Langages, Traducteurs et Sémantique, M1, 2006/2007
J.-C. Fernandez, Y. Lakhnech, C. Éne, L. Gonnord, F. Lagnier

Examen janvier 2007, éléments de correction

Exercice I

Question 1 $B_D(n : S, LS) = \begin{cases} B_D(LS) \cup \{n\} & \text{si } n \notin B_D(LS) \\ \texttt{\{erreur\}} & \text{sinon} \end{cases}$

Question 2

$$\frac{(LS, \sigma, v) \to \sigma' \quad A[a]\sigma = v}{(\texttt{case } a \texttt{ of } LS \texttt{ endcase}, \sigma) \to \sigma'}$$

$$(n : S, \sigma, v) \to \sigma \text{ si } \mathcal{N}(n) \neq v \qquad \frac{(S, \sigma) \to \sigma' \; \mathcal{N}(n) = v}{(n : S, \sigma, v) \to \sigma'}$$

$$\frac{(n : S, \sigma, v) \to \sigma' \; \mathcal{N}(n) = v}{(n : S, LS, \sigma, v) \to \sigma'} \qquad \frac{(LS, \sigma, v) \to \sigma' \mathcal{N}(n) \neq v}{(n : S, LS, \sigma, v) \to \sigma'}$$

Hoare

$$\frac{\bigwedge_{i=1}^{k} \{a = n_i \wedge P\} S_i \{Q\} \qquad \bigwedge_{i=1}^{k} a \neq n_i \Longrightarrow (P \Longrightarrow Q)}{\{P\}\texttt{case } a \texttt{ of } n_1 : S_1, \cdots n_k : S_k \texttt{ endcase}\{Q\}}$$

Exercice II - Types

Question 1 Pour la soustraction, le seul cas qui ammène de l'information sont `Neg - Plus = Neg` :

$$\frac{(\Gamma, a_1) \to \texttt{Neg} \quad (\Gamma, a_2) \to \texttt{Pos}}{(\Gamma, a_1 - a_2) \to \texttt{Neg}} \qquad \frac{(\Gamma, a_1) \to \texttt{Pos} \quad (\Gamma, a_2) \to \texttt{Neg}}{(\Gamma, a_1 - a_2) \to \texttt{Pos}}$$

$$\frac{(\Gamma, a_1) \to \texttt{Neg} \quad (\Gamma, a_2) \to \texttt{Neg}}{(\Gamma, a_1 - a_2) \to \texttt{Int}} \qquad \frac{(\Gamma, a_1) \to \texttt{Pos} \quad (\Gamma, a_2) \to \texttt{Pos}}{(\Gamma, a_1 - a_2) \to \texttt{Int}}$$

Pour la multiplication le résultat est plus précis :

$$\frac{(\Gamma, a_1) \to \texttt{Neg} \quad (\Gamma, a_2) \to \texttt{Neg}}{(\Gamma, a_1 * a_2) \to Pos} \qquad \frac{(\Gamma, a_1) \to \texttt{Pos} \quad (\Gamma, a_2) \to \texttt{Pos}}{(\Gamma, a_1 * a_2) \to \texttt{Pos}}$$

$$\frac{(\Gamma, a_1) \to \texttt{Neg} \quad (\Gamma, a_2) \to \texttt{Pos}}{(\Gamma, a_1 * a_2) \to \texttt{Neg}} \qquad \frac{(\Gamma, a_1) \to \texttt{Pos} \quad (\Gamma, a_2) \to \texttt{Neg}}{(\Gamma, a_1 * a_2) \to \texttt{Neg}}$$

Question 2 Composition séquentielle : les informations sont transmises de la première liste d'instructions à la deuxième :

$$\frac{(\Gamma, S_1) \to \Gamma_1 \quad (\Gamma, S_2) \to \Gamma_2}{(\Gamma, S_1; S_2) \to \Gamma_2}$$

Pour le if-then-else, on se rappelle que l'on fait du statique, et donc on réunit les informations des deux branches :

$$\frac{(\Gamma, b) \to ok \quad (\Gamma, S_1) \to \Gamma_1 \quad (\Gamma, S_2) \to \Gamma_2}{(\Gamma, \text{if } b \text{ then } S_1 \text{ else } S_2) \to \Gamma_1 \sqcup \Gamma_2}$$

Pertie Facultative (Enlever la question vide)

1. On montre par induction sur a :
   - si a est une constante positive, alors $(\Gamma, a) \to$ `Pos`) alors l'implication droite du "et" est vraie (prémisse fausse). L'implication gauche est aussi vraie puisque $[a]\sigma = p \geq 0$ (relation $\mathcal{R}$)
   - si a est une constante négative, preuve similaire.
   - si x est une variable, soit elle est positive ou nulle, soit elle est strictement négative, ce qui est donné par $[x]\sigma$. preuve similaire aux deux précédentes.
   - si $a = a_1 + a_2$, supposons $(\Gamma, a) \to$ `Pos`. Alors d'après la règle de typage on a $(\Gamma, a_1) \to$ `Pos` et $(\Gamma, a_2) \to Pos$), on peut donc appliquer l'HR et on obtient donc $[a_1]\sigma \geq 0$ et de même pour a_2. Ensuite, comme $[a_1 + a_2]\sigma = [a_1]\sigma + [a_2]\sigma$, cette quantité est positive. Le cas `Neg` est similaire.
   - Démos similaires
2. On montre par induction sur S LE DIRE ? ? ? et flemme.

Exercice III - Variables actives

1. Initialisation :

B_i	Gen	$Kill$
B_1	$\emptyset$	i, j
B_2	i, N	$\emptyset$
B_3	i, j, c	t_1, t_2
B_4	i, j	i
B_4	j	$\emptyset$

2. Itérations

B_i	In	Out	In	Out	In	Out	In	Out
B_1	$\emptyset$	i, N	N	i, j, c, N	c, N	i, j, c, N	cN	$idem$
B_2	i, N	i, j, c	i, j, c, N	i, j, c	i, j, c, N	i, j, c, N	i, j, c, N	$idem$
B_3	i, j, c	i, j	i, j, c	i, N, j	i, j, c, N	i, j, c, N	i, j, c, N	$idem$
B_4	i, j	i, N	N, i, j	i, j, c, N	i, j, c, N	i, j, c, N	j, c, N	$idem$
B_5	j	$\emptyset$	j	$\emptyset$	j	$\emptyset$	j	$idem$

3. Conclusions : t_1 et t_2 sont inactives en fin du bloc 3, donc les affectations (4) et (5) peuvent être supprimées.

Génération de Code

1. La pile est donnée Figure 1.

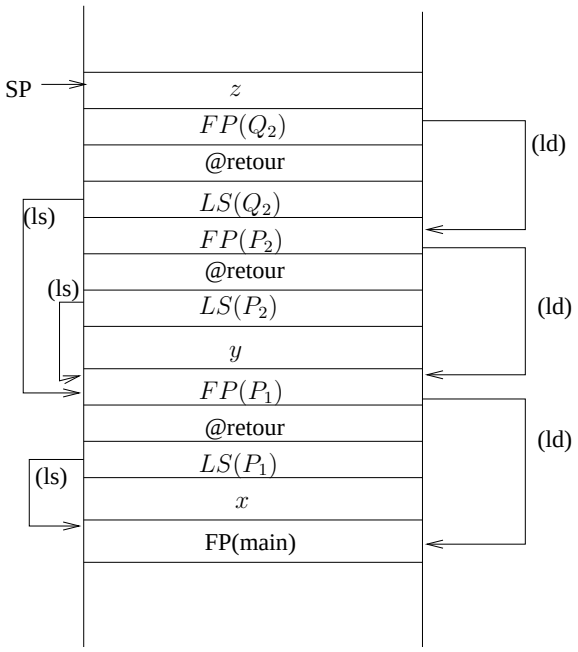

FIG. 1 – Pile à l'exécution de Q2

2. La variable y (dans P_2) est définie dans la procédure P_1, c'est-à-dire la procédure directement englobante, donc :

```
LD R1, [FP+8] -- 1 indirection sur le LS pour récupérer fp(P1)
ADD R2, R0, 3  -- R0 <- 3
ST R2, [R1-4] -- y est la seule variable locale de P1
```

3. Pour l'appel de Q_2, on empile l'environnement de définition de Q_2, donc une indirection sur le lien statique :

```
LD R1, [FP+8]
empiler (R1) ????
call Q2
ADD SP, SP, 4
```

4. La variable z est une variable locale, la variable y est locale à la procédure directement englobante, la variable x est un cran plus haut :

```
LD R1 [FP-4]   -- R1<-z
LD R2 [FP+8]
LD R3 [R2-4]   -- R3<-y
LD R4 [R2+8]
LD R5 [R4-4]   -- R5<-x
ADD R6 R1 R3
ADD R7 R6 R5   -- R7 <-x+y+z
ST R7 [R4-4]   -- x<- R7
```

5. Dans $P1$, la variable x est une variable locale à la procédure directement englobante :

```
LD R1, [FP+8]
LD R2, [R1-4]
ADD R3, R2, 10
ST R3, [FP+12]
```

6. Dans la procédure main, x est locale, la seule difficulté est de récupérer la valeur de retour de la procédure $P1$.

```
ADD SP, SP, -4  -- pour le résultat
empiler(FP)
CALL P1
ADD SP, SP, 4
LD R1, [SP]
ADD SP, SP, 4
ADD R1, R1, 2
ST R1, [FP -4]
```