Université de Sherbrooke, Département d'informatique

IGL501 : Méthodes formelles en génie logiciel, Examen périodique

Professeur : Marc Frappier, Vendredi 12 octobre 2012, 8h30 à 11h20, local D4-2021

Documentation permise. La correction est, entre autres, basée sur le fait que chacune de vos réponses soit *claire*, c'est-à-dire lisible et compréhensible pour le lecteur; *précise*, c'est-à-dire exacte et sans erreur; *concise*, c'est-à-dire qu'il n'y ait pas d'élément superflu; *complète*, c'est-à-dire que tous les éléments requis sont présents.

Pondération :

1	10 pts	2	15 pts	3	25 pts	4	25 pts	5	25 pts	Total	100 pts

1. (10 pts) Soit **SETS** $AA=\{a1,a2\}$. Indiquez si les expressions suivantes sont vraies ou fausses, Justifiez votre réponse.

1.	$a1 \mapsto a2 \in AA \times AA$; **True**	6.	$AA ⤖ AA \subseteq AA \leftrightarrow AA$; **True**
2.	$\{a1 \mapsto a2\} \in AA \rightarrowtail AA$; **False**	7.	$\{a1 \mapsto \{a2 \mapsto a2\}\} \in AA \leftrightarrow (AA \leftrightarrow AA)$; **True**
3.	$\{a1 \mapsto a2, a2 \mapsto a2\} \in AA \rightarrow AA$; **True**	8.	$\forall (ff).(ff \in AA ⤖ AA \Rightarrow ff^{-1} \in AA ⤖ AA)$; **True**
4.	$\{a1 \mapsto a2, a2 \mapsto a2\} \in AA ⤀ AA$; **False**	9.	$\forall (ff).(ff \in AA \twoheadrightarrow AA \Rightarrow \mathbf{dom}(ff) = \mathbf{ran}(ff))$; **True**
5.	$\{\{a1 \mapsto a2, a2 \mapsto a2\}\} \subseteq AA \leftrightarrow AA$; **True**	10.	$\forall (ff).(ff \in AA ⇸ AA \Rightarrow AA ⩤ ff = \varnothing)$ **True**

2. (15 pts) Soit la machine B suivante :

```
MACHINE q2
VARIABLES v1
INVARIANT v1 ∈ 0..4
INITIALISATION v1 := 0
OPERATIONS
  res ← op1 =
  PRE v1 ≤ 2 THEN
    ANY x1 WHERE x1 ∈ NAT ∧ x1 ∈ 0..2 THEN res := x1 || v1 := v1+x1 END
  END;

  res ← op2 =
  PRE v1 ∈ NAT THEN
    ANY x1 WHERE x1 ∈ NAT ∧ x1 ∈ 0..2 THEN res := x1 || v1 := v1+x1 END
  END;

  op3 =
  PRE v1 ∈ 0..2 THEN
    SELECT v1 = 0 THEN v1:=v1+1
    WHEN v1 = 1 THEN v1:=0
    WHEN v1 = 2 THEN v1 := 1
    ELSE v1 := 3 END
  END
END
```

a) Indiquez si l'invariant est préservé. Justifiez votre réponse.
Solution : Non, car l'opération op2 viole l'invariant si l'état courant est v1=4 : op2 peut choisir x1=1 ou x1 = x2, ce qui donne v1=5 ou v1=6, respectivement, ce qui ne satisfait pas $v1 \in 0..4$.

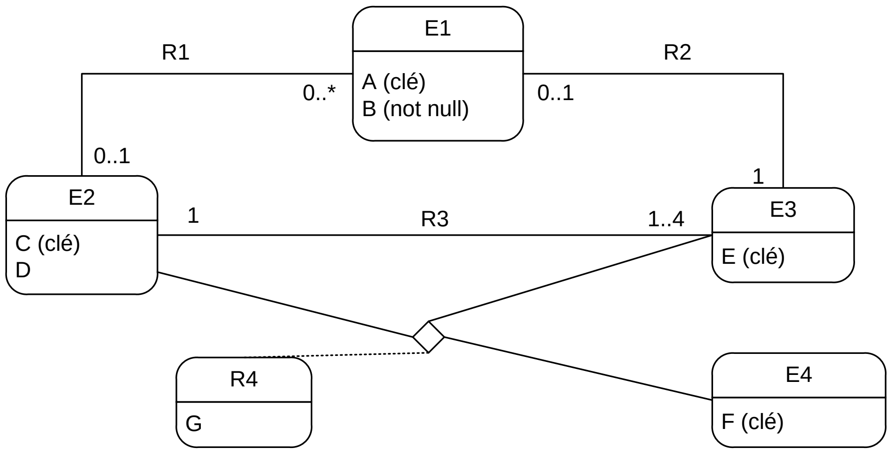
b) Quel est l'état de la machine après avoir exécuté les 3 opérations suivantes :

1 ← op1 ; 2 ← op1 ; 2 ← op2.

La valeur du paramètre de sortie est indiquée avec la flèche. Par exemple, 1 ← op1 signifie que l'opération a retourné la valeur 1 en sortie après exécution.
Solution : v1=5

3. (25 pts) Traduisez le diagramme suivant en une machine B. Ne donnez que les variables et les invariants. Notez que R4 est une classe associative. Représentez bien toutes les contraintes d'intégrité du diagramme. Prenez aussi en compte les contraintes suivantes :

a) si une paire $(e2,e3) \in R3$, alors il doit aussi exister un triplet $(e2,e3,e4) \in R4$, et vice-versa.
b) chaque instance de E4 apparaît dans au moins un triplet de R4.



Solution :

VARIABLES e1,e2,e3,e4,r1,r2,r3,r4,a,b,c,d,e,f,g
INVARIANT
$e1 \subseteq E1 \wedge$
$e2 \subseteq E2 \wedge$
$e3 \subseteq E3 \wedge$
$e4 \subseteq E4 \wedge$
$r1 \in E1 \text{ +-> } E2 \wedge$
$r2 \in E1 \text{ >-> } E3 \wedge$
$r3 \in E3 \text{ --> } E2 \wedge$
$\forall(xe2).(xe2 \in ran(r3) \Rightarrow card(r3 \rhd \{xe2\}) \in 1..4) \wedge$
$r4 \subseteq (E2*E3)*E4 \wedge$
$a \in e1 \text{ >-> } TYPE\_A \wedge$
$b \in e1 \text{ --> } TYPE\_B \wedge$
$c \in e2 \text{ >-> } TYPE\_C \wedge$
$d \in e2 \text{ +-> } TYPE\_D \wedge$
$e \in e3 \text{ >-> } TYPE\_E \wedge$
$f \in e3 \text{ >-> } TYPE\_F \wedge$
$g \in r4 \text{ +-> } TYPE\_G \wedge$
(* pour question a) *)
$\forall(xe2,xe3).(xe2 \mapsto xe3 \in r3 \Rightarrow \exists(xe4).((xe2 \mapsto xe3) \mapsto xe4 \in r4)) \wedge$
$\forall(xe2,xe3,xe4).((xe2 \mapsto xe3) \mapsto xe4 \in r4 \Rightarrow xe2 \mapsto xe3 \in r3) \wedge$
Ou bien simplement $dom(r4) = r3$
(* question b) *)
$e4 \subseteq ran(r4)$

4. (25 pts) Définissez une machine B spécifiant une liste. Les n éléments de la liste sont numérotés de 1 à n. La capacité maximale de la liste est *maxliste*. Voici les variables et invariants que vous devez utiliser.

ABSTRACT_VARIABLES ll
INVARIANT $ll \in 1..maxliste \mapsto ELEMENT \wedge$
$\mathbf{dom}(ll) = 1..\mathbf{card}(ll)$

Voici les opérations à spécifier

a) **insert**$(p1,x1)$: insère l'élément $x1$ à la position $p1$; les éléments de la position $p1$ à n sont décalés d'une position vers le haut (position supérieure); pour insérer à la fin, on spécifie $p1=n+1$.
b) **delete**$(p1)$: supprime l'élément à la position $p1$; les éléments de la position $p1+1$ à n sont décalés d'une position vers le bas.
c) $val \leftarrow$ **get**$(p1)$: retourne l'élément à la position $p1$; si $p1$ n'existe pas dans la liste, l'opération n'est pas définie.

Indice : pour décaler le domaine d'une fonction $f \in NAT \mapsto NAT$ d'une position vers le haut, on fait $succ^{\sim}$ **;** f, où $succ$ est la fonction successeur : succ(x)=x+1. Pour décaler vers une position vers le bas, on fait $succ$ **;** f

Solution :

MACHINE *q4*

SETS *ELEMENT*

ABSTRACT_CONSTANTS *maxliste*

PROPERTIES $maxliste \in \mathbf{NAT}$

VARIABLES *ll*

INVARIANT
$ll \in 1..maxliste \mapsto ELEMENT \wedge$
$\mathbf{dom}(ll) = 1..\mathbf{card}(ll)$

INITIALISATION
$ll := \varnothing$

OPERATIONS

insert$(p1, x1) =$
PRE
$x1 \in ELEMENT \wedge$
$p1 \in \mathbf{NAT} \wedge$
$p1 \in 1..\mathbf{card}(\mathbf{dom}(ll))+1 \wedge$
$\mathbf{card}(\mathbf{dom}(ll)) < maxliste$
THEN
$ll := (1..p1\text{-}1 \lhd ll) \cup \{p1 \mapsto x1\} \cup (\mathbf{succ}^{-1};(p1..maxliste) \lhd ll)$
END;

delete$(p1) =$
PRE
$p1 \in \mathbf{NAT} \wedge$
$p1 \in 1..\mathbf{card}(\mathbf{dom}(ll))$
THEN
$ll := (1..p1\text{-}1 \lhd ll) \cup (\mathbf{succ};(p1+1..maxliste) \lhd ll)$
END;

$val \leftarrow \mathbf{get}(p1) =$
PRE
$p1 \in \mathbf{NAT} \wedge$
$p1 \in 1..\mathbf{card}(\mathbf{dom}(ll))$
THEN
$val := ll(p1)$
END

END

5. (25 pts) Définissez un raffinement de la machine de la question 4. Voici les variables et une partie de l'invariant du raffinement à utiliser.

ABSTRACT_VARIABLES

tt, $count$

INVARIANT

$tt \in 1..maxliste \rightarrow ELEMENT \wedge$

$count \in 0..maxliste \wedge$ … *à compléter* …

Solution

REFINEMENT
q4_r
REFINES
q4

ABSTRACT_VARIABLES
tt, *count*

INVARIANT
$tt \in 1..maxliste \rightarrow ELEMENT \land$
$count \in 0..maxliste \land$
$1..count \lhd tt = ll$

INITIALISATION
$tt :\in 1..maxliste \rightarrow ELEMENT \parallel$
$count := 0$

OPERATIONS
insert $(p1 , x1) =$
PRE
$x1 \in ELEMENT \land p1 \in \mathbf{NAT} \land p1 \in 1 .. count + 1 \land count < maxliste$
THEN
$tt := tt \mathbin{\lhd\!\!+} (\{ p1 \mapsto x1 \} \cup (\mathbf{succ}^{-1} ; (p1 .. count) \lhd tt)) \parallel$
$count := count+1$
END
;

delete $(p1) =$
PRE
$p1 \in \mathbf{NAT} \land p1 \in 1 .. count$
THEN
$tt := tt \mathbin{\lhd\!\!+} (\mathbf{succ} ; (p1 + 1 .. count) \lhd tt) \parallel$
$count := count-1$
END
;

$val \leftarrow$ **get** $(p1) =$
PRE
$p1 \in \mathbf{NAT} \land p1 \in 1 .. count$
THEN
$val := tt (p1)$
END

END