

TD n 8 - Correction

Calcul de Hoare

Exercice 1 On rappelle la règle de la conditionnelle :

$$\frac{\hat{f}p \wedge fgS_1 \hat{f}qg \quad \hat{f}p \wedge : fgS_2 \hat{f}qg}{\hat{f}pg \text{ if } f \text{ then } S_1 \text{ else } S_2 \text{ fi } \hat{f}qg} \quad \text{où } p, q, f \geq BExpr, S_1, S_2 \geq Imp$$

Montrer sa correction.

Correction : Supposons $\not\models \hat{f}p \wedge fgS_1 \hat{f}qg$ et $\not\models \hat{f}p \wedge : fgS_2 \hat{f}qg$, et montrons :

$$\not\models \hat{f}pg \text{ if } f \text{ then } S_1 \text{ else } S_2 \text{ fi } \hat{f}qg$$

Soit donc une affectation σ telle que $\sigma \not\models p$ et $\sigma' := \llbracket \text{if } f \text{ then } S_1 \text{ else } S_2 \text{ fi} \rrbracket \sigma \notin ?$. Il s'agit de montrer $\sigma' \not\models q$. D'après la règle de transition associée à la conditionnelle, on a :

$$\sigma' = \begin{cases} \llbracket S_1 \rrbracket \sigma & \text{si } \sigma \not\models f \\ \llbracket S_2 \rrbracket \sigma & \text{sinon} \end{cases}$$

premier cas : $\sigma \not\models f$. Puisque $\sigma \not\models p$ par hypothèse, on a $\sigma \not\models p \wedge f$. Or $\hat{f}p \wedge fgS_1 \hat{f}qg$ est valide, et $\sigma' = \llbracket S_1 \rrbracket \sigma \notin ?$, donc $\sigma' \not\models q$.

deuxième cas : même chose en remplaçant S_1 par S_2 et f par $\neg f$.

Exercice 2 Étant donné une condition p et un programme S , écrire une formule de Hoare qui soit valide si et seulement si l'exécution de S boucle à partir de toute affectation vérifiant p , c'est-à-dire $\llbracket S \rrbracket \sigma = ?$ pour tout $\sigma \models p$.

Montrer par le calcul de Hoare qu'il n'y a pas de programme suivant boucle dans tout contexte où la valeur de x est strictement positive :

```
while (x>0) do
  x:=x+1
od
```

Correction : Une formule possible est $\hat{f}pgS \hat{f}\text{False}g$. Ici, il s'agit de montrer $\hat{f}x > 0gS \hat{f}\text{False}g$. On utilise la règle de la boucle :

$$\frac{\hat{f}p \wedge (x > 0)gx := x + 1 \hat{f}pg}{\hat{f}pg \text{ while } (x > 0) \text{ do } x := x + 1 \text{ od } \hat{f}p \wedge : (x > 0)g}$$

avec $p = (x > 0)$. Il faut montrer la prémisse : les formules $\hat{f}x + 1 > 0gx := x + 1 \hat{f}x > 0g$ et $p \wedge (x > 0) \rightarrow x + 1 > 0$ sont valides donc la prémisse est valide (règle de conséquence).

On conclut en remarquant que $p \wedge : (x > 0) \rightarrow \text{False}$ est valide (puis règle de conséquence).

Exercice 3 On considère l'instruction :

repeat S **until** f où $f \geq BExpr, S \geq Imp$

Écrire un programme IMP équivalent à cette instruction, en déduisant la règle d'inférence de sa correction.

Correction : L'instruction est équivalente au programme :

$S; \text{while} : f \text{ do } S \text{ od}$

De l'arbre de preuve :

$$\frac{\frac{\frac{f_q \wedge : fgS \ f_qg}{q \vdash q} \quad \frac{f_qg \ \text{while} : f \text{ do } S \text{ od} \ f_q \wedge : : fg}{f_qg \ \text{while} : f \text{ do } S \text{ od} \ f_q \wedge fg}}{fpgS \ f_qg} \quad \frac{q \wedge : : f \vdash q \wedge f}{f_qg \ \text{while} : f \text{ do } S \text{ od} \ f_q \wedge fg}}{fpgS; \text{while} : f \text{ do } S \text{ od} \ f_q \wedge fg}$$

on déduit la règle :

$$\frac{fpgS \ f_qg \quad f_q \wedge : fgS \ f_qg}{fpg \ \text{repeat } S \ \text{until } f \ f_q \wedge fg} \quad \text{où } p, q, f \geq BExpr, S \geq Imp$$

La correction des règles utilisées dans l'arbre de preuve implique la correction de cette règle.

Exercice 4 Considérons le programme suivant :

```

y1 := 0;
y2 := 1;
y3 := 1;
while (y3 < x) do
  y1 := y1 + 1;
  y2 := y2 + 2;
  y3 := y3 + y2
od

```

Nous allons montrer avec le calcul de Hoar qu'il calcule la racine carrée. Plus précisément, qu'il est partiellement correct par rapport à la pré-condition $(x \geq 0)$ et à la post-condition $(y_1 = y_1^2 \wedge (y_1 + 1)^2 > x)$.

Appelons S_0 le sous-programme constitué des 3 premières instructions, et S le sous-programme qui forme le corps de la boucle **while**.

1. Calculer les valeurs que prennent les variables y_1, y_2 et y_3 au début des premières exécutions du corps du **while**. Conjecturer la valeur de y_2 et y_3 en fonction de y_1 . On notera p la conjonction des 2 égalités et de $(y_1 = y_1^2 \wedge (y_1 + 1)^2 > x)$.
2. p va être l'invariant de la boucle. C'est-à-dire qu'il faut montrer :

$$fpg \ \text{while} (y_3 < x) \text{ do } S \text{ od} \ f_p \wedge : (y_3 < x)g$$

en utilisant la règle d'inférence du **while**. Écrire la prémisses qu'il faut utiliser, et la montrer par le calcul de Hoar (indication : partir de la post-condition pour trouver les conditions intermédiaires).

3. Montrer par le calcul de Hoar que p est vérifié après les 3 premières instructions du programme, sous la condition $(x \geq 0)$. C'est-à-dire, montrer que $f_x \geq 0gS_0 \ fpg$ est valide.
4. Conclure.

Correction :

1. $y_2 = 2 \cdot y_1 + 1$ et $y_3 = (y_1 + 1)^2$. Donc p est :

$$\underbrace{(y_2 = 2 \cdot y_1 + 1)}_{f_1} \wedge \underbrace{(y_3 = (y_1 + 1)^2)}_{f_2} \wedge \underbrace{(y_1 = y_1^2 \wedge (y_1 + 1)^2 > x)}_{f_3}$$

2. La règle s'écrit ici :

$$\frac{\hat{r}p \wedge (y_3 = x)g \ S \ \hat{r}pg}{\hat{r}pg \ \mathbf{while} \ (y_3 = x) \ \mathbf{do} \ S \ \mathbf{od} \ \hat{r}p \wedge : (y_3 = x)g}$$

On part de la fin de S . Par simples applications de la règle de l'affectation, on a :

$$\begin{array}{lll} \hat{r}p'g & y_3 := y_3 + y_2 & \hat{r}pg \\ \hat{r}p''g & y_2 := y_2 + 2 & \hat{r}p'g \\ \hat{r}p'''g & y_1 := y_1 + 1 & \hat{r}p''g \end{array}$$

avec

$$\begin{array}{llll} p' = & f_1 & \wedge & (y_3 + y_2 = (y_1 + 1)^2) \quad \wedge \quad f_3 \\ p'' = & (y_2 + 2 = 2 \ y_1 + 1) & \wedge & (y_3 + y_2 + 2 = (y_1 + 1)^2) \quad \wedge \quad f_3 \\ p''' = & (y_2 + 2 = 2 \ (y_1 + 1) + 1) & \wedge & (y_3 + y_2 + 2 = (y_1 + 2)^2) \quad \wedge \quad ((y_1 + 1)^2 = x) \end{array}$$

ce qui, par règle de composition, donne $\hat{r}p'''g \ S \ \hat{r}pg$.

Or $p''' \not\models (y_2 = 2 \ y_1 + 1) \wedge (y_3 = (y_1 + 1)^2) \wedge (y_3 = x)$ c'est-à-dire $f_1 \wedge f_2 \wedge (y_3 = x)$. Mais cette formule est une conséquence de $p \wedge (y_3 = x)$. Donc par règle de conséquence on obtient la prémisse