Calculabilité et décidabilité : pre-examen

(*Documents autorisés : notes de cours et de TD, photocopies*)

Licence Info 3 - Michele Pagani

16 mai 2012 - Durée : 1h30

Exercice 1. Enoncer le Théorème de Rice et donner un exemple de son application aux programmes du langage `WHILE`.

Solution 1. *Toute propriété des programmes extensionelle et non-triviale n'est pas décidable. Par exemple, le sous-ensemble des arbres de $\mathbb{D}$ associés aux programmes qui calculent la fonction identité n'est pas décidable.*

Exercice 2. Ecrire une machine de Turing `M` qui réalise la fonction miroir, i.e. `M` prend en entrée une liste de bits et renvoie en sortie la liste inversée. Par exemple,

$$[\![\texttt{M}]\!](\texttt{0110011}) = \texttt{1100110}$$

La machine peut utiliser un ou plusieurs rubans, le premier ruban étant celui d'entrée *et* de sortie. Détailler le calcul de la valeur de $[\![\texttt{M}]\!](\texttt{01})$.

Solution 2. *On définit une machine* `M` *avec deux rubans, le ruban 1 comme ruban d'entrée et de sortie et le ruban 2 comme ruban de travail.*

```
1. right_1
2. if_1 B then goto 10 else 3                          % test fin mot en r1
3. if_1 0 then goto 4 else 7                            % lecture bit en r1
4. write_1 0                                              % copie bit en r2
5. left_2
6. goto 1
7. write_2 1                                              % copie bit en r2
8. left_2
9. goto 1
10. right_2                                          % deplace tete r2 toute à droite
11. if_2 B then goto 12 else 10
12. left_1                                             % copie résultat de r2 dans r1
13. left_2
14. if_2 B then goto 20 else 15
15. if_2 0 then goto 16 else 18
16. write_1 0
17. goto 12
18. write_1 1
19. goto 12
```

Les transitions de l'exécution de `M` *sur* `01` *sont les suivantes :*

compteur	*ruban 1*	*ruban 2*
1	$\underline{\texttt{B}}\texttt{01}$	$\underline{\texttt{B}}$
2	$\texttt{B}\underline{\texttt{0}}\texttt{1}$	$\underline{\texttt{B}}$
3	$\texttt{B}\underline{\texttt{0}}\texttt{1}$	$\underline{\texttt{B}}$
4	$\texttt{B}\underline{\texttt{0}}\texttt{1}$	$\underline{\texttt{B}}$
5	$\texttt{B}\underline{\texttt{0}}\texttt{1}$	$\underline{\texttt{0}}$
6	$\texttt{B}\underline{\texttt{0}}\texttt{1}$	$\underline{\texttt{B}}\texttt{0}$
1	$\texttt{B}\underline{\texttt{0}}\texttt{1}$	$\underline{\texttt{B}}\texttt{0}$
2	$\texttt{B0}\underline{\texttt{1}}$	$\underline{\texttt{B}}\texttt{0}$
3	$\texttt{B0}\underline{\texttt{1}}$	$\underline{\texttt{B}}\texttt{0}$
7	$\texttt{B0}\underline{\texttt{1}}$	$\underline{\texttt{B}}\texttt{0}$
8	$\texttt{B0}\underline{\texttt{1}}$	$\underline{\texttt{1}}\texttt{0}$
9	$\texttt{B0}\underline{\texttt{1}}$	$\underline{\texttt{B}}\texttt{10}$
1	$\texttt{B0}\underline{\texttt{1}}$	$\underline{\texttt{B}}\texttt{10}$
1	$\texttt{B01}\underline{\texttt{B}}$	$\underline{\texttt{B}}\texttt{10}$
10	$\texttt{B01}\underline{\texttt{B}}$	$\underline{\texttt{B}}\texttt{10}$
11	$\texttt{B01}\underline{\texttt{B}}$	$\texttt{B}\underline{\texttt{1}}\texttt{0}$
10	$\texttt{B01}\underline{\texttt{B}}$	$\texttt{B}\underline{\texttt{1}}\texttt{0}$
11	$\texttt{B01}\underline{\texttt{B}}$	$\texttt{B1}\underline{\texttt{0}}$
10	$\texttt{B01}\underline{\texttt{B}}$	$\texttt{B1}\underline{\texttt{0}}$
11	$\texttt{B01}\underline{\texttt{B}}$	$\texttt{B10}\underline{\texttt{B}}$
12	$\texttt{B01}\underline{\texttt{B}}$	$\texttt{B10}\underline{\texttt{B}}$
13	$\texttt{B0}\underline{\texttt{1}}\texttt{B}$	$\texttt{B10}\underline{\texttt{B}}$
14	$\texttt{B0}\underline{\texttt{1}}\texttt{B}$	$\texttt{B1}\underline{\texttt{0}}\texttt{B}$
15	$\texttt{B0}\underline{\texttt{1}}\texttt{B}$	$\texttt{B1}\underline{\texttt{0}}\texttt{B}$
16	$\texttt{B0}\underline{\texttt{1}}\texttt{B}$	$\texttt{B1}\underline{\texttt{0}}\texttt{B}$
17	$\texttt{B0}\underline{\texttt{0}}\texttt{B}$	$\texttt{B1}\underline{\texttt{0}}\texttt{B}$
12	$\texttt{B0}\underline{\texttt{0}}\texttt{B}$	$\texttt{B1}\underline{\texttt{0}}\texttt{B}$
13	$\texttt{B}\underline{\texttt{0}}\texttt{0B}$	$\texttt{B1}\underline{\texttt{0}}\texttt{B}$
14	$\texttt{B}\underline{\texttt{0}}\texttt{0B}$	$\texttt{B}\underline{\texttt{1}}\texttt{0B}$
15	$\texttt{B}\underline{\texttt{0}}\texttt{0B}$	$\texttt{B}\underline{\texttt{1}}\texttt{0B}$
18	$\texttt{B}\underline{\texttt{0}}\texttt{0B}$	$\texttt{B}\underline{\texttt{1}}\texttt{0B}$
19	$\texttt{B}\underline{\texttt{1}}\texttt{0B}$	$\texttt{B}\underline{\texttt{1}}\texttt{0B}$
12	$\texttt{B}\underline{\texttt{1}}\texttt{0B}$	$\texttt{B}\underline{\texttt{1}}\texttt{0B}$
13	$\underline{\texttt{B}}\texttt{10B}$	$\texttt{B}\underline{\texttt{1}}\texttt{0B}$
14	$\underline{\texttt{B}}\texttt{10B}$	$\underline{\texttt{B}}\texttt{10B}$

On a donc que $[\![\texttt{M}]\!](\texttt{01}) = \texttt{10}$.

Exercice 3. Étant donné un `WHILE`-programme `P` (resp. une variable `X`), on rappelle que $\underline{\texttt{P}}$ (resp. $\underline{\texttt{X}}$) dénote son encodage comme arbre binaire de $\mathbb{D}$. Prouver que la fonction suivante $f : \mathbb{D} \to \mathbb{D}_\perp$,

$$f((\underline{\texttt{P}}.\texttt{d}).\underline{\texttt{X}}) = \begin{cases} \texttt{true} & \text{si l'exécution de P sur l'éntrée d change la valeur de la variable X,} \\ \texttt{false} & \text{sinon.} \end{cases}$$

n'est pas calculable. (*Suggestion : par réduction au problème de l'arrêt. On pourra supposer l'existence d'un programme* `L` *qui prend en entrée l'arbre* $\underline{\texttt{P}}$ *associé à un programme* `P` *et donne en sortie l'arbre* $\underline{V}$ *associé à une variable qui n'apparaisse pas dans* `P`.)

Solution 3. *Supposons par contradiction que f soit calculable par un programme* `F`. *On construira un*

programme H *décidant le problème de l'arrêt, i.e. :*

$$[\![\mathtt{H}]\!](\underline{\mathtt{P}}.\mathtt{d}) = \begin{cases} \mathtt{true} & \textit{si } [\![\mathtt{P}]\!](\mathtt{d}) \downarrow, \\ \mathtt{false} & \textit{sinon.} \end{cases}$$

Ceci étant en contradiction avec le Théorème de l'arrêt, on conclue que f n'est pas calculable.
Soit L *le programme qui prend en entrée l'arbre* $\underline{\mathtt{P}}$ *associé à un programme* P *et donne en sortie l'arbre* $\underline{\mathtt{V}_\ell}$ *associé à une variable qui n'apparaisse pas dans* P*. A partir de* L*, on peut construire un programme* M *qui prend en entrée l'arbre* $\underline{\mathtt{P}} = (\underline{\mathtt{V}_i}\ \underline{\mathtt{C}}\ \underline{\mathtt{V}_j})$ *associé à un programme* P *et qui donne en sortie l'arbre* $(\underline{\mathtt{V}_i}\ \underline{\mathtt{C}; \mathtt{V}_\ell := \mathtt{true}}\ \underline{\mathtt{V}_j})$ *d'un programme* Q *obtenu en ajoutant l'affectation* $\mathtt{V}_\ell := \mathtt{true}$ *à la fin des commandes de* P*, où* $\underline{\mathtt{V}_\ell} = \mathtt{L}(\underline{\mathtt{P}})$*. Par exemple,* M *peut être écrit comme suit :*

```
read X;
 Xin := hd X;
 Xc := hd (hd X);
 Xout := hd (hd (hd X));
 Z := L X;
 Xc := list ; Xc (list := Z true);
 Y := list Xin Xc Xout;
write Y
```

où list *est la macro prenant en éntrée une suite d'expressions* $\mathtt{E}_1, \ldots, \mathtt{E}_n$ *et donnant en sortie la liste* $(\mathtt{E}_1 \ldots \mathtt{E}_n)$ *de ces expressions.*
On remarque que $[\![\mathtt{P}]\!](\mathtt{d}) \downarrow$ *ssi l'exécution de* Q *sur l'éntrée* d *change la valeur de la variable* $\mathtt{V}_\ell$ *de* nil *à* true*.*
Maintenant, le programme H *décidant le problème de l'arrêt peut être défini comme suit :*

```
read X;
 Xp := hd X;
 Xd := tl X;
 Xx := L Xp;
 Xq := M Xp;
 Y := F cons (cons Xq Xd) Xx;
write Y
```

On a que $[\![\mathtt{H}]\!](\mathtt{P}, \mathtt{d}) = \mathtt{true}$ *ssi* $[\![\mathtt{F}]\!]((\mathtt{Q}, \mathtt{d}).\mathtt{V}_\ell) = \mathtt{true}$ *ssi* $[\![\mathtt{P}]\!](\mathtt{d}) \downarrow$*, et que* $[\![\mathtt{H}]\!](\mathtt{P}, \mathtt{d}) = \mathtt{false}$ *ssi* $[\![\mathtt{F}]\!]((\mathtt{Q}, \mathtt{d}).\mathtt{V}_\ell) = \mathtt{false}$ *ssi* $[\![\mathtt{P}]\!](\mathtt{d}) \uparrow$*. On conclue que le programme* H *décide le problème de l'arrêt. Ceci est en contradiction avec le Théorème de l'arrêt, donc le programme* F *qu'on a supposé calculer la fonction f ne peut pas exister.*

Exercice 4. Étant donnés deux fonctions calculables $f, g : \mathbb{D} \to \mathbb{D}_\perp$, prouver que leur composée $f \circ g$ est aussi calculable. En plus, étant P un programme calculant $f \circ g$, définir le sous-ensemble des éléments de $\mathbb{D}$ où P termine, à partir des domaines de définition de f et g.

Solution 4. *Soit F (resp. G) le programme qui calcule f (resp. g). La fonction $f \circ g$ est calculée par le programme* P *suivant :*

```
read X;
X := G X;
X := F X;
write X
```

Pour tout arbre binaire $\mathtt{d} \in \mathbb{D}$*, on a que* $[\![\mathtt{P}]\!](\mathtt{d}) \downarrow$ *ssi* $\mathtt{d} \in \mathrm{Dom}(g)$ *et* $g(\mathtt{d}) \in \mathrm{Dom}(f)$*, où* $\mathrm{Dom}(f)$ *(resp.* $\mathrm{Dom}(g)$*) est le domaine de définition de f (resp. g).*

Exercice 5. Traduire dans le langage CM, l'instruction `X0 := hd X0` du langage GOTO, etant donné l'encodage $c : \mathbb{D} \mapsto \mathbb{N}$ des arbres binaires dans les nombres entiers positifs comme suit :

$$c(\texttt{nil}) = 0$$
$$c((\texttt{d.e})) = 2^{c(\texttt{d})} \cdot 3^{c(\texttt{e})}$$

On pourra utiliser la macro `Ri := DIV2 Rj`, qui affecte au registre `Ri` la moitié de la valeur contenue dans le registre `Rj`, si celle-ci est un nombre pair, sinon 0.

Solution 5. *L'instruction* `X0 := hd X0` *est simulée par le* CM *programme suivant,* `R0` *étant le registre d'entrée et de sortie.*

```
1. if R0=0 goto 5 else 2
2. R1 := R1+1
3. R0 := DIV2 R0
4. goto 1
5. R0 := R1-1
```

Le lecteur pourra noter que sur une entrée $2^n \cdot 3^k$*, le programme sort de la boucle des lignes 1-4 après avoir répété* $n + 1$ *fois l'instruction* `R0 := DIV2R0` *. Ceci explique la soustraction de la ligne 5.*