Calculabilité et décidabilité : TD3

Licence Info 3 - Michele Pagani

14 février 2012

1 Sémantique des programmes de WHILE

1.1 Évaluation des expressions

On rappelle que l'évaluation des expressions est une fonction qui associe à une expression E une fonction totale $\mathcal{E}[\![\mathtt{E}]\!]$ qui envoie une évaluation σ des variables de E (appelée *store*) sur un arbre $\mathcal{E}[\![\mathtt{E}]\!]\sigma \in \mathbb{D}$.

$$\begin{array}{ccccc} \texttt{Expressions} & \to & (\texttt{Stores} & \to & \mathbb{D}) \\ \texttt{E} & \mapsto & (\sigma & \mapsto & \mathcal{E}[\![\texttt{E}]\!]\sigma) \end{array}$$

La définition de $\mathcal{E}[\![\mathtt{E}]\!]\sigma$ est donnée par induction structurelle sur E selon les règles suivantes :

$$\mathcal{E}[\![\mathtt{X}]\!]\sigma = \sigma(\mathtt{X})$$

$$\mathcal{E}[\![\mathtt{d}]\!]\sigma = d$$

$$\mathcal{E}[\![\texttt{cons E F}]\!]\sigma = (\mathcal{E}[\![\mathtt{E}]\!]\sigma.\mathcal{E}[\![\mathtt{F}]\!]\sigma)$$

$$\mathcal{E}[\![\texttt{hd E}]\!]\sigma = \begin{cases} \texttt{e} & \text{si } \mathcal{E}[\![\mathtt{E}]\!]\sigma = \texttt{(e.f)} \\ \texttt{nil} & \text{si } \mathcal{E}[\![\mathtt{E}]\!]\sigma = \texttt{nil} \end{cases}$$

$$\mathcal{E}[\![\texttt{tl E}]\!]\sigma = \begin{cases} \texttt{f} & \text{si } \mathcal{E}[\![\mathtt{E}]\!]\sigma = \texttt{(e.f)} \\ \texttt{nil} & \text{si } \mathcal{E}[\![\mathtt{E}]\!]\sigma = \texttt{nil} \end{cases}$$

$$\mathcal{E}[\![\texttt{=? E F}]\!]\sigma = \begin{cases} \texttt{true} & \text{si } \mathcal{E}[\![\mathtt{E}]\!]\sigma = \mathcal{E}[\![\mathtt{F}]\!]\sigma \\ \texttt{false} & \text{sinon} \end{cases}$$

Exercice 1. Étant donnés les *stores* suivants

$$\sigma = \begin{cases} X \mapsto & \texttt{nil} \\ Y \mapsto & \texttt{(nil.nil)} \\ Z \mapsto & \underline{2} \end{cases} \qquad \sigma' = \begin{cases} X \mapsto & \texttt{(nil.nil)} \\ Y \mapsto & \underline{1} \\ Z \mapsto & \texttt{((nil.nil) (nil.nil) nil)} \end{cases}$$

Calculer les évaluations $\mathcal{E}[\![\mathtt{E}]\!]\sigma$ et $\mathcal{E}[\![\mathtt{E}]\!]\sigma'$ des expressions :

```
cons Y (cons nil X)        cons X (cons nil Y)

   =? X Y        tl (hd Z)        =? (hd Z) Y
```

1.2 Sémantique des commandes

On rappelle que l'évaluation des commandes est la plus petite relation $\vdash \subseteq \texttt{Commands} \times \texttt{Stores} \times \texttt{Stores}$ qui satisfait les règles suivantes (on note $\mathtt{C} \vdash \sigma \to \sigma'$ pour dire que le triplet $(\mathtt{C}, \sigma, \sigma')$ appartient à la relation $\vdash$) :

$$\frac{\mathcal{E}[\![\mathtt{E}]\!]\sigma = \mathtt{d}}{\mathtt{X:=E} \vdash \sigma \to \sigma[\mathtt{X} \mapsto \mathtt{d}]} \qquad \frac{\mathtt{C} \vdash \sigma \to \sigma' \qquad \mathtt{D} \vdash \sigma' \to \sigma''}{\mathtt{C;D} \vdash \sigma \to \sigma''} \qquad \frac{\mathcal{E}[\![\mathtt{E}]\!]\sigma = \texttt{nil}}{\texttt{while E do } \{\mathtt{C}\} \vdash \sigma \to \sigma}$$

$$\frac{\mathcal{E}[\![\mathtt{E}]\!]\sigma \neq \texttt{nil} \qquad \mathtt{C} \vdash \sigma \to \sigma' \qquad \texttt{while E do } \{\mathtt{C}\} \vdash \sigma' \to \sigma''}{\texttt{while E do } \{\mathtt{C}\} \vdash \sigma \to \sigma''}$$

Exercice 2. Décrire l'évaluation des commandes suivantes :

1. `Z:=X; X:=Y; Y:=Z`
2. `Y:= hd X; Z:= tl X; X:= cons Z Y`
3. `while X do {X:=tl X;}`

Exercice 3 (Difficile)**.** Prouver que la relation $\vdash$ est une fonction de type $\texttt{Commands} \to (\texttt{Stores} \to \texttt{Stores}_\perp)$. (*Suggestion : il faut démontrer que pour toute commande* $\texttt{C}$ *et tout store* σ *si* $\texttt{C} \vdash \sigma \to \sigma'$ *et* $\texttt{C} \vdash \sigma \to \sigma''$*, alors* $\sigma' = \sigma''$*. La preuve se fait par induction sur la longueur d'une dérivation de* $\texttt{C} \vdash \sigma \to \sigma'$*).*)

Exercice 4. Soit σ le store $\texttt{X} \mapsto \texttt{(nil.nil)}$ et `C` la commande `while X do {X:=X}`. Prouver que il n'existe aucun store σ' tel que $\texttt{C} \vdash \sigma \to \sigma'$.

1.3 Sémantique des programmes

On rappelle que la sémantique du langage `WHILE` est la fonction $[\![\bullet]\!] : \texttt{Programme} \to (\mathbb{D} \to \mathbb{D}_\perp)$ associant à un programme $\texttt{P} = \texttt{read X; C; write Y}$ la fonction $\mathbb{D} \to \mathbb{D}_\perp$ suivante

$$[\![\texttt{P}]\!](\texttt{d}) = \texttt{e} \quad \text{si} \quad \texttt{C} \vdash \sigma_0^{\texttt{P}}(\texttt{d}) \to \sigma \quad \text{et} \quad \sigma(\texttt{Y}) = \texttt{e}$$

où $\sigma_0^{\texttt{P}}(\texttt{d})$ est le store initial de `P`, associant `d` à `X` et `nil` à toute autre variable.

Exercice 5. Étant donné le programme `pred` suivant

```
read X;
  Y:=tl X;
write Y
```

montrer que $[\![\texttt{p}red]\!](\underline{n}) = \underline{n-1}$

Exercice 6. Etant donné le programme `reverse` suivant

```
read X;
  Y:=nil;
  while X do
  {
    Y:= cons (hd X) Y;
    x:= tl X;
  };
write Y
```

calculer la valeur de $[\![$ `reverse`$]\!]((\underline{1}\ \underline{2}\ \underline{3}))$, en détaillant toutes les étapes de calcul.

Exercice 7. Montrer que $[\![$ `reverse`$]\!]((\texttt{d}_1 \cdots \texttt{d}_n)) = (\texttt{d}_n \cdots \texttt{d}_1)$. (*Suggestion : par induction sur* n*.*)

Exercice 8 (Resumé)**.** De la même façon que pour le programme `reverse` (Exercice 7), montrer la correction des programmes développés dans les exercices 8 et 9 du TD2.