DEA '98: Examen du cours de Sémantique

Tous documents autorisés

17 Décembre 1997

1 Points fixes, monades

Exercice 1.1 (Points fixes)
Considérons la définition recursive pour $f : N \rightarrow N_\perp$:

$$f(k) = \begin{cases} 1 & si\ k = 0 \\ f(k+2) & sinon \end{cases}$$

- *montrez que $f^1(k) = 1$ est une solution*
- *trouvez toutes les solutions*
- *quelle est la plus petite solution?*

Exercice 1.2 (Sémantique des opérations et du if-then-else) *En utilisant l'approche modulaire de la sémantique dénotationnelle, nous pouvons définir uniformement la sémantique des opérations sur les booléens et pour le if-then-else comme suit:*

- $[\![\Gamma| - \texttt{not exp1: } Exp\texttt{[bool]}]\!]_e =$
  let $x \Leftarrow [\![\Gamma| - \texttt{exp1: } Exp\texttt{[bool]}]\!]_e$ *in val*($\underline{not}$ x)
- $[\![\Gamma| - \texttt{if b then exp1 else exp2: } Exp\texttt{[t]}]\!]_e =$
  let $v \Leftarrow [\![\Gamma| - \texttt{b: } Exp\texttt{[bool]}]\!]_e$ *in*
  IF v THEN $[\![\Gamma| - \texttt{exp1: } Exp\texttt{[t]}]\!]_e$ *ELSE* $[\![\Gamma| - \texttt{exp2: } Exp\texttt{[t]}]\!]_e$

où <u>not</u> est la fonction de negation dans les domaines sémantiques, et IF THEN ELSE est l'operation de test booleen dans les domaines sémantiques.
Démontrez, à l'aide des équations pour les monades, que dans toute sémantique on a, pour toutes expressions `exp1` *et* `exp2` *et expression booléenne* `b`*:*

$$[\![\Gamma| - \texttt{if (not b) then exp1 else exp2: } Exp\texttt{[t]}]\!]_e = \\ [\![\Gamma| - \texttt{if b then exp2 else exp1: } Exp\texttt{[t]}]\!]_e$$

et

$$[\![\Gamma| - \texttt{not (not b): } Exp\texttt{[bool]}]\!]_e = [\![\Gamma| - \texttt{b: } Exp\texttt{[bool]}]\!]_e$$

et

$$[\![\Gamma| - \texttt{(not b): } Exp\texttt{[bool]}]\!]_e = [\![\Gamma| - \texttt{if b then false else true: } Exp\texttt{[bool]}]\!]_e$$

Exercice 1.3 (Fonctions mutuellement récursives) *Nous voulons rajouter au langage TFun avec recursion la possibilité de définir deux fonctions mutuellement récursives. Donnez donc la règle de typage et la sémantique, en appel par nom, de la construction suivante:*

```
let rec f(x) = e1 and g(y) = e2 in e
```

où l'on entend que l'on souhaite utiliser dans l'environnement `e` *les fonctions* `f` *et* `g` *qui sont mutuellement recursives.*

2 Bisimulation, équivalence observationnelle et λ-calcul

Voici une définition de bisimulation sur le λ-calcul non-typé avec appel par nom.

- L'ensemble des λ-termes est définit par cette grammaire:

  $$e ::= x \;\Big|\; \lambda x.e \;\Big|\; e_1 e_2$$

  Λ^0 est l'ensemble des termes clos (cet à dire, les termes sans variables libres).

- Définition de la relation $e \Downarrow e'$ (on lit: e peux se réduire en e'):

  $$\texttt{Abs}\frac{}{\lambda x.e \Downarrow \lambda x.e} \qquad \texttt{App}\frac{e_1 \Downarrow \lambda x.e_0 \qquad e_0\{{}^{e_2}\!/\!_x\} \Downarrow e'}{e_1 e_2 \Downarrow e'}$$

  On écrit $e \Downarrow$ si il existe e' telle que $e \Downarrow e'$; et on écrit $e \Uparrow$ si $e \Downarrow$ n'est pas vrai.

- Sur Λ^0 on définit maintenant une notion de bisimulation.
  Une relation binaire $\mathcal{R} \subseteq \Lambda^0 \times \Lambda^0$ est une A-bisimulation, si $e_1 \mathcal{R} e_2$ implique

  1. si $e_1 \Downarrow \lambda x.e_1'$, alors il existe $\lambda z.e_2'$ telle que $e_2 \Downarrow \lambda z.e_2'$ et pour tous les $e \in \Lambda^0$, $e_1'\{{}^{e}\!/\!_x\} \mathcal{R} e_2'\{{}^{e}\!/\!_z\}$;
  2. le clause symétrique: si $e_2 \Downarrow \lambda z.e_2'$, alors il existe $\lambda x.e_1'$ telle que $e_1 \Downarrow \lambda x.e_1'$ et pour tous les $e \in \Lambda^0$, $e_1'\{{}^{e}\!/\!_x\} \mathcal{R} e_2'\{{}^{e}\!/\!_z\}$.

  On écrit $e_1 \approx e_2$ si $e_1 \mathcal{R} e_2$ pour une A-bisimulation R.

Vous pouvez faire les suppositions que: la relation identité sur Λ^0 (c'est-à-dire, la relation $\{(e, e) \;:\; e \in \Lambda^0\}$) est un A-bisimulation; $\approx$ est une relation de congruence sur Λ^0; $\approx$ est la plus grande A-bisimulation. Prouvez les assertions suivantes.

1. $(\lambda x.e)e' \approx e\{{}^{e'}\!/\!_x\}$ pour tous $(\lambda x.e)e' \in \Lambda^0$.
2. $(\lambda x.(xx))(\lambda x.(xx)) \Uparrow$.
3. Il n'est pas vrais que $\lambda x.(ex) \approx e$ pour tous $e \in \Lambda^0$.
4. $\lambda x.((\lambda y.e)x) \approx \lambda y.e$ pour tous $\lambda y.e \in \Lambda^0$.
5. Si $e \in \Lambda^0$, alors: ou bien $e \Uparrow$, ou bien il existe une abstraction $\lambda x.e'$ telle que $e \Downarrow \lambda x.e'$
6. Si $e \Uparrow$ alors $ee' \approx e$, pour tous les $e' \in \Lambda^0$.
7. En utilisant (4-6), prouvez que $\lambda x.(xx) \approx \lambda x.(x(\lambda y.xy))$.

3 Bisimulation pour la concurrence

Dans cet exercice on travaille avec un Système de Transition Etiqueté (Labeled Transition System, LTS) et la bisimilarité forte et faible (strong and weak bisimilarity), respectivement écrites $\sim$ and $\approx$. Nous vous rappelons que $\Longrightarrow$ est la clôture reflexive et transitive de $\xrightarrow{\tau}$, et que $\stackrel{\mu}{\Longrightarrow}$ est défini comme $\Longrightarrow\xrightarrow{\mu}\Longrightarrow$. Considérez la définition suivante:

Une relation $\mathcal{R}$ sur les processus du LTS est une *W-bisimulation* si $P \mathrel{\mathcal{R}} Q$ implique

1. si $P \xrightarrow{\mu} P'$, et $\mu \neq \tau$, alors il existe Q' telle que $Q \stackrel{\mu}{\Longrightarrow} Q'$ et $P' \approx \mathcal{R} \approx Q'$;
2. si $P \xrightarrow{\tau} P'$, alors il existe Q' telle que $Q \Longrightarrow Q'$ et $P' \sim \mathcal{R} \approx Q'$;
3. les clauses symétriques de (1) et (2), sur les actions de Q (c'est-à-dire, les rôles de P et Q sont inversés).

Supposons que $\mathcal{R}$ est a W-bisimulation. Prouvez les assertions suivantes.

1. Si $P \mathrel{\mathcal{R}} Q$ et $P \Longrightarrow P'$, alors il existe Q' telle que $Q \Longrightarrow Q'$ et $P' \sim \mathcal{R} \approx Q'$.
2. Si $P \approx \mathcal{R} \approx Q$ et $P \Longrightarrow P'$, alors il existe Q' telle que $Q \Longrightarrow Q'$ et $P' \approx \mathcal{R} \approx Q'$.
3. Si $P \sim \mathcal{R} \approx Q$ et $P \xrightarrow{\mu} P'$, avec $\mu \neq \tau$, alors il existe Q' telle que $Q \stackrel{\mu}{\Longrightarrow} Q'$ et $P' \approx \mathcal{R} \approx Q'$.
4. Si $P \mathrel{\mathcal{R}} Q$ et $P \stackrel{\mu}{\Longrightarrow} P'$, avec $\mu \neq \tau$, alors il existe Q' telle que $Q \stackrel{\mu}{\Longrightarrow} Q'$ et $P' \approx \mathcal{R} \approx Q'$. (Essayez d' exploiter 1, 3, et 2.)
5. $\mathcal{R} \subseteq \approx$

4 Bisimulation pour la concurrence: Solutions

1. [of Lemma 1] Induction on the length n of $P \Longrightarrow P'$. If $n = 0$, then $P = P'$; then take $Q' = Q$; the assertion is true because $\sim$ and $\approx$ are reflexive.

   For the case $n+1$, decompose $P \Longrightarrow P'$ as $P \Longrightarrow P'' \xrightarrow{\tau} P'$ where the length of $P \Longrightarrow P''$ is n. By induction, there is Q'' s.t. $Q \Longrightarrow Q''$ and $P'' \sim \mathcal{R} \approx Q''$. That is, there are P_3 and Q_3 s.t. $P'' \approx P_3 \mathcal{R} Q_3 \approx Q''$. Now, using the definition of $\sim, \approx$ and of W-bisimulation, we can find P_4 and Q_4 to complete this diagram:

$$\begin{array}{ccccccc} P'' & \approx & P_3 & \mathcal{R} & Q_3 & \approx & Q'' \\ \downarrow^{\tau} & & \downarrow^{\tau} & & \Downarrow & & \Downarrow \\ P' & \sim & P_4 & \sim\mathcal{R}\approx & Q_4 & \approx & Q' \end{array}$$

   Now conclude by transitivity of $\sim$ and $\approx$.

2. [of Lemma 2] There are P_3 and Q_3 s.t. $P \approx P_3 \mathcal{R} Q_3 \approx Q$.

   Use the definitions of $\approx$ and Lemma 1, we can find P_4 and Q_4 to complete this diagram:

$$\begin{array}{ccccccc} P & \approx & P_3 & \mathcal{R} & Q_3 & \approx & Q \\ \Downarrow & & \Downarrow & & \Downarrow & & \Downarrow \\ P' & \approx & P_4 & \sim\mathcal{R}\approx & Q_4 & \approx & Q' \end{array}$$

   Now conclude by $\sim\subseteq\approx$ and transitivity of $\approx$.

3. [of Lemma 3] There are P_3 and Q_3 s.t. $P \sim P_3 \mathcal{R} Q_3 \approx Q$.

   Use the definitions of $\sim$, $\approx$, and Lemma 1, we can find P_4 and Q_4 to complete this diagram:

$$\begin{array}{ccccccc} P & \sim & P_3 & \mathcal{R} & Q_3 & \approx & Q \\ \downarrow^{\mu} & & \downarrow^{\mu} & & \Downarrow^{\mu} & & \Downarrow^{\mu} \\ P' & \sim & P_4 & \approx\mathcal{R}\approx & Q_4 & \approx & Q' \end{array}$$

   Now conclude by $\sim\subseteq\approx$ and transitivity of $\approx$.

4. [of Lemma 4] $P \stackrel{\mu}{\Longrightarrow} P'$ can be decomposed into $P \Longrightarrow P_1 \xrightarrow{\mu} P_2 \Longrightarrow P'$, for some P_1, P_2.

   ¿From Lemmas 1, 3 and 2, and definitions of $\sim$ and $\approx$ we can find processes to fill these diagrams:

$$\begin{array}{ccccccc} P & & & \mathcal{R} & & & Q \\ \Downarrow & & & & & & \Downarrow \\ P_1 & \sim & P_3 & \mathcal{R} & Q_3 & \approx & Q_1 \\ \downarrow^{\mu} & & \downarrow^{\mu} & & \Downarrow^{\mu} & & \Downarrow^{\mu} \\ P_2 & \sim & P_4 & \approx\mathcal{R}\approx & Q_4 & \approx & Q_2 \\ \Downarrow & & \Downarrow & & \Downarrow & & \Downarrow \\ P' & \sim & P_5 & \approx\mathcal{R}\approx & Q_5 & \approx & Q' \end{array}$$

5. [of Lemma 5] The following diagrams show that $\mathcal{S} \stackrel{\text{def}}{=} \{(P, Q) \ : \ P \approx \mathcal{R} \approx Q\}$ is a weak bisimulation.

   Case $\mu \neq \tau$: we use the definitions of $\approx$ and Lemma 4:

$$\begin{array}{ccccccc} P & \approx & P_3 & \mathcal{R} & Q_3 & \approx & Q \\ \Downarrow^{\mu} & & \Downarrow^{\mu} & & \Downarrow^{\mu} & & \Downarrow^{\mu} \\ P' & \approx & P_4 & \approx\mathcal{R}\approx & Q_4 & \approx & Q' \end{array}$$

   Case $\mu \neq \tau$; we use Lemma 2:

$$\begin{array}{ccc} P & \approx\mathcal{R}\approx & Q \\ \Downarrow & & \Downarrow \\ P' & \approx\mathcal{R}\approx & Q' \end{array}$$

5 Bisimulation, équivalence observationnelle et λ-calcul: Solutions

1. $(\lambda x.e)e' \Downarrow e''$ iff $e\{e'/x\} \Downarrow e''$. Therefore $\Big\{\Big((\lambda x.e)e', e\{e'/x\}\Big)\Big\} \cup \mathcal{I}$ is a A-bisimulation.
2. By absurd, reasoning on the shortest proof of $(\lambda x.(xx))(\lambda x.(xx)) \Downarrow$
3. Take e s.t. $e \Uparrow$, for instance $(\lambda x.(xx))(\lambda x.(xx))$ Then $e \Uparrow$ whereas $\lambda x.(ex) \Downarrow \lambda x.(ex)$.
4. $\mathcal{R} \stackrel{\text{def}}{=} \Big\{\Big(\lambda x.((\lambda y.e)x), \lambda y.e\Big)\Big\} \cup \approx$ is a A-bisimulation. Since $\approx$ is a A-bisimulation, we only have to check the bisimulation diagram on the pair $(\lambda x.((\lambda y.e)x), \lambda y.e)$.

   They are both abstractions; for all $e'' \in \Lambda^0$, we have, by (1), $(\lambda y.e)e'' \approx e\{e''/y\}$.
5. By induction on the depth of the the proof of $e \Downarrow e'$, one gets that e can only converge to an abstraction.
6. By (5), e does not reduce to an abstraction. Therefore neither ee' can reduce to an abstraction: by rule `App` this is only possible if e reduces to an abstraction.

   The relation
   $$\{(e_1, e_2)\ :\ e_i \text{ does not reduce to an abstraction}\}$$
   is a A-bisimulation.
7. Take
   $$\mathcal{R} \stackrel{\text{def}}{=} \Big\{\Big(\lambda x.(xx), \lambda x.(x(\lambda y.xy))\Big)\Big\} \cup \approx$$
   This is a A-bisimulation.

   Both $\lambda x.(xx)$ and $\lambda x.(x(\lambda y.xy))$ are convergent. Take any $e \in \Lambda^0$. We show that $ee \approx e(\lambda y.ey)$.

   By (5) either $e \Uparrow$ or $e \Downarrow \lambda z.e'$, for some $\lambda z.e'$. In the former case, conclude using (6) and transitivity of $\approx$; in the latter case, conclude using (4) and the congruence of $\approx$.