0) \stackrel{def}{=} x_0, y_0, z_0 \in \mathbb{Z}$
- $g \stackrel{def}{=} \lambda u, v, w.max(u, v, w)$
- $Q(x_0, y_0, z_0, x, y, z) \stackrel{def}{=} x_0, y_0, z_0 \in \mathbb{Z} \wedge x, y, z \in \mathbb{Z} \wedge x = max(x_0, y_0, z_0) \wedge y = y_0 \wedge z = z_0$
- $R_0(x_0, y_0, z_0, x, y, z) \stackrel{def}{=} \mathbb{Z}(x_0, y_0, z_0, x, y, z)$

Ecrire les conditions de vérification de ce contrat et vérifier leur correction. En particulier, il faut définir une relation $R_f(x_0, y_0, z_0, x, y, z)$ *permettant d'établir la correction selon les règles.*

Exercice 2 *Montrer que chaque annotation est correcte ou incorrecte selon les conditions de vérifications énoncées comme suit :*
$\forall v, v'.P_\ell(v) \ \wedge \ cond_{\ell,\ell'}(v) \ \wedge \ v' = f_{\ell,\ell'}(v) \Rightarrow P_{\ell'}(v')$
Cette condition s'écrit initialement :
$\forall v, v', pc, pc'.pc = \ell \wedge P_\ell(v) \ \wedge \ pc = \ell \wedge \ cond_{\ell,\ell'}(v) \ \wedge \ v' = f_{\ell,\ell'}(v) \wedge \ pc' = \ell' \Rightarrow pc' = \ell' \wedge \ P_{\ell'}(v')$
mais on peut réduire en oubliant la variable pc.

— $\ell_1 : x = 10 \ \wedge\ y = z{+}x \ \wedge z = 2{\cdot}x$
$y := z{+}x$
$\ell_2 : x = 10 \ \wedge\ y = x{+}2{\cdot}10$

— $\ell_1 : x = 1 \ \wedge\ y = 12$
$x := 2{\cdot}y$
$\ell_2 : x = 1 \ \wedge\ y = 24$

— *On suppose que p est un nombre premier :*
$\ell_1 : x = 2^p \ \wedge\ y = 2^{p+1} \ \wedge\ x{\cdot}y = 2^{2\cdot p+1}$
$x := y{+}x{+}2^x$
$\ell_2 : x = 5{\cdot}2^p \ \wedge\ y = 2^{p+1}$

— $\ell_1 : x = 11 \ \wedge\ y = 13$
$z := x; x := y; y := z;$
$\ell_2 : x = 26/2 \ \wedge\ y = 33/3$

Exercice 3 ☑

Montrer que chaque annotation est correcte ou incorrecte selon les conditions de vérifications énoncées comme suit

$\forall x, y, , x', y'.P_\ell(x,y) \ \wedge\ cond_{\ell,\ell'}(x,y) \ \wedge\ (x',y') = f_{\ell,\ell'}(x,y) \Rightarrow P_{\ell'}(x',y')$

— *(1)* $\ell_1 : x = 9 \ \wedge\ y = z{+}x$
$y := x{+}9$
$\ell_2 : x = 9 \ \wedge\ y = x{+}9$

— $\ell_1 : x = 3 \ \wedge\ y = 3$
$x := y{+}x$
$\ell_2 : x = 6 \ \wedge\ y = 3$

— *(2)* $\ell_1 : x = 1 \ \wedge\ y = 3 \ \wedge\ x{+}y = 12$
$x := y{+}x$
$\ell_2 : x = 567 \ \wedge\ y = 34$

— $\ell_1 : x = 1 \ \wedge\ y = 3$
$z := x; x := y; y := z;$
$\ell_2 : x = 3 \ \wedge\ y = 1$

1. $c = \ell_1 \wedge x = 9 \ \wedge\ y = z{+}x \ \wedge\ \mathbf{TRUE} \ \wedge\ (x',y',c') = (x, x{+}9, \ell_2) \Rightarrow c' = \ell_2 \wedge x' = 9 \ \wedge\ y' = x'{+}9$ *:*
   (a) $c = \ell_1 \wedge x = 9 \ \wedge\ y = z{+}x \ \wedge c' = \ell_2 \Rightarrow c' = \ell_2 \wedge x = 9 \ \wedge\ x{+}9 = x{+}9$
   (b) $c = \ell_1 \wedge x = 9 \ \wedge\ y = z{+}x \ \wedge c' = \ell_2 \Rightarrow x = 9 \ \wedge\ x{+}9 = x{+}9$
   (c) *CORRECT*
2. $c = \ell_1 \wedge x = 1 \ \wedge\ y = 3 \ \wedge\ x{+}y = 12 \ \wedge \mathbf{TRUE} \ \wedge\ (x',y',c') = (y{+}x, y, \ell_2) \Rightarrow c' = \ell_2 \wedge x' = 567 \ \wedge\ y' = 34$ *:*
   (a) $c = \ell_1 \wedge x = 1 \ \wedge\ y = 3 \ \wedge\ x{+}y = 12 \ \ \wedge\ (x',y',c') = (y{+}x, y, \ell_2) \Rightarrow c' = \ell_2 \wedge y{+}x = 567 \ \wedge\ y = 34$
   (b) $c = \ell_1 \wedge x = 1 \ \wedge\ y = 3 \ \wedge\ x{+}y = 12 \ \ \wedge\ c' = \ell_2 \Rightarrow c' = \ell_2 \wedge y{+}x = 567 \ \wedge\ y = 34$
   (c) $c = \ell_1 \wedge x = 1 \ \wedge\ y = 3 \ \wedge\ x{+}y = 12 \ \ \wedge\ c' = \ell_2 \Rightarrow x{+}y = 4 \wedge x{+}y = 12$
   (d) $c = \ell_1 \wedge x = 1 \ \wedge\ y = 3 \ \wedge\ x{+}y = 12 \ \ \wedge\ c' = \ell_2 \Rightarrow \mathbf{FALSE}$
   (e) $\mathbf{FALSE} \Rightarrow c' = \ell_2 \wedge y{+}x = 567 \ \wedge\ y = 34$
   (f) *CORRECT*
3. $c = \ell_1 \wedge x = 3 \ \wedge\ y = 3 \wedge \mathbf{TRUE} \ \wedge\ (x',y',c') = (y{+}x, y, \ell_2) \Rightarrow c' = \ell_2 \wedge x' = 6 \ \wedge\ y' = 3$
   (a) $c = \ell_1 \wedge x = 3 \ \wedge\ y = 3 \ \wedge\ c' = \ell_2 \Rightarrow c' = \ell_2 \wedge y{+}x = 6 \ \wedge\ y = 3$
   (b) $c = \ell_1 \wedge x = 3 \ \wedge\ y = 3 \ \wedge\ c' = \ell_2 \Rightarrow c' = \ell_2 \wedge y{+}x = 6 \ \wedge\ y = 3$
   (c) $c = \ell_1 \wedge x = 3 \ \wedge\ y = 3 \ \wedge\ c' = \ell_2 \Rightarrow c' = \ell_2 \wedge 6 = 6 \ \wedge\ y = 3$
   (d) *CORRECT*
4. $c = \ell_1 \wedge x = 1 \ \wedge\ y = 3 \ \mathbf{TRUE} \ \wedge\ (x',y',z',c') = (y, x, x, \ell_2) \Rightarrow c' = \ell_2 \wedge x' = 3 \ \wedge\ y' = 1$
   (a) $c = \ell_1 \wedge x = 1 \ \wedge\ y = 3 \ \mathbf{TRUE} \ \wedge\ (x',y',z',c') = (y, x, x, \ell_2) \Rightarrow c' = \ell_2 \wedge y = 3 \ \wedge\ x = 1$
   (b) *CORRECT*

Exercice 4 ☑

Montrer que chaque annotation est correcte ou incorrecte selon les conditions de vérifications énoncées comme suit

$\forall x, y, , x', y'. P_\ell(x, y) \ \wedge \ cond_{\ell,\ell'}(x, y) \ \wedge \ (x', y') = f_{\ell,\ell'}(x, y) \Rightarrow P_{\ell'}(x', y')$

— $\ell_1 : x = 3 \ \wedge \ y = z+x \ \wedge z = 2\cdot x$
  $y := z+x$
  $\ell_2 : x = 3 \ \wedge \ y = x+6$

— $\ell_1 : x = 2^4 \ \wedge \ y = 2^{345} \ \wedge \ x\cdot y = 2^{350}$
  $x := y+x+2^x$
  $\ell_2 : x = 2^{56} \ \wedge \ y = 2^{345}$

— $\ell_1 : x = 1 \ \wedge \ y = 12$
  $x := 2\cdot y+x$
  $\ell_2 : x = 1 \ \wedge \ y = 25$

Exercice 5 ☑

Soit le petit algorithme annoté suivant :

$l0 : \{v = 3\}$
$v := v+2$
$l1 : \{v = 5\}$

Ecrire un module TLA$^+$ explicitant la relation de transition, les conditions initiales, l'invariant et la propriété de sûreté pour la correction partielle.

◇– **Solution de l'exercice 5**

MODULE an_1

EXTENDS *Integers*, *TLC*

VARIABLES v, pc

$titi \triangleq pc = \text{"l0"} \wedge v = 3$

$skip \triangleq \text{UNCHANGED } \langle pc, v\rangle$
$skip_2 \triangleq pc' = pc \wedge v' = v$

$trans \triangleq pc = \text{"l0"} \wedge TRUE \wedge pc' = \text{"l1"} \wedge v' = v+2$
$trans_2 \triangleq pc = \text{"l0"} \wedge pc' = \text{"l1"} \wedge v' = v+2$
$trans_3 \triangleq$
  $\wedge\, pc = \text{"l0"} \wedge TRUE$
  $\wedge\, pc' = \text{"l1"}$
  $\wedge\, v' = v+2$

$toto \triangleq skip \vee trans$

$i \triangleq$
  $\wedge\, pc \in \{\text{"l0"}, \text{"l1"}\}$
  $\wedge\, pc = \text{"l0"} \Rightarrow v = 3$
  $\wedge\, pc = \text{"l1"} \Rightarrow v = 6$

$safety \triangleq pc = \text{"l1"} \Rightarrow v = 5$

Fin 5

Exercice 6 ☑

Définir les conditions de vérification de la correction partielle pour les structures suivantes. Définir un modèle TLA⁺ pour vérifier la bonne annotation.

Question 6.1

$$\begin{array}{l} \ell 1 : \{P_{\ell 1}(x,y)\} \\ \quad x := x{+}y{+}7; \\ \ell 2 : \{P_{\ell 2}(x,y)\} \end{array}$$

◇— **Solution de la question 6.1**

Conditions de vérification pour la correction partielle pc *désigne la variable de contrôle.*

$$pc = \ell 1 \land P_{\ell 1}(x,y) \land pc = \ell 1 \land pc' = \ell 2 \land x' = x{+}y{+}7 \land y' = y \Rightarrow pc' = \ell 2 \land P_{\ell 2}(x',y')$$

qui se simplifie en :

$$P_{\ell 1}(x,y) \land x' = x{+}y{+}7 \land y' = y \Rightarrow P_{\ell 2}(x',y')$$

qui se simplifie en :

$$P_{\ell 1}(x,y) \land x' = x{+}y{+}7 \Rightarrow P_{\ell 2}(x',y)$$

qui se simplifie en :

$$P_{\ell 1}(x,y) \Rightarrow P_{\ell 2}(x{+}y{+}7,y)$$

MODULE $an2$

Modèle TLA⁺ pour vérifier la bonne annotation EXTENDS *Naturals*

VARIABLES x, y, pc

Define actions from the text of annotated algorithm

$al0l_1 \triangleq$
$\quad \land pc = \text{"l0"}$
$\quad \land pc' = \text{"l1"}$
$\quad \land x' = x{+}y{+}7$
$\quad \land y' = y$

Define the computation relation

$next \triangleq al0l1$

Define the initial conditions

$init \triangleq pc = \text{"l0"} \land x = 3 \land y = 8$

Define the invariant from the annotation

$i \triangleq$

$\land pc = \text{"l0"} \Rightarrow x = 3 \land y = 8$

$\land pc = \text{"l1"} \Rightarrow x = 6 \land y = 89$

Define the safety property to check namely the partial correctness

$safe \triangleq pc = \text{"l1"} \Rightarrow x = 7 \land y = 89$

Modification History

Last modified Tue Dec 15 17 :30 :19 CET 2015 by mery

Created Wed Sep 09 17 :02 :47 CEST 2015 by mery

MODULE $an2$

EXTENDS *Naturals*

VARIABLES x, y, pc

Define actions from the text of annotated algorithm

$al0l_1 \triangleq$

$\land pc = \text{"l0"}$

$\land pc' = \text{"l1"}$

$\land x' = x+y+7$

$\land y' = y$

Define the computation relation

$next \triangleq al0l1$

Define the initial conditions

$init \triangleq pc = \text{"l0"} \land x = 3 \land y = 8$

Define the invariant from the annotation

$i \triangleq$

$\land pc = \text{"l0"} \Rightarrow x = 3 \land y = 8$

$\land pc = \text{"l1"} \Rightarrow x = 18 \land y = 8$

Define the safety property to check namely the partial correctness

$safe \triangleq pc = \text{"l1"} \Rightarrow x = 18 \land y = 8$

$prop \triangleq i \Rightarrow safe$

$Init \triangleq init$

$Next \triangleq next$

$principe \triangleq init \Rightarrow \land prop$

Modification History

Last modified Wed Sep 21 13 :28 :17 CEST 2016 by mery

Created Wed Sep 09 17 :02 :47 CEST 2015 by mery

Fin 6.1

Question 6.2

$$\begin{array}{l} \ell : \{P_\ell(x,y)\} \\ \quad x, y := y, x; \\ \ell' : \{P_{\ell'}(x,y)\} \end{array}$$

◇— **Solution de la question 6.2**

Conditions de vérification pour la correction partielle *c désigne la variable de contrôle.*

$$c = \ell 1 \wedge P_{\ell 1}(x,y) \wedge c' = \ell 2 \wedge (x',y') = (y,x) \Rightarrow c' = \ell 2 \wedge P_{\ell 2}(x',y')$$

qui se simplifie en :

$$P_{\ell 1}(x,y) \wedge x' = y \wedge y' = x \Rightarrow P_{\ell 2}(x',y')$$

qui se simplifie en :

$$P_{\ell 1}(x,y) \Rightarrow P_{\ell 2}(y,x)$$

MODULE *an3*

Modèle TLA$^+$ pour vérifier la bonne annotation EXTENDS *Naturals*

CONSTANTS a, b

VARIABLES x, y, pc

Define actions from the text of annotated algorithm

$al1l_2 \triangleq$

$\quad \wedge\ pc = \text{"l1"}$

$\quad \wedge\ pc' = \text{"l2"}$

$\quad \wedge\ x' = y \ \wedge\ y' = x$

$\quad newaction \triangleq pc = \text{"l2"} \wedge pc' = \text{"l1"} \wedge x' = x \wedge y' = y$

Define the computation relation

$next \triangleq al1l2$

$newnext \triangleq al1l_2 \vee newaction$

Define the initial conditions

$init \triangleq pc = \text{"l1"} \wedge x = a \wedge y = b$

Define the invariant from the annotation

$i \triangleq$

$\quad \wedge\ pc = \text{"l1"} \Rightarrow x = a \wedge y = b$

$\quad \wedge\ pc = \text{"l2"} \Rightarrow x = b \wedge y = a$

Define the safety property to check namely the partial correctness

$safe \triangleq pc = \text{"l2"} \Rightarrow x = b \wedge y = a$

Fin 6.2

Exercice 7 ☑

Déterminer les conditions de vérification pour la structure de boucle bornée.
On suppose que S ne modifie pas i.

$\ell_1 : \{P_{\ell_1}(x)\}$
  FOR $i := 1$ ***TO*** n ***DO***
    $\ell_2 : \{P_{\ell_2}(i, x)\}$
      $S(x);$
    $\ell_3 : \{P_{\ell_3}(i, x)\}$
  ENDFOR
$\ell_4 : \{P_{\ell_4}(x)\}$

⋄— **Solution de la question 7.0**

(1) $c = \ell_1 \wedge P_{\ell_1}(x) \wedge 1 \leq n \wedge c' = \ell_2 \wedge i' = 1 \wedge x' = x \Rightarrow c' = \ell_2 \wedge P_{\ell_2}(i', x')$
(2) $c = \ell_1 \wedge P_{\ell_1}(x) \wedge \neg(1 \leq n) \wedge c' = \ell_4 \wedge x' = x \Rightarrow c' = \ell_4 \wedge P_{\ell_4}(x')$
(3) $c = \ell_3 \wedge P_{\ell_3}(x, i) \wedge i{+}1 \leq n \wedge c' = \ell_2 \wedge i' = i{+}1 \wedge x' = x \Rightarrow c' = \ell_2 \wedge P_{\ell_2}(i', x')$
(4) $c = \ell_2 \wedge P_{\ell_3}(x, i) \wedge \neg(i{+}1 \leq n) \wedge c' = \ell_4 \wedge x' = x \wedge i' = i{+}1 \Rightarrow c' = \ell_4 \wedge P_{\ell_4}(x')$

Fin 7.0

Exercice 8 ☑

Variables : X,Y,Z
Requires : $x_0, y_0 \in \mathbb{N} \ \wedge z_0 \in \mathbb{Z}$
Ensures : $z_f = max(x_0, y_0)$

$\ell_0 : \{\ldots\}$
if $X < Y$ **then**
  $\ell_1 : \{\ldots\}$
  $Z := Y;$
  $\ell_2 : \{\ldots\}$
else
  $\ell_3 : \{\ldots\}$
  $Z := X;$
  $\ell_4 : \{\ldots\}$
;
$\ell_5 : \{\ldots\}$

Algorithme 1: maximum de deux nombres non annotée

Question 8.1 *Compléter l'algorithme 8 en l'annotant.*

⋄— **Solution de la question 8.1**

Annotation *L'annotation de cet algorithme est donnée à la référence d'algorithme 8 et la figure est placée au gré de LaTeX.*

Fin 8.1

Question 8.2 *Vérifier la bonne annotation, en appliquant les règles de correction partielle.*

Question 8.3 *Vérifier la bonne annotation, en traduisant l'algorithme annoté en un module TLA.*

Variables : X,Y,Z
Requires : $x_0, y_0 \in \mathbb{N} \wedge z_0 \in \mathbb{Z}$
Ensures : $z_f = max(x_0, y_0)$

$\ell_0 : \{x = x_0 \wedge y = y_0 \wedge z = z_0 \wedge x_0, y_0 \in \mathbb{N} \wedge z_0 \in \mathbb{Z}\}$
if $X < Y$ **then**
  $\ell_1 : \{x < y \wedge x = x_0 \wedge y = y_0 \wedge z = z_0 \wedge x_0, y_0 \in \mathbb{N} \wedge z_0 \in \mathbb{Z}\}$
  $Z := Y;$
  $\ell_2 : \{x < y \wedge x = x_0 \wedge y = y_0 \wedge x_0, y_0 \in \mathbb{N} \wedge z_0 \in \mathbb{Z} \wedge z = y_0\}$
else
  $\ell_3 : \{x \geq y \wedge x = x_0 \wedge y = y_0 \wedge z = z_0 \wedge x_0, y_0 \in \mathbb{N} \wedge z_0 \in \mathbb{Z}\}$
  $Z := X;$
  $\ell_4 : \{x \geq y \wedge x = x_0 \wedge y = y_0 \wedge x_0, y_0 \in \mathbb{N} \wedge z_0 \in \mathbb{Z} \wedge z = x_0\}$
;
$\ell_5 : \{z = max(x_0, y_0) \wedge x = x_0 \wedge y = y_0 \wedge x_0, y_0 \in \mathbb{N} \wedge z_0 \in \mathbb{Z}\}$

Algorithme 2: maximum de deux nombres non annotée

⋄— **Solution de la question 8.3**
Modèle TLA$^+$ pour vérifier la bonne annotation

```
-------------------------------- MODULE appex3_77 --------------------------------
EXTENDS Naturals, Integers
CONSTANTS x0,y0,z0
VARIABLES  x,y,z,pc
ASSUME x0 \in Nat /\ y0 \in Nat
typeInt(u) == u \in Int
maxi(u,v) == IF u < v THEN v ELSE u
pre ==  x0 \in Nat /\ y0 \in Nat /\ z0 \in Int
----------------------------------------------------------------------
al0l1 ==
    /\ pc="l0"
    /\ pc'="l1"
    /\ x<y
    /\ z'=z /\ x'=x /\ y'=y
al1l2 ==
    /\ pc="l1"
    /\ pc'="l2"
    /\ z'=y
    /\ x'=x /\ y'=y
al2l5 ==
    /\ pc="l2"
    /\ pc'="l5"
    /\ z'=z /\ x'=x /\ y'=y
al0l3 ==
    /\ pc="l0"
    /\ pc'="l3"
    /\ x \geq y
    /\ z'=z /\ x'=x /\ y'=y
al3l4 ==
    /\ pc="l3"
    /\ pc'="l4"
```

```
    /\ z'=x
    /\ x'=x /\ y'=y
  al4l5 ==
    /\ pc="l4"
    /\ pc'="l5"
    /\ z'=z /\ x'=x /\ y'=y
---------------------
Next == al0l1 \/ al1l2 \/ al2l5  \/ al0l3 \/ al3l4 \/ al4l5 \/ UNCHANGED <<x,y,z,pc
Init == pc="l0" /\ x=x0 /\ y =y0 /\ z = z0
----------------------
i ==
    /\ typeInt(x) /\ typeInt(y) /\ typeInt(z)
    /\ pc="l0" =>  x=x0 /\ y=y0 /\ z=z0 /\ pre
    /\ pc="l1" => x<y /\ x=x0 /\ y=y0 /\ z=z0 /\ pre
    /\ pc="l2" => x<y /\ x=x0 /\ y=y0 /\ z=y0 /\ pre
    /\ pc="l3" => x \geq y /\ x=x0 /\ y=y0 /\ z=z0 /\ pre
    /\ pc="l4" => x \geq y /\ x=x0 /\ y=y0 /\ z=x0  /\ pre
    /\ pc="l5" => z = maxi(x0,y0) /\ x=x0 /\ y=y0  /\ pre
safe ==  pc="l5" =>  z = maxi(x0,y0)
safeab == x=x0 /\ y=y0
==============================================================================
\* Modification History
\* Last modified Wed Sep 29 20:32:22 CEST 2021 by mery
\* Created Wed Sep 09 18:19:08 CEST 2015 by mery
```

Fin 8.3

Question 8.4 *Enoncer et vérifier la correction partielle.*

◇— **Solution de la question 8.4**

Il suffit de donner tout d'abord la précondition et la postcondition et de vérifier les conditoons de vérifications de la correction partielle.

Fin 8.4

Exercice 9 ☑

Il s'agit d'étudier et d'annoter le programme proposé en vu d'obtenir sa correction partielle (c'est-à-dire sans la preuve de terminaison). On appelle état un ensemble de valeurs précises (spécifié par un prédicat) des variables du programme, nous allons considérer une étiquette (ℓ) entre chaque instruction du programme considéré. On appelle une annotation le prédicat décrivant les valeurs possibles des variables pour un état du programme. Cette annotation est notée : $P_\ell(v)$ et exprime la propriété satisfaite par la variable v en ℓ.

Question 9.1 *On vous demande :*

1. *d'annoter toutes les étiquettes du programme*
2. *de proposer un modèle TLA$^+$ pour vérifier les annotations et la correction partielle*

Question 9.2 *Vérifier les conditions à vérifier pour montrer que l'annotation est valide et qu'elle montre la correction partielle.*

Question 9.3 *Traduire l'algorithme annoté en un module TLA comportant la définition de l'algorithme sous forme de Next et Init et comportant la définition de l'invariant défini par les annotations et les définitions de la correction partielle et l'absence d'erreurs à l'exécution.*

◇— **Solution de l'exercice 9**

Variables : X
Requires : $x_0 \in \mathbb{N}$
Ensures : $x_f = 0$

$\ell_0 : \{\ldots\}$
while $0 < X$ **do**
    $\ell_1 : \{\ldots\}$
    $X := X - 1;$
    $\ell_2 : \{\ldots\}$
;
$\ell_3 : \{\ldots\}$

Algorithme 3: Exemple non annoté

Variables : X
Requires : $x_0 \in \mathbb{N}$
Ensures : $x_f = 0$

$\ell_0 : \{\ x = x_0 \wedge x_0 \in \mathbb{N}\}$
while $0 < X$ **do**
    $\ell_1 : \{0 < x \leq x_0 \wedge x_0 \in \mathbb{N}\}$
    $X := X - 1;$
    $\ell_2 : \{0 \leq x \leq x_0 \wedge x_0 \in \mathbb{N}\}$
;
$\ell_3 : \{x = 0\}$

Algorithme 4: exemple annoté

Annotation L'annotation (cf algorithme) est construite par propagation des assertions selon les instructions. Il faut ensuite vérifier que les conditions sont vraies.

MODULE $ex3$

Modèle TLA$^+$ pour vérifier la bonne annotation EXTENDS *Naturals*

CONSTANTS $x0$
VARIABLES x, pc

$al0l_1 \triangleq$
$\land\, pc = \text{"l0"}$
$\land\, pc' = \text{"l1"}$
$\land\, 0 < x$
$\land\, x' = x$

$al0l_3 \triangleq$
$\land\, pc = \text{"l0"}$
$\land\, pc' = \text{"l3"}$
$\land\, x = 0$
$\land\, x' = x$

$al1l_2 \triangleq$
$\land\, pc = \text{"l1"}$
$\land\, pc' = \text{"l2"}$
$\land\, x' = x-1$

$al2l_1 \triangleq$
lgforithme $\land\, pc = \text{"l2"}$
$\land\, pc' = \text{"l1"}$
$\land\, 0 < x$
$\land\, x' = x$

$al2l_3 \triangleq$
$\land\, pc = \text{"l2"}$
$\land\, pc' = \text{"l3"}$
$\land\, 0 = x$
$\land\, x' = x$

$next \triangleq al0l_1 \lor al0l_3 \lor al1l_2 \lor al2l_1 \lor al2l3$

$init \triangleq pc = \text{"l0"} \land x = x0$

$i \triangleq$
$\land\, pc = \text{"l0"} \Rightarrow x = x0$
$\land\, pc = \text{"l1"} \Rightarrow 0 < x \land x \leq x_0$
$\land\, pc = \text{"l2"} \Rightarrow 0 \leq x \land x \leq x0$
$\land\, pc = \text{"l3"} \Rightarrow x = 0$

$safe \triangleq pc = \text{"l3"} \Rightarrow x = 0$

$safeplus \triangleq x \geq 0$

Modification History

Last modified Thu Sep 10 09 :35 :48 CEST 2015 by mery

Created Wed Sep 09 18 :07 :50 CEST 2015 by mery

Fin 9

Exercice 10 Question 10.1 *Soit un tableau t (dans $\mathbb{N}$), donner un prédicat max$(m, t, a, b) = \ldots$ exprimant qu'un nombre $m \in \mathbb{N}$ est le maximum de ce tableau t dans l'intervalle $a..b$.*

◇— **Solution de l'exercice 10**

$$\boldsymbol{max}(m, t, a, b) \stackrel{def}{=} m \in \mathrm{ran}(t) \wedge (\forall i \cdot i \in a..b \Rightarrow t(i) \leq m)$$

Fin 10

Question 10.2 *De même pour trié(t, a, b), donnez un prédicat spécifiant que le tableau t est trié dans l'intervalle $a..b$.*

◇— **Solution de l'exercice 10**

$$\boldsymbol{trié}(t, a, b) \stackrel{def}{=} \forall i, j \cdot ((i \in a..b \wedge j \in a..b \wedge i \leq j) \Rightarrow t(i) \leq t(j))$$

Fin 10

Exercice 11 *Dans l'algorithm 11, on calcule le maximum d'une suite de valeurs entières. On vous demande :*

- *Définir la précondition et la postcondition.*
- *Annoter cet algorithme*
- *Vérifier les conditions de vérification pour la correction partielle*
- *Vérifier les conditions pour l'absence d'erreurs à l'exécution*

Variables : F,N,M,I

Requires : $\begin{pmatrix} n_0 \in \mathbb{N} \wedge \\ n_0 \neq 0 \wedge \\ f_0 \in 0..n_0-1 \rightarrow \mathbb{N} \end{pmatrix}$

Ensures : $\begin{pmatrix} m_f \in \mathbb{N} \wedge \\ m_f \in ran(f_0) \wedge \\ (\forall j \cdot j \in 0..n_0-1 \Rightarrow f_0(j) \leq m_f) \end{pmatrix}$

$M := F(0)$;
$I := 1$;
while $I < N$ **do**
  if *F(i)>M* **then**
    $M := F(I)$;
  ;
  $I{+}{+}$;
;

Algorithme 5: Algorithme du maximum d'une liste non annotée

◇— **Solution de l'exercice 11**

La solution de cette annotation est dans l'algorithme annoté.

MODULE *algo_maximum*

computing the maximum value of an array f

/* algorithme de calcul du maximum avec une boucle while de l'exercice 11 */
Variables : F,N,M,I

Requires : $\left(\begin{array}{l} n_0 \in \mathbb{N} \wedge \\ n_0 \neq 0 \wedge \\ f_0 \in 0 \mathinner{..} n_0{-}1 \rightarrow \mathbb{N} \end{array}\right)$

Ensures : $\left(\begin{array}{l} m_f \in \mathbb{N} \wedge \\ m_f \in ran(f_0) \wedge \\ (\forall j \cdot j \in 0 \mathinner{..} n_0{-}1 \Rightarrow f_0(j) \leq m_f) \end{array}\right)$

$$\ell_0 : \{ \left(\begin{array}{l} n_0 \in \mathbb{N} \wedge \\ n_0 \neq 0 \wedge \\ f_0 \in 0 \mathinner{..} n_0{-}1 \rightarrow \mathbb{N} \\ m_0, i_0 \in \mathbb{Z} \end{array}\right) \wedge n = n_\wedge f = f0 \wedge i = i_0 \wedge m = m_0 \}$$

$M := F(0);$

$$\ell_1 : \{ \left(\begin{array}{l} n_0 \in \mathbb{N} \wedge \\ n_0 \neq 0 \wedge \\ f_0 \in 0 \mathinner{..} n_0{-}1 \rightarrow \mathbb{N} \\ m_0, i_0 \in \mathbb{Z} \end{array}\right) \wedge n = n_\wedge f = f0 \wedge i = i_0 \wedge m = f(0) \}$$

$I := 1;$

$$\ell_2 : \{ \left(\begin{array}{l} n_0 \in \mathbb{N} \wedge \\ n_0 \neq 0 \wedge \\ f_0 \in 0 \mathinner{..} n_0{-}1 \rightarrow \mathbb{N} \\ m_0, i_0 \in \mathbb{Z} \\ n = n_0 \wedge f = f0 \end{array}\right) \wedge i = 1 \wedge \left(\begin{array}{l} m \in \mathbb{N} \wedge \\ m \in ran(f[0..i{-}1]) \wedge \\ (\forall j \cdot j \in 0 \mathinner{..} i{-}1 \Rightarrow f(j) \leq m) \end{array}\right) \}$$

while $I < N$ **do**

$$\ell_3 : \{ \left(\begin{array}{l} n_0 \in \mathbb{N} \wedge \\ n_0 \neq 0 \wedge \\ f_0 \in 0 \mathinner{..} n_0{-}1 \rightarrow \mathbb{N} \\ m_0, i_0 \in \mathbb{Z} \\ n = n_0 \wedge f = f0 \end{array}\right) \wedge i \in 1..n{-}1 \wedge \left(\begin{array}{l} m \in \mathbb{N} \wedge \\ m \in ran(f[0..i{-}1]) \wedge \\ (\forall j \cdot j \in 0 \mathinner{..} i{-}1 \Rightarrow f(j) \leq m) \end{array}\right) \}$$

if *F(I)>M* **then**

$$\ell_4 : \{ \left(\begin{array}{l} n_0 \in \mathbb{N} \wedge \\ n_0 \neq 0 \wedge \\ f_0 \in 0 \mathinner{..} n_0{-}1 \rightarrow \mathbb{N} \\ m_0, i_0 \in \mathbb{Z} \\ n = n_0 \wedge f = f0 \end{array}\right) \wedge i \in 1..n{-}1 \wedge \left(\begin{array}{l} m \in \mathbb{N} \wedge \\ m \in ran(f[0..i{-}1]) \wedge \\ (\forall j \cdot j \in 0 \mathinner{..} i{-}1 \Rightarrow f(j) \leq m) \end{array}\right) \wedge$$
$$f(i) > m \}$$

$M := F(I);$

$$\ell_5 : \{ \left(\begin{array}{l} n_0 \in \mathbb{N} \wedge \\ n_0 \neq 0 \wedge \\ f_0 \in 0 \mathinner{..} n_0{-}1 \rightarrow \mathbb{N} \\ m_0, i_0 \in \mathbb{Z} \\ n = n_0 \wedge f = f0 \end{array}\right) \wedge i \in 1..n{-}1 \wedge \left(\begin{array}{l} m \in \mathbb{N} \wedge \\ m \in ran(f[0..i]) \wedge \\ (\forall j \cdot j \in 0 \mathinner{..} i \Rightarrow f(j) \leq m) \end{array}\right) \}$$

;

$$\ell_6 : \{ \left(\begin{array}{l} n_0 \in \mathbb{N} \wedge \\ n_0 \neq 0 \wedge \\ f_0 \in 0 \mathinner{..} n_0{-}1 \rightarrow \mathbb{N} \\ m_0, i_0 \in \mathbb{Z} \\ n = n_0 \wedge f = f0 \end{array}\right) \wedge i \in 1..n{-}1 \wedge \left(\begin{array}{l} m \in \mathbb{N} \wedge \\ m \in ran(f[0..i]) \wedge \\ (\forall j \cdot j \in 0 \mathinner{..} i \Rightarrow f(j) \leq m) \end{array}\right) \}$$

$I{+}{+};$

$$\ell_7 : \{ \left(\begin{array}{l} n \in \mathbb{N} \wedge \\ n \neq 0 \wedge \\ f \in 0 \mathinner{..} n{-}1 \rightarrow \mathbb{N} \end{array}\right) \wedge i \in \mathbb{Z} \wedge \wedge i \in 1..n \wedge \left(\begin{array}{l} m \in \mathbb{N} \wedge \\ m \in ran(f[0..i{-}1]) \wedge \\ (\forall j \cdot j \in 0 \mathinner{..} i{-}1 \Rightarrow f(j) \leq m) \end{array}\right) \}$$

;

$$\ell_8 : \{ \left(\begin{array}{l} n \in \mathbb{N} \wedge \\ n \neq 0 \wedge \\ f \in 0 \mathinner{..} n{-}1 \rightarrow \mathbb{N} \end{array}\right) \wedge i \in \mathbb{Z} \wedge \wedge i = n \wedge \left(\begin{array}{l} m \in \mathbb{N} \wedge \\ m \in ran(f[0..n{-}1]) \wedge \\ (\forall j \cdot j \in 0 \mathinner{..} n{-}1 \Rightarrow f(j) \leq m) \end{array}\right) \}$$

Algorithme 6: Algorithme du maximum d'une liste annoté

EXTENDS *Naturals*, *TLC*
CONSTANTS n
VARIABLES m, i, l

$f \triangleq [j \in 0..n-1 \mapsto j]$

$$\textit{Init} \triangleq \land\ i = 0 \\ \land\ m = 0 \\ \land\ l = \text{"l0"}$$

$$l0l_1 \triangleq \land\ l = \text{"l0"} \\ \land\ m' = f[0] \\ \land\ i' = i \\ \land\ l' = \text{"l1"}$$

$$l1l_2 \triangleq \land\ l = \text{"l1"} \\ \land\ m' = m \\ \land\ i' = 1 \\ \land\ l' = \text{"l2"}$$

$$l2l_3 \triangleq \land\ l = \text{"l2"} \\ \land\ i < n \\ \land\ m' = m \\ \land\ i' = i \\ \land\ l' = \text{"l3"}$$

$$l2l_8 \triangleq \land\ l = \text{"l2"} \\ \land\ (i \geq n) \\ \land\ m' = m \\ \land\ i' = i \\ \land\ l' = \text{"l8"}$$

$$l3l_4 \triangleq \land\ l = \text{"l3"} \\ \land\ f[i] > m \\ \land\ m' = m \\ \land\ i' = i \\ \land\ l' = \text{"l4"}$$

$$l3l_6 \triangleq \land\ l = \text{"l3"} \\ \land\ (f[i] \leq m) \\ \land\ m' = m \\ \land\ i' = i \\ \land\ l' = \text{"l6"}$$

$$l4l_5 \triangleq \land\ l = \text{"l4"} \\ \land\ m' = f[i] \\ \land\ i' = i \\ \land\ l' = \text{"l5"}$$

$$l5l_6 \triangleq \land\ l = \text{"l5"} \\ \land\ m' = m \\ \land\ i' = i \\ \land\ l' = \text{"l6"}$$

$$l6l_7 \triangleq \land\ l = \text{"l6"}$$

$\land\, m' \;=\; m$
$\land\, i' \;=\; i+1$
$\land\, l' = \text{"l7"}$

$l7l_2 \;\triangleq\; \land\, l \;=\; \text{"l7"}$
$\land\, m' \;=\; m$
$\land\, i' \;=\; i$
$\land\, l' = \text{"l2"}$

$\textit{Next} \;\triangleq\; \lor\, l0l1$
$\lor\, l1l2$
$\lor\, l2l3$
$\lor\, l2l8$
$\lor\, l3l4$
$\lor\, l3l6$
$\lor\, l4l5$
$\lor\, l5l6$
$\lor\, l6l7$
$\lor\, l7l2$

$\textit{Safel}_3 \;\triangleq\; l = \text{"l3"} \;\Rightarrow\; \land\,(i \;\in\; 1..n{-}1)$
$\land\,(\exists\, k \;:\; (k \;\in\; 0.i{-}1) \;\land\; f[k] = m)$
$\land\,(\forall\, j \;:\; j \;\in\; 0..i{-}1 \;\Rightarrow\; f[j] \;\leq\; m)$
$\textit{safety} \;\triangleq\; l \;=\; \text{"l8"} \;\Rightarrow\; (\forall\, k \;\in\; 0..n{-}1 :\; m \;\geq\; f[k])$ partial correctnes
$\textit{Safety}_2 \;\triangleq\; l \;\neq\; \text{"l8"}$

```
------------------------------ MODULE appex3_10 ------------------------------
(* computing the maximum value of an array f *)

EXTENDS Naturals, TLC,Integers
CONSTANTS undef,n0,f0,i0,m0,min,max
VARIABLES n,f,m,i,pc
-------------------------------
def0 == [j \in 0..n0-1 |-> n0-j]

-------------------------------
(* precondition *)

ASSUME n0 \in Nat /\ n0 # 0  /\ f0 = def0 /\ i0 \in Int

Init == /\ i = i0
        /\ m = m0
        /\ f=f0
        /\ n=n0
        /\ pc = "l0"
---------------------------------
l0l1 == /\ pc = "l0"
        /\ m' = f[0]
        /\ pc'= "l1"
        /\ UNCHANGED <<n,f,i>>
```

```
l1l2 == /\ pc = "l1"
        /\ i' = 1
        /\ pc'= "l2"
        /\ UNCHANGED <<n,f,m>>


l2l3 == /\ pc = "l2"
        /\ i < n
        /\ pc'= "l3"
        /\ UNCHANGED <<n,f,m,i>>


l2l8 == /\ pc = "l2"
        /\ (i \geq n)
        /\ m' = m
        /\ i' = i
        /\ pc'= "l8"
        /\ UNCHANGED <<n,f>>

l3l4 == /\ pc = "l3"
        /\ f[i] > m
        /\ m' = m
        /\ i' = i
        /\ pc'= "l4"
        /\ UNCHANGED <<n,f>>


l3l6 == /\ pc = "l3"
        /\ (f[i] \leq m)
        /\ m' = m
        /\ i' = i
        /\ pc'= "l6"
        /\ UNCHANGED <<n,f>>

l4l5 == /\ pc = "l4"
        /\ m' = f[i]
        /\ i' = i
        /\ pc'= "l5"
        /\ UNCHANGED <<n,f>>


l5l6 == /\ pc = "l5"
        /\ m' = m
        /\ i' = i
        /\ pc'= "l6"
        /\ UNCHANGED <<n,f>>

l6l7 == /\ pc = "l6"
        /\ m' = m
        /\ i' = i + 1
        /\ pc'= "l7"
       /\ UNCHANGED <<n,f>>


l7l3 == /\ pc = "l7"
```

```
        /\ i < n
        /\ m' = m
        /\ i' = i
        /\ pc= "l3"
        /\ UNCHANGED <<n,f>>

 l7l8 ==
        /\ pc = "l7"
        /\ i \geq n
        /\ m' = m
        /\ i' = i
        /\ pc'= "l8"
        /\ UNCHANGED <<n,f>>



Next == \/ l0l1
        \/ l1l2
        \/ l2l3
        \/ l2l8
        \/ l3l4
        \/ l3l6
        \/ l4l5
        \/ l5l6
        \/ l6l7
        \/ l7l3
        \/ l7l8
        \/ UNCHANGED <<n,m,i,f,pc>>

pre0 == n0 \in Nat /\ n0 # 0  /\ f0 = def0 /\ i0 \in Int
pre1 == f=f0 /\ n=n0 /\ pre0

   zinf == min..max
  ninf == 0..max


  Dl0l1 ==  0\leq 0 /\ 0 \leq n0-1
  Dl1l2 == 1 \in zinf
 inv ==
   /\ pc \in {"l0","l1","l2","l3","l4","l5","l6","l7","l8"}
   /\ n \in Int /\ f = def0 /\ i \in Int /\ m \in Int
   /\ pc="l0" =>   f=f0 /\ n=n0 /\ m=m0 /\ i = i0/\ pre0 /\ Dl0l1
   /\ pc="l1" =>  f=f0 /\ n=n0 /\ m=f[0] /\ i = i0 /\ pre0 /\ Dl1l2
   /\ pc ="l2" =>f=f0 /\ n=n0 /\ m=f[0] /\ i = 1 /\ pre0
   /\ pc="l3" =>  (\E j \in 0..i-1 : f[j]=m) /\ (\A k \in 0..i-1: f[k] \leq m) /\ (
   /\ pc="l4" =>  (\E j \in 0..i-1 : f[j]=m) /\ (\A k \in 0..i-1: f[k] \leq m) /\ (
   /\ pc="l5"  =>  (\E j \in 0..i-1 : f[j]=m) /\ (\A k \in 0..i: f[k] \leq m) /\ (i
   /\ pc="l6" =>   (\E j \in 0..i : f[j]=m) /\ (\A k \in 0..i: f[k] \leq m)  /\pre1
   /\ pc="l7" =>   (\E j \in 0..i-1 : f[j]=m) /\ (\A k \in 0..i-1: f[k] \leq m) /\
   /\ pc="l8" =>   (\E j \in 0..i-1 : f[j]=m) /\ (\A k \in 0..i: f[k] \leq m)  /\pr



runtimeerrors ==  m \in zinf /\ i \in zinf /\ n \in zinf
```

==

Fin 11

Exercice 12 *On considère l'algorithme* $squareroot$ *12 calculant la racine carrée entière d'un nombre naturel* $x \in \mathbb{N}$*.*

Question 12.1 *Complétez cet algorithme en proposant trois assertions :*
— $P_{\ell_2}(z, y1, y2, y3)$
— $P_{\ell_4}(z, y1, y2, y3)$
— $P_{\ell_5}(z, y1, y2, y3)$

Question 12.2 *Pour chaque paire* (ℓ, ℓ') *d'étiquettes correspondant à un pas élémentaire ; on vérifie la propriété suivante :*
$\forall x, y, q, r, x', y', q', r'.P_\ell(y_1, y_2, y_3, z) \wedge cond_{\ell,\ell'}(y_1, y_2, y_3, z) \wedge (y_1', y_2', y_3', z') = f_{\ell,\ell'}(y_1, y_2, y_3, z) \Rightarrow P_{\ell'}(y_1', y_2', y_3', z')$
Enoncez et vérifiez cette propriété pour les paires d'étiquettes suivantes : (ℓ_1, ℓ_2) *;*(ℓ_1, ℓ_4) *;* (ℓ_2, ℓ_3) *;* (ℓ_3, ℓ_2) *;* (ℓ_3, ℓ_4) *;* (ℓ_4, ℓ_5) *;*

Question 12.3 *On suppose que toutes les conditions de vérifications associées aux paires d'étiquettes successives de l'algorithme sont vérifiées. Quelles sont les deux conditions à montrer pour déduire que l'algorithme est partiellement correct par rapport aux pré et post conditions ? Vous donnerez explicitement les conditions et vous expliquerez pourquoi elles sont correctes.*

Question 12.4 *Expliquer que cet algorithme est sans erreurs à l'exécution, si les données initiales sont dans un domaine à définir inclus dans le domaine des entiers informatiques c'est-à-dire les entiers codables sur n bits. L'ensemble des entiers informatiques sur n bits est l'ensemble noté* $\mathbb{Z}_n$ *et défini par* $\{i | i \in \mathbb{Z} \wedge -2^{n-1} \leq i \wedge i \leq 2^{n-1} - 1\}$.

Variables : X,Y1,Y2,Y3,Z
Requires : $x_0 \in \mathbb{N}$
Ensures : $z_f^2 \leq x_0 \wedge x_0 < (z_f + 1)^2$

$\ell_0 : \{x_0 \in \mathbb{N} \wedge z_0 \in \mathbb{Z} \wedge y1_0 \in \mathbb{Z} \wedge y2_0 \in \mathbb{Z} \wedge y3_0 \in \mathbb{Z} \wedge (x, y1, y2, y3, z) = (x_0, y1_0, y2_0, y3_0, z_0)\}$
$(y_1, y_2, y_3) := (0, 1, 1);$
$\ell_1 : \{x_0 \in \mathbb{N} \wedge z_0 \in \mathbb{Z} \wedge y1_0 \in \mathbb{Z} \wedge y2_0 \in \mathbb{Z} \wedge y3_0 \in \mathbb{Z} \wedge y2 = (y1+1) \cdot (y1+1) \wedge y3 = 2 \cdot y1 + 1 \wedge y1 \cdot y1 \leq x \wedge (x, z) = (x_0, , z_0)\}$
while $y_2 \leq x$ **do**
  $\ell_2 : \{\ldots\}$
  $(y_1, y_2, y_3) := (y_1 + 1, y_2 + y_3 + 2, y_3 + 2);$
  $\ell_3 : \{\ldots\}$
;
$\ell_4 : \{\ldots\}$
$z := y_1;$
$\ell_5 : \{\ldots\}$

Algorithme 7: $squareroot$ partiellement annotée

L'algorithme annoté est décrit par l'algorithme 12

MODULE *algo_squareroot*

precondition : $x \in \mathbb{N}$
postcondition : $z^2 \leq x \wedge x < (z+1)^2$
local variables : $y_1, y_2, y_3 \in \mathbb{N}$

$pre : \{x \in \mathbb{N}\}$
$post : \{z \cdot z \leq x \wedge x < (z+1) \cdot (z+1)\}$
$\ell_0 : \{x \in \mathbb{N} \wedge z \in \mathbb{Z} \wedge y1 \in \mathbb{Z} \wedge y2 \in \mathbb{Z} \wedge y3 \in \mathbb{Z}\}$
$(y_1, y_2, y_3) := (0, 1, 1);$
$\ell_1 : \{y2 = (y1+1) \cdot (y1+1) \wedge y3 = 2 \cdot y1+1 \wedge y1 \cdot y1 \leq x\}$
while $y_2 \leq x$ **do**
  $\ell_2 : \{y2 = (y1+1) \cdot (y1+1) \wedge y3 = 2 \cdot y1+1 \wedge y2 \leq x\}$
  $(y_1, y_2, y_3) := (y_1+1, y_2+y_3+2, y_3+2);$
  $\ell_3 : \{y2 = (y1+1) \cdot (y1+1) \wedge y3 = 2 \cdot y1+1 \wedge y1 \cdot y1 \leq x\}$
;
$\ell_4 : \{y2 = (y1+1) \cdot (y1+1) \wedge y3 = 2 \cdot y1+1 \wedge y1 \cdot y1 \leq x \wedge x < y2\}$
$z := y_1;$
$\ell_5 : \{y2 = (y1+1) \cdot (y1+1) \wedge y3 = 2 \cdot y1+1 \wedge y1 \cdot y1 \leq x \wedge x < y2 \wedge z = y1 \wedge z \cdot z \leq x \wedge x < (z+1) \cdot (z+1)\}$

Algorithme 8: *squareroot* annotée

EXTENDS *Integers*, *TLC*
CONSTANTS x *x is the input*
VARIABLES $pc, y_1, y2, y_3, z$

$vars \triangleq \langle pc, y_1, y2, y_3, z \rangle$
$al0l_1 \triangleq pc = \text{"l0"} \wedge y_1' = 0 \wedge y_2' = 1 \wedge y_3' = 1 \wedge pc' = \text{"l1"} \wedge z' = z$

$al1l_2 \triangleq pc = \text{"l1"} \wedge y_2 \leq x \wedge pc' = \text{"l2"} \wedge \text{UNCHANGED } \langle y_1, y2, y_3, z \rangle$

$al1l_4 \triangleq pc = \text{"l1"} \wedge y_2 > x \wedge pc' = \text{"l4"} \wedge \text{UNCHANGED } \langle y_1, y2, y_3, z \rangle$

$al2l_3 \triangleq pc = \text{"l2"} \wedge y_1' = y_1+1 \wedge y_2' = y_2+y3+2 \wedge y_3' = y_3+2 \wedge pc' = \text{"l3"} \wedge z' = z$

$al3l_2 \triangleq pc = \text{"l3"} \wedge y_2 \leq x \wedge pc' = \text{"l2"} \wedge \text{UNCHANGED } \langle y_1, y2, y_3, z \rangle$

$al3l_4 \triangleq pc = \text{"l3"} \wedge y_2 > x \wedge pc' = \text{"l4"} \wedge \text{UNCHANGED } \langle y_1, y2, y_3, z \rangle$

$al4l_5 \triangleq pc = \text{"l4"} \wedge z' = y_1 \wedge pc' = \text{"l5"} \wedge \text{UNCHANGED } \langle y_1, y2, y_3 \rangle$

$Init \triangleq y_1 = 0 \wedge y_2 = 0 \wedge y_3 = 0 \wedge z = 0 \wedge pc = \text{"l0"}$

$Next \triangleq al0l_1 \vee al1l_2 \vee al1l_4 \vee al2l_3 \vee al3l_2 \vee al3l_4 \vee al4l_5$

$MAX \triangleq 32768$ *16 bits*
$D \triangleq 0..32768$
x \leq 32760

$Safety\_absence \triangleq (y_1 \in D) \wedge (y_2 \in D) \wedge (y_3 \in D) \wedge (z \in D)$

$i \triangleq$
  $\wedge\, pc = \text{"l0"} \Rightarrow y_1 \in D \wedge y_2 \in D \wedge y_3 \in D \wedge z \in D$

$\wedge\, pc =$ *"l1"* $\Rightarrow\ y_2 \ = \ (y_1{+}1){\cdot}(y_1{+}1) \ \wedge\ y_3 = 2{\cdot}y_1{+}1 \ \wedge\ y_1{\cdot}y1 \leq\ x \ \wedge$ *Safety_absence*
$\wedge\, pc =$ *"l2"* $\Rightarrow\ y_2 \ = \ (y_1{+}1){\cdot}(y_1{+}1) \ \wedge y_3 = 2{\cdot}y_1{+}1 \ \wedge y_1{\cdot}y1 \leq\ x \ \wedge y_2 \ \leq\ x \wedge$ *Safety_absence*

$\wedge\, pc =$ *"l3"* $\Rightarrow\ y_2 \ = \ (y_1{+}1){\cdot}(y_1{+}1) \ \wedge\ y_3 = 2{\cdot}y_1{+}1 \ \wedge\ y_1{\cdot}y1 \leq\ x \ \wedge$ *Safety_absence*
$\wedge\, pc =$ *"l4"* $\Rightarrow\ y_2 \ = \ (y_1{+}1){\cdot}(y_1{+}1) \ \wedge y_3 = 2{\cdot}y_1{+}1 \ \wedge y_1{\cdot}y1 \leq\ x \ \wedge x \ < \ y_2 \wedge$ *Safety_absence*

$\wedge\, pc =$ *"l5"* $\Rightarrow\ y_2 \ = \ (y_1{+}1){\cdot}(y_1{+}1) \ \wedge y_3 = 2{\cdot}y_1{+}1 \ \wedge z{\cdot}z \leq\ x \wedge x \ < \ (z{+}1){\cdot}(z{+}1) \wedge$ *Safety_absence*

Safety_partialcorrectness $\triangleq$ *pc = "l5"* $\Rightarrow$ $\wedge\, y_2 \ = \ (y_1{+}1){\cdot}(y_1{+}1)$
$\wedge\, y_3 = 2{\cdot}y_1{+}1$
$\wedge\, z{\cdot}z \leq\ x \ \wedge\ x \ < \ (z{+}1){\cdot}(z{+}1)$

Exercice 13

Montrer, pour chaque question, que chaque annotation est correcte ou incorrecte selon les conditions de vérifications énoncées comme suit
$\forall v, v'.P_\ell(v) \ \wedge\ cond_{\ell,\ell'}(v) \ \wedge\ v' \ = \ f_{\ell,\ell'}(v) \ \Rightarrow\ P_{\ell'}(v')$. *Vous devez répondre en énonçant et en démontrant les Conditions de vérification.*

Question 13.1

$\ell_1 : x = 12 \ \wedge\ y = 2 \ \wedge z = 3{\cdot}x$
$x := z{+}y$
$\ell_2 : x = 38 \ \wedge\ y = 2$

Question 13.2

$\ell_1 : x = 3 \ \wedge\ y = 9$
$x := 3{\cdot}y$
$\ell_2 : x = 27 \ \wedge\ y = 9$

Question 13.3 *Soit* p *un nombre différent d'une puissance de 3 c'est-à-dire différent de 3, 6, 9, 12, . . .*

$\ell_1 : x = 3{+}z \ \wedge\ y = 1 \ \wedge\ z = 3 \ \wedge x = y$
$x := p{\cdot}y$
$\ell_2 : x = z \ \wedge\ y = z \ \wedge z = 4{\cdot}p$

Question 13.4 *Soit* r *un nombre cubique c'est-à-dire de la forme* $p = q^3$.

$\ell_1 : x = r \ \wedge\ u = x^r \ \wedge\ z = 6 \ \wedge x = u$
$y := r{\cdot}r{\cdot}r$
$\ell_2 : x = z \ \wedge\ y = z \ \wedge z = 4{\cdot}p$

Exercice 14

Soit l'algorithme annoté suivant se trouvant à la page suivante et les pré et postconditions définies pour cet algorithme comme suit : On suppose que $x1$ *et* $x2$ *sont des constantes.*

Question 14.1 *Compléter les annotations associées à chaque étiquette* $\ell \in \{\ell_3, \ell_6, \ell_8, \ell_9\}$. *Vous devez écrire les annotations complètes de chaque point de contrôle demandé.*

Question 14.2 *Pour chaque paire* (ℓ, ℓ') *d'étiquettes correspondant à un pas élémentaire ; on vérifie la propriété suivante :*
$\forall x, y, q, r, x', y', q', r'.P_\ell(y_1, y_2, y_3, z) \ \wedge\ cond_{\ell,\ell'}(y_1, y_2, y_3, z) \ \wedge\ (y'_1, y'_2, y'_3, z') = f_{\ell,\ell'}(y_1, y_2, y_3, z) \Rightarrow P_{\ell'}(y'_1, y'_2, y'_3, z')$

Variables : X1,X2,Y1,Y2,Y3,Z
Requires : $x1_0 \in \mathbb{N} \wedge x2_0 \in \mathbb{N} \wedge x1_0 \neq 0$
Ensures : $z_f = x1_0{}^{x2_0}$

ℓ_0 : $\{x1_0 \in \mathbb{N} \wedge x2_0 \in \mathbb{N} \wedge x1_0 \neq 0 \wedge y1_0, y2_0, y3_0, z_0 \in \mathbb{Z} \wedge (x1, x2, y1, y2, y3, z) = (x1_0, x2_0, y1_0, y2_0, y3_0, z0)\}$
$(y_1, y_2, y_3) := (x_1, x_2, 1)$;
ℓ_1 : $\{x1_0 \in \mathbb{N} \wedge x2_0 \in \mathbb{N} \wedge x1_0 \neq 0 \wedge y1_0, y2_0, y3_0, z_0 \in \mathbb{Z} \wedge (x1, x2, z) = (x1_0, x2_0, z0) \wedge y_3 \cdot y_1{}^{y_2} = x_1{}^{x_2}\}$
while $y_2 \neq 0$ **do**
  ℓ_2 : $\{x1_0 \in \mathbb{N} \wedge x2_0 \in \mathbb{N} \wedge x1_0 \neq 0 \wedge y1_0, y2_0, y3_0, z_0 \in \mathbb{Z} \wedge (x1, x2, z) = (x1_0, x2_0, z0) \wedge y_3 \cdot y_1{}^{y_2} = x_1{}^{x_2} \wedge 0 < y2 \leq x2\}$
  if $impair(y_2)$ **then**
    ℓ_3 : $\{x1_0 \in \mathbb{N} \wedge x2_0 \in \mathbb{N} \wedge x1_0 \neq 0 \wedge y1_0, y2_0, y3_0, z_0 \in \mathbb{Z} \wedge (x1, x2, z) = (x1_0, x2_0, z0) \wedge y_3 \cdot y_1{}^{y_2} = x_1{}^{x_2} \wedge 0 < y2 \leq x2 \wedge impair(y2)\}$
    $y_2 := y_2 - 1$;
    ℓ_4 : $\{x1_0 \in \mathbb{N} \wedge x2_0 \in \mathbb{N} \wedge x1_0 \neq 0 \wedge y1_0, y2_0, y3_0, z_0 \in \mathbb{Z} \wedge (x1, x2, z) = (x1_0, x2_0, z0) \wedge y_3 \cdot y1 \cdot y_1{}^{y_2} = x_1{}^{x_2} \wedge 0 \leq y2 \leq x2 \wedge pair(y2)\}$
    $y_3 := y_3 \cdot y_1$;
    ℓ_5 : $\{x1_0 \in \mathbb{N} \wedge x2_0 \in \mathbb{N} \wedge x1_0 \neq 0 \wedge y1_0, y2_0, y3_0, z_0 \in \mathbb{Z} \wedge (x1, x2, z) = (x1_0, x2_0, z0) \wedge y_3 \cdot y_1{}^{y_2} = x_1{}^{x_2} \wedge 0 \leq y2 \leq x2 \wedge pair(y2))\}$
  ;
  ℓ_6 : $\{x1_0 \in \mathbb{N} \wedge x2_0 \in \mathbb{N} \wedge x1_0 \neq 0 \wedge y1_0, y2_0, y3_0, z_0 \in \mathbb{Z} \wedge (x1, x2, z) = (x1_0, x2_0, z0) \wedge y_3 \cdot y_1{}^{y_2} = x_1{}^{x_2} \wedge 0 \leq y2 \leq x2 \wedge pair(y2)\}$

  $y_1 := y_1 \cdot y_1$;
  ℓ_7 : $\{x1_0 \in \mathbb{N} \wedge x2_0 \in \mathbb{N} \wedge x1_0 \neq 0 \wedge y1_0, y2_0, y3_0, z_0 \in \mathbb{Z} \wedge (x1, x2, z) = (x1_0, x2_0, z0) \wedge y_3 \cdot y_1{}^{y_2 \ div \ 2} = x_1{}^{x_2} \wedge 0 \leq y2 \leq x2 \wedge pair(y2)\}$

  $y_2 := y_2 \ div \ 2$;
  ℓ_8 : $\{x1_0 \in \mathbb{N} \wedge x2_0 \in \mathbb{N} \wedge x1_0 \neq 0 \wedge y1_0, y2_0, y3_0, z_0 \in \mathbb{Z} \wedge (x1, x2, z) = (x1_0, x2_0, z0) \wedge y_3 \cdot y_1{}^{y_2} = x_1{}^{x_2} \wedge 0 \leq y2 \leq x2\}$
;
ℓ_9 : $\{x1_0 \in \mathbb{N} \wedge x2_0 \in \mathbb{N} \wedge x1_0 \neq 0 \wedge y1_0, y2_0, y3_0, z_0 \in \mathbb{Z} \wedge (x1, x2, z) = (x1_0, x2_0, z0) \wedge y_3 \cdot y_1{}^{y_2} = x_1{}^{x_2} \wedge y2 = 0\}$

$z := y_3$;
ℓ_{10} : $\{x1_0 \in \mathbb{N} \wedge x2_0 \in \mathbb{N} \wedge x1_0 \neq 0 \wedge y1_0, y2_0, y3_0, z_0 \in \mathbb{Z} \wedge (x1, x2) = (x1_0, x2_0) \wedge y_3 \cdot y_1{}^{y_2} = x_1{}^{x_2} \wedge y2 = 0 \wedge z = x1^{x2}\}$

Algorithme 9: Algorithme de l'exponentitaion indienne annoté

Enoncer et vérifier cette propriété pour les paires d'étiquettes suivantes : (ℓ_0, ℓ_1) *;* (ℓ_1, ℓ_2) *;* (ℓ_3, ℓ_4) *;* (ℓ_6, ℓ_7) *;* (ℓ_7, ℓ_8) *;* (ℓ_1, ℓ_9) *;* (ℓ_9, ℓ_{10}).
Il est clair que cette vérification confirmera les complétions réalisées dans la question précédente.

Question 14.3 *On suppose que toutes les conditions de vérifications associées aux paires d'étiquettes successives de l'algorithme sont vérifiées. Quelles sont les deux conditions à montrer pour déduire que l'algorithme est partiellement correct par rapport aux pré et post conditions ? Vous donnerez explicitement les conditions et vous expliquerez pourquoi elles sont correctes.*

Question 14.4 *Selon la définition mathématique de la puissance* $x_1^{x_2}$ *est définie pour une valeur* x_1 *non nulle et c'est pour cela que la précondition indique que* x_1 *est différent de* 0. *Cependant, si on utilise une valeur de* x_1 *nulle, l'algorithme fonctionne et renvoie une valeur. Un jour, un mathématicien a appliqué cet algorithme sans veiller à ce que la valeur de* x_1 *soit nulle ou non nulle et il 'est emporté !... Il vous accuse de ne pas lui avoir fourni le bon algorithme répondant à son cahier des charges et il vous demande des dommages et intérêts. Expliquer de manière courte que le texte de l'algorithme et sa preuve de correction suffisent pour vous sauver, en expliquant clairement le rôle de la précondition et de la postcndition.*