

Preuves de programmes – TP n° 6

Logique de Hoare

Les règles d'inférence de la logique de Hoare sont données par :

$$\frac{}{\{P\} \text{nop} \{P\}} \quad \frac{}{\{P[x := E]\} x := E \{P\}}$$

$$\frac{\{P\} C \{Q\} \quad \{Q\} D \{R\}}{\{P\} C; D \{R\}}$$

$$\frac{\{E = \text{true} \wedge P\} C \{Q\} \quad \{E = \text{false} \wedge P\} D \{Q\}}{\{P\} \text{if } E \text{ then } C \text{ else } D \{R\}}$$

$$\frac{\{E = \text{true} \wedge I \wedge V = z\} C \{I \wedge V < z\} \quad I \Rightarrow V \geq 0}{\{I\} \text{while } E \text{ do } C \text{ done} \{E = \text{false} \wedge I\}}$$

$$\frac{P' \Rightarrow P \quad \{P\} C \{Q\} \quad Q \Rightarrow Q'}{\{P'\} C \{Q'\}}$$

(où C, D désignent des commandes, E, V des expressions sans effets de bord).

De ces règles se déduit un algorithme très simple permettant de calculer la précondition la plus faible (*weakest precondition*) à partir d'une commande C et d'une postcondition Q . On notera que le calcul de cette précondition $\mathbf{WP}(C, Q)$ nécessite d'annoter chaque boucle while par un invariant et un variant (syntaxe : `{inv P var V }`), et que ce calcul est susceptible de produire des obligations de preuve :

$$\begin{aligned} \mathbf{WP}(\text{nop}, Q) &\equiv Q \\ \mathbf{WP}(x := E, Q) &\equiv Q[x := E] \\ \mathbf{WP}(C; D, Q) &\equiv \mathbf{WP}(C, \mathbf{WP}(D, Q)) \\ \mathbf{WP}(\text{if } E \text{ then } C \text{ else } D, Q) &\equiv \\ & (E = \text{true} \Rightarrow \mathbf{WP}(C, Q)) \wedge (E = \text{false} \Rightarrow \mathbf{WP}(D, Q)) \\ \mathbf{WP}(\text{while } E \text{ do } C \text{ done}\{\text{inv } I \text{ var } V\}, Q) &\equiv I \\ & + 3 \text{ obligations de preuve : } (E = \text{true} \wedge I \wedge V = z) \Rightarrow \mathbf{WP}(C, I \wedge V < z) \\ & \quad I \Rightarrow V \geq 0 \\ & \quad (E = \text{false} \wedge I) \Rightarrow Q \end{aligned}$$

Pour montrer que $\{P\} C \{Q\}$, il suffit alors de montrer la proposition $P \Rightarrow \mathbf{WP}(C, Q)$ ainsi que toutes les obligations de preuve engendrées par le calcul de $\mathbf{WP}(C, Q)$.

Exercice 1 (Boucle for) Soit la construction : `for i := E to F do C done`.

1. Écrire la règle de Hoare correspondant à cette construction.
2. Déterminer la règle de calcul de la plus faible précondition correspondante. Quelle annotation(s) est-il nécessaire d'ajouter pour que ce calcul soit possible ?

On pourra déduire ces résultats en utilisant le codage de `for` à l'aide de `while`.

Exercice 2 (Calcul de minimum) On considère le programme suivant, qui calcule le minimum d'une fonction entre 1 et n.

```
m := f(1) ;
for i := 2 to n do
  if f(i) < m then m := f(i)
done
```

1. Quelles sont la précondition et la post-condition de ce programme ? Comment la boucle for doit-elle être annotée ?
2. Calculer les obligations de preuves correspondantes, et vérifier qu'elles sont satisfaites.

Exercice 3 (Tableaux) En logique de Hoare, les tableaux sont manipulés à l'aide des deux fonctions suivantes :

- La fonction `access(t, i)` qui retourne le i -ème élément du tableau t ;
- La fonction `store(t, i, v)` qui retourne un nouveau tableau ayant les mêmes éléments que t , sauf le i -ème qui est remplacé par v .

Ces deux fonctions permettent de définir la lecture et l'écriture dans un tableau :¹

$$t[E] \equiv \text{access}(t, E) \quad \text{et} \quad t[E] := F \equiv t := \text{store}(t, E, F)$$

1. Quels axiomes est-il raisonnable de supposer sur les fonctions `access` et `store` ?
2. Montrer la correction du programme suivant (où x et y sont des variables fraîches) :

$$\{t[i] = x \wedge t[j] = y\} \quad v := t[i]; \quad t[i] := t[j]; \quad t[j] := v \quad \{t[i] = y \wedge t[j] = x\}$$

Exercice 4 (Tri sélection) Montrer à l'aide de ce qui précède la correction de l'algorithme de tri suivant :

```
for i := 0 to n - 2 do
  for j := i + 1 to n - 1 do
    if t[j] < t[i] then begin
      tmp := t[i] ;
      t[i] := t[j] ;
      t[j] := tmp
    end
  done
done
```

¹On notera que l'opération qui consiste à écrire dans une seule case du tableau est traduite en logique de Hoare par le remplacement du tableau complet.