Sémantique des langages de programmation
Examen final

ÉNSIIE, semestre 5

mardi 23 octobre 2018

Durée : 1h45.
Aucun document ni aucun appareil électronique autorisé.
Ce sujet comporte 3 exercices indépendants, qui peuvent être traités dans l'ordre voulu. Il contient 4 pages.
Le barème est donné à titre indicatif, il est susceptible d'être modifié. Le total est sur 20 points.
Certaines questions, précédées par le symbole ($\star$) sont plus difficiles et pourront être traitées à la fin. Il va de soi que toute réponse devra être justifiée.

Exercice 1 : Équivalence de programmes (5 points)

Montrer l'équivalence ou la non-équivalence des programmes IMP suivants :

1. `x := 1+1; y := 2` et `y := 1+1; x := 2`
2. `while not (0 = 0) do y := 1` et `skip`
3. `while 0 = 0 do y := 1` et `y := 1`
4. `while b do c` et `while b and true do c`
5. `while b do c` et `while true and b do c`

Exercice 2 : IMP + pointeurs (8 points)

On souhaite étendre le langage IMP avec la gestion de pointeurs. Pour cela, on étend la syntaxe des expressions et des instructions avec les constructions suivantes :

$$a ::= n \mid x \mid a_1 + a_2 \mid a_1 - a_2 \mid a_1 \times a_2 \mid a_1/a_2 \mid \boldsymbol{\&x} \mid \boldsymbol{*a}$$
$$c ::= \mathsf{skip} \mid x := a \mid c_1; c_2 \mid \mathsf{if}\ b\ \mathsf{then}\ c_1\ \mathsf{else}\ c_2 \mid \mathsf{while}\ b\ \mathsf{do}\ c \mid \boldsymbol{a_1 \leftarrow a_2}$$

où x représente un nom de variable, a, a_1, a_2 des expressions arithmétiques, b une expression booléenne (inchangée par rapport au cours) c, c_1, c_2 des instructions.

$$(A_1)\frac{}{\langle n,\sigma\rangle \rightsquigarrow n}\ (n\in\mathbb{Z}) \qquad (A_2)\frac{}{\langle x,\sigma\rangle \rightsquigarrow \sigma(x)}\ (x\in V)$$

$$(A_3)\frac{\langle a_1,\sigma\rangle \rightsquigarrow n_1 \quad \langle a_2,\sigma\rangle \rightsquigarrow n_2}{\langle a_1+a_2,\sigma\rangle \rightsquigarrow n_1[\![+]\!]n_2} \quad (A_4)\frac{\langle a_1,\sigma\rangle \rightsquigarrow n_1 \quad \langle a_2,\sigma\rangle \rightsquigarrow n_2}{\langle a_1\times a_2,\sigma\rangle \rightsquigarrow n_1[\![\times]\!]n_2}$$

$$(A_5)\frac{\langle a_1,\sigma\rangle \rightsquigarrow n_1 \quad \langle a_2,\sigma\rangle \rightsquigarrow n_2}{\langle a_1-a_2,\sigma\rangle \rightsquigarrow n_1[\![-]\!]n_2} \quad (A_6)\frac{\langle a_1,\sigma\rangle \rightsquigarrow n_1 \quad \langle a_2,\sigma\rangle \rightsquigarrow n_2}{\langle a_1/a_2,\sigma\rangle \rightsquigarrow n_1[\![/]\!]n_2}$$

FIGURE 1 – Sémantique opérationnelle à grands pas des expressions arithmétiques

$$(C_1)\frac{}{\langle \mathsf{skip},\sigma\rangle \to \sigma} \quad (C_2)\frac{\langle a,\sigma\rangle \rightsquigarrow n}{\langle x:=a,\sigma\rangle \to \sigma[x\leftarrow n]}\ (n\in\mathbb{Z}) \quad (C_3)\frac{\langle c_1,\sigma\rangle \to \sigma_1 \quad \langle c_2,\sigma_1\rangle \to \sigma_2}{\langle c_1;c_2,\sigma\rangle \to \sigma_2}$$

$$(C_4)\frac{\langle b,\sigma\rangle \rightsquigarrow \mathsf{true} \quad \langle c_1,\sigma\rangle \to \sigma_1}{\langle \mathsf{if}\ b\ \mathsf{then}\ c_1\ \mathsf{else}\ c_2,\sigma\rangle \to \sigma_1} \qquad (C_5)\frac{\langle b,\sigma\rangle \rightsquigarrow \mathsf{false} \quad \langle c_2,\sigma\rangle \to \sigma_2}{\langle \mathsf{if}\ b\ \mathsf{then}\ c_1\ \mathsf{else}\ c_2,\sigma\rangle \to \sigma_2}$$

$$(C_6)\frac{\langle b,\sigma\rangle \rightsquigarrow \mathsf{false}}{\langle \mathsf{while}\ b\ \mathsf{do}\ c,\sigma\rangle \to \sigma} \qquad (C_7)\frac{\langle b,\sigma\rangle \rightsquigarrow \mathsf{true} \quad \langle c,\sigma\rangle \to \sigma_1 \quad \langle \mathsf{while}\ b\ \mathsf{do}\ c,\sigma_1\rangle \to \sigma_2}{\langle \mathsf{while}\ b\ \mathsf{do}\ c,\sigma\rangle \to \sigma_2}$$

FIGURE 2 – Sémantique opérationnelle à grands pas de IMP

L'intuition est que $\&x$ représente l'adresse de la variable x, $*a$ le contenu en mémoire à l'adresse calculée par a, et $a_1 \leftarrow a_2$ met la valeur calculée par a_2 à l'adresse calculée par a_1. (En C, cela s'écrirait `*a1 = a2`.)

On va utiliser un modèle mémoire très simple pour représenter la sémantique de ce langage. On va considérer que n'importe quel entier correspond à une adresse valide, et que chaque adresse-mémoire contient un entier. Pour cela, on ne considère plus seulement une valuation qui associe à une variable directement une valeur, mais une première valuation α qui associe à une variable son adresse et une deuxième valuation ν qui associe à une adresse la valeur présente à cette adresse. Les configurations seront donc de la forme $\langle c,\alpha,\nu\rangle$.

1. Étendre la sémantique opérationnelle à grands pas des expressions arithmétiques et de IMP (cf. figures 1 et 2) pour prendre en compte ces changements. Les jugements seront donc de la forme respectivement $\langle a,\alpha,\nu\rangle \rightsquigarrow n$ et $\langle c,\alpha,\nu\rangle \to \langle \alpha',\nu'\rangle$. On indiquera en particulier comment les règles (A_2) et (C_2) sont modifiées.
2. Étendre la sémantique opérationnelle à petits pas des expressions arithmétiques et de IMP (cf. figures 3 et 4). Les jugements seront donc de la forme $\langle e,\alpha,\nu\rangle \hookrightarrow \langle e',\alpha',\nu'\rangle$. On indiquera en particulier comment les règles (AS_1) et (S_1) sont modifiées.
3. À partir de valuations quelconques α et ν telles que $\alpha(x)\geq 0$, donner la sémantique à petits pas (en donnant les arbres d'inférence justifiant les étapes) et à grands pas du programme suivant :
   $x := \&x + 5; \mathsf{while\ not}\ (x=0)\ \mathsf{do}\ (x := x-1; x \leftarrow x - \&x)$
4. Montrer que si $\langle c,\alpha,\nu\rangle \to \langle \alpha',\nu'\rangle$ alors $\alpha=\alpha'$. (Autrement dit, l'adresse des variables ne peut pas être modifié par un programme.)

$$(AS_1)\frac{}{\langle x,\sigma\rangle \hookrightarrow \langle \sigma(x),\sigma\rangle}(\text{si } x \in V) \qquad (AS_4)\frac{}{\langle n_1+n_2,\sigma\rangle \hookrightarrow \langle n,\sigma\rangle}(\text{si } n_1,n_2,n \in \mathbb{Z}, n = n_1[\![+]\!]n_2)$$

$$(AS_7)\frac{\langle e_1,\sigma\rangle \hookrightarrow \langle e_1',\sigma\rangle}{\langle e_1+e_2,\sigma\rangle \hookrightarrow \langle e_1'+e_2,\sigma\rangle} \qquad (AS_8)\frac{\langle e_2,\sigma\rangle \hookrightarrow \langle e_2',\sigma\rangle}{\langle n_1+e_2,\sigma\rangle \hookrightarrow \langle n_1+e_2',\sigma\rangle}(\text{si } n_1 \in \mathbb{Z})$$

FIGURE 3 – Sémantique opérationnelle à petits pas des expressions arithmétiques

$$(S_1)\frac{\langle e,\sigma\rangle \hookrightarrow \langle n,\sigma\rangle}{\langle x := e,\sigma\rangle \hookrightarrow \langle \mathsf{skip},\sigma[x \leftarrow n]\rangle} \qquad (S_2)\frac{\langle c_1,\sigma\rangle \hookrightarrow \langle c_1',\sigma'\rangle}{\langle c_1;c_2,\sigma\rangle \hookrightarrow \langle c_1';c_2,\sigma'\rangle} \qquad (S_3)\frac{}{\langle \mathsf{skip};c,\sigma\rangle \hookrightarrow \langle c,\sigma\rangle}$$

$$(S_4)\frac{\langle b,\sigma\rangle \rightsquigarrow \mathsf{true}}{\langle \mathsf{if}\ b\ \mathsf{then}\ c_1\ \mathsf{else}\ c_2,\sigma\rangle \hookrightarrow \langle c_1,\sigma\rangle} \qquad (S_5)\frac{\langle b,\sigma\rangle \rightsquigarrow \mathsf{false}}{\langle \mathsf{if}\ b\ \mathsf{then}\ c_1\ \mathsf{else}\ c_2,\sigma\rangle \hookrightarrow \langle c_2,\sigma\rangle}$$

$$(S_6)\frac{\langle b,\sigma\rangle \rightsquigarrow \mathsf{false}}{\langle \mathsf{while}\ b\ \mathsf{do}\ c,\sigma\rangle \hookrightarrow \langle \mathsf{skip},\sigma\rangle} \qquad (S_7)\frac{\langle b,\sigma\rangle \rightsquigarrow \mathsf{true}}{\langle \mathsf{while}\ b\ \mathsf{do}\ c,\sigma\rangle \hookrightarrow \langle c;\mathsf{while}\ b\ \mathsf{do}\ c,\sigma\rangle}$$

FIGURE 4 – Sémantique opérationnelle à petits pas de IMP

5. ($\star$) Montrer l'équivalence entre les sémantiques opérationnelles à petits et à grands pas. On pourra ne considérer que les cas qui changent.
6. On part de la valuation $\alpha_0 = \{x \mapsto 0, y \mapsto 3\}$ et
   $\nu_0 = \{0 \mapsto 1, 1 \mapsto 2, 2 \mapsto 0, 3 \mapsto 4, 4 \mapsto 5, 5 \mapsto 6, 6 \mapsto 7\}$. Donner la sémantique à petits pas et à grands pas de
   while not $(*x = 0)$ do $(y \leftarrow *x; y := y + 1; x := x + 1)$.
   On donnera les arbres de dérivation.
7. Quelle est la sémantique du même programme si on part de $\alpha_0' = \{x \mapsto 0, y \mapsto 2\}$ et du même ν_0 ?
8. Les programmes ou expressions suivantes sont-elles équivalentes ? Si oui le démontrer, sinon donner un contre exemple.
   a) $\&x \leftarrow a$ et $x := a$
   b) $*(\&x)$ et x
   c) $x \leftarrow a; y := *x$ et $y := a$
   d) $x \leftarrow *x + 1; y \leftarrow *y \times 2$ et $y \leftarrow *y \times 2; x \leftarrow *x + 1$
   e) $tmp := *x; x \leftarrow *y; y \leftarrow tmp$ et
      $x \leftarrow *x + *y; y \leftarrow *x - *y; x \leftarrow *x - *y$

Exercice 3 : Enregistrements en MiniML (7 points)

On étend la syntaxe du langage MiniML vu en cours avec le possibilité de créer des enregistrements et d'accéder à leurs champs :

$$e ::= n \mid x \mid \texttt{fun}\ x\ \texttt{->}\ e \mid e\ e \mid e + e \mid \boldsymbol{\{l_1 = e_1, \ldots, l_n = e_n\}} \mid \boldsymbol{e.l}$$

où $l, l_1, \ldots, l_n$ sont des étiquettes.

$$(F_1)\frac{}{n \to_v n} \qquad (F_2)\frac{}{\texttt{fun } x \texttt{ -> } e \to_v \texttt{fun } x \texttt{ -> } e}$$

$$(F_{v3})\frac{e_1 \to_v \texttt{fun } x \texttt{ -> } e \qquad e_2 \to_v v \qquad \{v/x\}e \to_v v'}{e_1\ e_2 \to_v v'}$$

$$(F_4)\frac{e_1 \to_v v_1 \qquad e_2 \to_v v_2}{e_1 + e_2 \to_v v_1 [\![+]\!] v_2}$$

FIGURE 5 – Sémantique opérationnelle à grands pas par valeur de MiniML

$$(FS_{v1})\frac{}{(\texttt{fun } x \texttt{ -> } e)v \hookrightarrow_v \{v/x\}e} \qquad (FS_{v2})\frac{e_1 \hookrightarrow_v e_1'}{e_1\ e_2 \hookrightarrow_v e_1'\ e_2} \qquad (FS_{v3})\frac{e_2 \hookrightarrow_v e_2'}{v\ e_2 \hookrightarrow_v v\ e_2'}$$

$$(FS_{v4})\frac{}{v_1 + v_2 \hookrightarrow_v v_1 [\![+]\!] v_2} \qquad (FS_{v5})\frac{e_1 \hookrightarrow_v e_1'}{e_1 + e_2 \hookrightarrow_v e_1' + e_2} \qquad (FS_{v6})\frac{e_2 \hookrightarrow_v e_2'}{v + e_2 \hookrightarrow_v v + e_2'}$$

FIGURE 6 – Sémantique opérationnelle à petits pas par valeur de MiniML

1. Étendre la sémantique à grand pas par valeur de MiniML (rappelée figure 5) pour donner une sémantique aux enregistrements et à l'accès à un champ. Informellement, un enregistrement est une valeur si chacun des ses champs est une valeur. Un enregistrement est évalué en évaluant chacun de ses champs. L'accès à un champ d'un enregistrement évalue d'abord l'enregistrement puis renvoie la valeur de ce champ dans l'enregistrement.
2. Donner la sémantique à grands pas par valeur des expressions suivantes :
   a) $\{x = 2 + 3, y = (\texttt{fun } x \texttt{ -> } 2)1\}$
   b) $(\texttt{fun } x \texttt{ -> } \{double = x + x, triple = x + x + x\})\ 2$
   c) $(\texttt{fun } f \texttt{ -> } \{zero = f\ 0, succ = f\ (\texttt{fun } x \texttt{ -> } x + 1)\})\ (\texttt{fun } y \texttt{ -> } y)$
   d) $\{x = 1 + 3, y = 2 + 1\}.x$
   e) $(\texttt{fun } x \texttt{ -> } x.left)\ \{left = 2 + 1, right = 3 + 2\}$
   f) $(\texttt{fun } e \texttt{ -> } \{x = e, pred = e.x\})\ \{x = 0\}$
3. Étendre la sémantique à petit pas par valeur de MiniML (rappelée figure 6) pour donner une sémantique aux enregistrements et à l'accès aux champs.
4. Donner la sémantique à petits pas des expressions de la question 2.
5. ($\star$) Démontrer l'équivalence des sémantiques opérationnelles à grands pas et à petit pas.
6. Quels changements faut-il apporter aux sémantiques pour avoir une sémantique d'appel par nom ?