Preuves de programme

Fabrice Rossi

28 janvier 2012

1 Affectation

Le traitement rigoureux de l'affectation n'est pas aussi simple qu'on le pense. Étant donné un prédicat (ou une expression de calcul) P contenant une variable libre x et ne contenant pas de variable y, on note $P[x \leftarrow y]$ le prédicat dans lequel on a remplacé toutes les occurrences de x par la nouvelle variable libre y. Le triplet suivant est alors totalement correct

$$\langle P \rangle \ \texttt{x} = \texttt{E} \ \langle \exists y, P[x \leftarrow y] \wedge x = E[x \leftarrow y] \rangle \, .$$

La valeur de y qui apparaît dans la postcondition est en fait celle que x avait avant l'exécution de l'affectation. Cette valeur est généralement inconnue, sauf si P la précise explicitement.

Exercice 1.1
Utilisez la règle ci-dessus pour montrer que le triplet suivant est vrai

$$\langle x - y \geq 0 \rangle \ \texttt{y} = \texttt{y} + \texttt{x} \ \langle \exists z, (x - z \geq 0) \wedge (y = z + x) \rangle \, .$$

Exercice 1.2
Donner une postcondition Q rendant le triplet suivant vrai (on montrera la correction totale du triplet proposé) :

$$\langle x - y \geq 0 \rangle \ \texttt{x} = \texttt{y} - \texttt{x}; \ \texttt{y} = \texttt{y} + \texttt{x} \ \langle Q \rangle \, .$$

2 Sélection

Exercice 2.1
On considère le programme suivant :

prog

```
if (x>0)
  y = 1
else
  y = -1
end if
```

1. Donner une postcondition Q telle que $\emptyset$ `prog` Q soit vrai au sens de la correction totale.
2. Donner des exemples de préconditions qui conduisent à une postcondition différente de celle proposée au dessus (pour des triplets totalement corrects).

Exercice 2.2
On considère le programme suivant :

prog
```
if (x>0)
  y = 1
  x = x + 2
else
  y = -1
  x = x - 2
end if
```

1. On considère la précondition $P = (x = a) \wedge (a > 0)$. Donner une postcondition telle que `prog` Q soit vrai au sens de la correction totale.
2. Même question pour $P = (x = a)$.
3. Même question pour $P = \emptyset$.

3 Correction partielle de boucles

On rappelle que si le triplet $\{I \wedge b\}$ `prog` $\{I\}$ est vrai, alors le triplet suivant est vrai

$$\{I\} \texttt{ while(}b\texttt{) prog } \{I \wedge \neg b\}.$$

Dans cette situation, I est un **invariant de boucle**.

Exercice 3.1
On suppose que `tab.length` donne la longueur d'un tableau et que `tab[i]` permet d'accéder à sa case numéro `i` (la numérotation commence à 0). On considère le programme suivant :

prog
```
i = 0
x = tab[0]
j = 1
while(j < tab.length)
  if(tab[j] > x)
    x = tab[j]
    i = j
  end if
  j++
end while
```

On considère la précondition $P = tab.length > 0$ et on cherche à montrer que x contient le plus grand élément de tab après l'exécution du programme.

1. Montrer que I défini par
$$I = \left(x = \max_{0 \leq k \leq j-1} tab[k]\right) \wedge (j \leq tab.length)$$
est un invariant de boucle.
2. En déduire la propriété recherchée (en explicitant la postcondition Q).
3. Montrer que les accès au tableau sont toujours corrects (c'est-à-dire qu'on ne tente jamais d'accéder à une case qui n'existe pas).
4. Montrer que i contient après l'exécution du programme le plus petit indice k tel que $tab[k]$ contienne la plus grande valeur du tableau.

5. Proposer une modification du programme qui donne à i la position du plus grand indice k tel que $tab[k]$ contienne la plus grande valeur du tableau.
6. Modifier le programme pour qu'il fonctionne même quand $tab.length = 0$. Par convention, on supposera que le maximum est alors $-\infty$ (valeur supposée représentable informatiquement) et que la position du maximum est -1. On montrera la correction partielle du programme proposé.

Exercice 3.2
On considère le programme suivant :

prog
```
x = a
y = b
while(y != 0)
  r = reste de la division de x par y
  x = y
  y = r
end while
```

Montrer que $\{P\}$ `prog` $\{Q\}$ est vrai si

$$P = a > 0 \wedge b > 0$$
$$Q = x \text{ est le PGCD de } a \text{ et } b$$

4 Correction totale de boucles

L'analyse de la correction *totale* des boucles est délicate. On doit à la fois proposer un invariant de boucle pour obtenir une postcondition intéressante, mais aussi introduire un **variant** de boucle pour prouver que la boucle s'arrête bien.

On se donne donc une expression V à valeur entière et un invariant I tel que le triplet suivant soit vrai

$$\langle I \wedge b \wedge (V = n) \rangle \ \texttt{prog} \ \langle I \wedge (V < n) \rangle.$$

Si on suppose en outre que $I \Rightarrow V \geq 0$, alors le triplet suivant est vrai

$$\langle I \rangle \ \texttt{while(}b\texttt{) prog} \ \langle I \wedge \neg b \rangle$$

Exercice 4.1
Montrer que le triplet suivant est vrai

$$\langle x \geq 0 \rangle \ \texttt{while(}x > 0\texttt{) x=x-1} \ \langle x = 0 \rangle$$

Exercice 4.2
On considère la précondition $y > 0$ et le programme suivant

prog
```
while y<=r
  r = r - y
  q = q + 1
end while
```

Donner une postcondition Q qui rende le programme totalement correct par rapport à P et Q.