DM Programmation I (2016-17)

Nous allons revenir au langage jouet Imp du cours :

Commandes				Expressions			
c	::=	$x := e$	affectation	e	::=	x	variables
	|	`skip`	ne rien faire		|	$\dot{n}$	constante entière ($n \in \mathbb{Z}$)
	|	$c_1; c_2$	séquence		|	$e \dot{+} e$	addition
	|	`if` e `then` c_1 `else` c_2	conditionnelle		|	$\dot{-} e$	opposé
	|	`while` e `do` c	boucle while				

Les variables sont supposées en nombre fini, et seront numérotées x_1, x_2, …, x_n.

On cherche à déterminer des intervalles de variation possibles des variables à travers l'exécution d'un programme. Pour ceci, on va considérer une sémantique dénotationnelle « quotient ».

On note $\mathcal{I}$ l'ensemble contenant l'élément spécial $\bot$, plus tous les couples (a, b), où $a, b \in \mathbb{Z} \cup \{-\infty, +\infty\}$, et $a + 1 < b$. On peut penser au couple (a, b) comme à une notation représentant l'intervalle d'entiers $]a, b[$ $(= \{c \in \mathbb{Z} \mid a < c < b\})$, et à $\bot$ comme à un symbole qui signifie l'intervalle vide.

On ordonne $\mathcal{I}$ par $v \leq w$ ssi :

— $v = \bot$,

— ou $v = (a, b)$, $w = (c, d)$, et $a \geq c$, $b \leq d$.

Ce n'est pas un ordre total ! Tout raisonnement de la forme « montrons que $u \leq v$; si on avait $u > v$, alors … contradiction » est donc faux. Dans la suite, ne confondez pas non plus « plus grand élément » (le plus grand élément de A s'il existe un est $v \in A$ tel que pour tout $u \in A$, $u \leq v$) et « élément maximal » (un élément maximal de A est un $v \in A$ tel que pour tout $u > v$, u n'est pas dans A). Ne confondez pas non plus ces notions avec « borne supérieure » : la borne supérieure de A, si elle existe, est le plus petit des majorants. Un majorant de A est un élément v, pas nécessairement dans A, tel que tout $u \in A$ est $\leq v$. La borne supérieure w de A, si elle existe, est le plus petit élément de l'ensemble des majorants : w est un majorant de A, et pour tout majorant v de A, $w \leq v$.

Définissons l'addition dans $\mathcal{I}$ par :

— $\bot + v = v + \bot = \bot$ pour tout $v \in \mathcal{I}$;

— $(a, b) + (c, d) = (a + c, b + d)$ (on conviendra que $(-\infty) + c = -\infty$ et $b + (+\infty) = (+\infty)$; noter que l'on n'aura jamais à calculer l'expression absurde $(-\infty) + (+\infty)$, au vu des contraintes de formation des éléments de $\mathcal{I}$).

Similairement, définissons l'opposé dans $\mathcal{I}$ par : $-\bot = \bot$, $-(a, b) = (-b, -a)$ sinon.

1. $\mathcal{I}$ est un treillis complet, autrement dit toute famille d'éléments a une borne supérieure et une borne inférieure, comme il est facile de le voir. Comment est définie l'opération $\vee$, borne supérieure de deux éléments ? (Oui, $v \vee w$ est le plus petit des majorants de $\{v, w\}$. Je veux une définition explicite, avec analyse des différents cas possibles.)
2. $\mathcal{I}^n$, l'ensemble des n-uplets d'éléments de $\mathcal{I}$, avec l'ordre composante par composante, est-il : (a) un dcpo ? (b) un treillis complet ?

On identifiera $\mathcal{I}^n$ à l'espace des fonctions de l'ensemble des variables vers $\mathcal{I}$, c'est-à-dire aux environnements « quotients ». Pour $\eta \in \mathcal{I}^n$, $\eta(x_i)$ sera donc la $i$ème composante du n-uplet η.

On étend la notation $\vee$ aux environnements par : $\eta \vee \eta'$ envoie toute variable x vers $\eta(x) \vee \eta'(x)$.

3. Une fonction monotone $F \colon \mathcal{I}^n \to \mathcal{I}^n$ est dite *inflationnaire* si et seulement si $\eta \leq F(\eta)$ pour tout η. Pour toute fonction inflationnaire F, pour tout $\eta_0 \in \mathcal{I}^n$, montrer que F a plus petit point fixe $\geq \eta_0$. A titre d'indication, considérez la fonction F' définie par $F'(\eta) = \eta_0 \vee F(\eta)$.

   On notera dans la suite $\mathrm{lfp}_\eta(F)$ ce plus petit point fixe de F au-dessus de η.

La sémantique quotient $Q[\![e]\!]$ des expressions e prend un environnement quotient η, et retourne une valeur dans $\mathcal{I}$, selon les clauses :

$$\begin{aligned} Q[\![x]\!]\eta &= \eta(x) \\ Q[\![\dot{n}]\!]\eta &= (n-1, n+1) \\ Q[\![e_1 \dot{+} e_2]\!]\eta &= Q[\![e_1]\!]\eta + Q[\![e_2]\!]\eta \\ Q[\![\dot{-} e]\!]\eta &= -Q[\![e]\!]\eta \end{aligned}$$

La sémantique quotient $Q[\![c]\!]$ des commandes c prend un environnement quotient η, et retourne un nouvel environnement quotient, selon les règles :

$$Q[\![x := e]\!]\eta = \eta \vee \eta[x \mapsto Q[\![e]\!]\eta]) \quad (1)$$

$$Q[\![\texttt{skip}]\!]\eta = \eta \quad (2)$$

$$Q[\![c_1; c_2]\!]\eta = Q[\![c_2]\!](Q[\![c_1]\!]\eta) \quad (3)$$

$$Q[\![\texttt{if}\ e\ \texttt{then}\ c_1\ \texttt{else}\ c_2]\!]\eta = \begin{cases} \eta & \text{si } Q[\![e]\!]\eta = \bot \\ Q[\![c_2]\!]\eta & \text{si } Q[\![e]\!]\eta = (-1, 1) \\ Q[\![c_1]\!]\eta & \text{si } Q[\![e]\!]\eta = (a, b) \text{ avec } a \geq 0 \text{ ou } b \leq 0 \\ Q[\![c_1]\!]\eta \vee Q[\![c_2]\!]\eta & \text{sinon} \end{cases} \quad (4)$$

$$Q[\![\texttt{while}\ e\ \texttt{do}\ c]\!]\eta = \mathrm{lfp}_\eta(F_{e,c}) \quad (5)$$

où $F_{e,c} \colon \mathcal{I}^n \to \mathcal{I}^n$ est la fonction suivante :

$$F_{e,c}(\eta') = \begin{cases} \eta' & \text{si } Q[\![e]\!]\eta' = \bot \text{ ou } Q[\![e]\!]\eta' = (-1, 1) \\ \eta' \vee Q[\![c]\!]\eta' & \text{sinon.} \end{cases}$$

On souhaite montrer que cette définition est sensée.

4. En supposant que $Q[\![e]\!]$ et $Q[\![c]\!]$ sont déjà définies et monotones en leur argument η, montrez que $F_{e,c}$ est inflationnaire de $\mathcal{I}^n$ vers $\mathcal{I}^n$. N'oubliez pas de démontrer la monotonie d'abord.

$$\frac{}{(x := e, \rho) \to \rho[x \mapsto [\![e]\!]\rho]}\ (\to :=) \qquad \frac{}{(\texttt{skip}, \rho) \to \rho}\ (\to \texttt{skip})$$

$$\frac{(c_1, \rho) \to \rho'}{(c_1; c_2, \rho) \to (c_2, \rho')}\ (\to Seq_{fin}) \qquad \frac{(c_1, \rho) \to (c_1', \rho')}{(c_1; c_2, \rho) \to (c_1'; c_2, \rho')}\ (\to Seq)$$

$$\frac{}{(\texttt{if}\ e\ \texttt{then}\ c_1\ \texttt{else}\ c_2, \rho) \to (c_1, \rho)}\ (\to \texttt{if}\ 1) \text{ si } [\![e]\!]\rho \neq 0 \qquad \frac{}{(\texttt{if}\ e\ \texttt{then}\ c_1\ \texttt{else}\ c_2, \rho) \to (c_2, \rho)}\ (\to \texttt{if}\ 2) \text{ si } [\![e]\!]\rho = 0$$

$$\frac{}{(\texttt{while}\ e\ \texttt{do}\ c, \rho) \to (c; \texttt{while}\ e\ \texttt{do}\ c, \rho)}\ (\to \texttt{while}) \text{ si } [\![e]\!]\rho \neq 0 \qquad \frac{}{(\texttt{while}\ e\ \texttt{do}\ c, \rho) \to \rho}\ (\to \texttt{while}_{fin}) \text{ si } [\![e]\!]\rho = 0$$

FIGURE 1 – Une sémantique opérationnelle à petits pas de IMP

5. La question 3 permet donc d'en conclure que $Q[\![\texttt{while}\ e\ \texttt{do}\ c]\!]$ est bien définie. Pourquoi ceci définit-il bien une fonction monotone de $\mathcal{I}^n$ dans $\mathcal{I}^n$? Autrement dit, supposons $\eta \leq \eta'$, alors pourquoi a-t-on $\mathrm{lfp}_\eta(F_{e,c}) \leq \mathrm{lfp}_{\eta'}(F_{e,c})$? On montrera, plus généralement, que si F est une fonction inflationnaire d'un treillis complet L dans lui-même (par exemple $F_{e,c}$, mais pas uniquement), et si $\eta \leq \eta'$, alors $\mathrm{lfp}_\eta(F) \leq \mathrm{lfp}_{\eta'}(F)$.

A partir de ces considérations, on peut démontrer que $Q[\![c]\!]$ est bien définie pour toute commande c. On peut aussi démontrer que c'est une fonction Scott-continue de $\mathcal{I}^n$ vers $\mathcal{I}^n$. Parmi ce qu'il faut démontrer dans ce but, on trouver les trois questions suivantes.

6. Montrer que la fonction $+ : \mathcal{I} \times \mathcal{I} \to \mathcal{I}$ est Scott-continue. En clair, vous devrez montrer que : (a) $+$ est monotone, (b) pour toute famille dirigée $(v_i, w_i)_{i \in I}$ de couples d'éléments de $\mathcal{I}$, $\sup_{i \in I}(v_i + w_i) = (\sup_{i \in I} v_i) + (\sup_{i \in I} w_i)$. On pourra utiliser sans démonstration les résultats suivants caractérisant les bornes supérieures de familles dirigées $(v_i)_{i \in I}$ dans $\mathcal{I}$: si tous les v_i sont égaux à $\bot$, alors $\sup_{i \in I} v_i = \bot$; sinon, écrivons v_i sous la forme (a_i, b_i) pour tout $i \in I$ tel que $v_i \neq \bot$, alors $\sup_{i \in I} v_i = (\inf_{i \in I, v_i \neq \bot} a_i, \sup_{i \in I, v_i \neq \bot} b_i)$, où les infs et les sups dans la dernière expression sont pris dans $\mathbb{Z} \cup \{-\infty, +\infty\}$ muni de son ordre usuel ($-\infty \leq \cdots \leq -3 \leq -2 \leq -1 \leq 0 \leq 1 \leq 2 \leq 3 \leq \cdots \leq +\infty$).
7. Montrer que la fonction $- : \mathcal{I} \to \mathcal{I}$ est Scott-continue.
8. Montrer que, si F est une fonction inflationnaire et Scott-continue de $\mathcal{I}$ dans $\mathcal{I}$, alors la fonction qui à $\eta \in \mathcal{I}$ associe $\mathrm{lfp}_\eta(F)$ est encore Scott-continue.

On peut en déduire que $Q[\![c]\!]$ est Scott-continue, par récurrence sur la taille de c, en faisant une récurrence auxiliaire pour démontrer que $Q[\![e]\!]$ est Scott-continue pour toute expression e. Le cas où e est une addition est traité par la question 6, le cas où c'est un opposé par la question 7. Le cas où c est une boucle while est traité par la question 8. Nous ne demandons pas de faire la démonstration complète, et admettrons dans la suite que $Q[\![c]\!]$ est Scott-continue.

Nous utilisons désormais une sémantique opérationnelle à petits pas de IMP—la première des notes de cours. Les règles sont en figure 1 ; ρ y dénote un environnement (réel).

Pour toute commande c, et tout ensemble $\mathcal{E}$ d'environnements (réels), disons que l'environnement ρ' est *accessible* depuis c et $\mathcal{E}$ si et seulement s'il existe un environnement (réel) ρ, dans $\mathcal{E}$, tel que $(c, \rho) \to^* \rho'$ ou bien $(c, \rho) \to^* (c', \rho')$ pour une certaine commande c'. On

notera $X[\![c]\!]\mathcal{E}$ l'ensemble des environnements accessibles depuis c et $\mathcal{E}$. On a les relations :

$$\begin{aligned}
X[\![x := e]\!]\mathcal{E} &= \mathcal{E} \cup \{\rho[x \mapsto [\![e]\!]\rho] \mid \rho \in \mathcal{E}\} && (6)\\
X[\![\texttt{skip}]\!]\mathcal{E} &= \mathcal{E} && (7)\\
X[\![c_1; c_2]\!]\mathcal{E} &= X[\![c_2]\!](X[\![c_1]\!]\mathcal{E}) && (8)\\
X[\![\texttt{if}\ e\ \texttt{then}\ c_1\ \texttt{else}\ c_2]\!]\mathcal{E} &= \mathcal{E} \cup X[\![c_2]\!]\{\rho \in \mathcal{E} \mid [\![e]\!]\rho = 0\} && (9)\\
&\quad \cup X[\![c_1]\!]\{\rho \in \mathcal{E} \mid [\![e]\!]\rho \neq 0\} &&\\
X[\![\texttt{while}\ e\ \texttt{do}\ c]\!]\mathcal{E} &= \mathrm{lfp}_{\mathcal{E}}(\Phi_{e,c}) && (10)
\end{aligned}$$

où $\Phi_{e,c}$ est la fonction définie par :

$$\Phi_{e,c}(\mathcal{E}') = \mathcal{E}' \cup X[\![c]\!]\{\rho \in \mathcal{E}' \mid [\![e]\!]\rho \neq 0\}$$

et l'on rappelle que $\mathrm{lfp}_{\mathcal{E}}$ désigne l'opérateur plus petit point fixe au-dessus de $\mathcal{E}$. Ici, $\mathcal{E}$, $\mathcal{E}'$ appartiennent au treillis complet dcpo $\mathbb{P}(Env)$ des ensembles d'environnements (réels), ordonné par inclusion $\subseteq$.

On ne demandera pas de démontrer ces égalités… sauf la dernière. C'est le sujet des deux questions qui viennent.

9. Montrer que la fonction $filt\colon \mathcal{E}' \mapsto \{\rho \in \mathcal{E}' \mid [\![e]\!]\rho \neq 0\}$ est Scott-continue. On pourra utiliser sans preuve que le supremum $\sup_{i\in I} \mathcal{E}_i$ dans $\mathbb{P}(Env)$ est l'union $\bigcup_{i\in I} \mathcal{E}_i$.

Au vu de la définition de $X[\![c]\!]\mathcal{E}$ comme ensemble de traces accessibles, il n'est pas trop difficile de démontrer que $X[\![c]\!]$ est Scott-continue. Comme toute composée de fonctions Scott-continues est Scott-continue, et que la fonction $\cup$ est trivialement Scott-continue, on en déduit que $\Phi_{e,c}$ est elle aussi Scott-continue. Je n'en demande pas de démonstration plus détaillée.

10. On admet qu'on peut démontrer que $X[\![\texttt{while}\ e\ \texttt{do}\ c]\!]\mathcal{E}$ est égal à $\mathcal{E}_0 \cup \mathcal{E}_1 \cup \cdots \cup \mathcal{E}_n \cup \cdots$, où la suite $\mathcal{E}_n$ est définie par :

$$\begin{aligned}
\mathcal{E}_0 &= \mathcal{E}\\
\mathcal{E}_{n+1} &= \mathcal{E}_n \cup X[\![c]\!]\{\rho \in \mathcal{E}_n \mid [\![e]\!]\rho \neq 0\}
\end{aligned}$$

A partir de cette observation, démontrer l'égalité (10).

Nous pouvons désormais admettre la validité des équations (6) à (10).

On définit la fonction γ de $\mathcal{I}$ vers $\mathbb{P}(\mathbb{Z})$: $\gamma(\bot) = \emptyset$, $\gamma((a, b))$ est l'intervalle entier $]a, b[$ (ouvert, et non vide).

Réciproquement, pour toute partie E de $\mathbb{Z}$, on définit $\alpha(E) \in \mathcal{I}$ par : $\alpha(\emptyset) = \bot$, et si E est non vide, alors $\alpha(E) = (\inf E - 1, \sup E + 1)$. (Les infs et sups sont de nouveau calculés dans $\mathbb{Z} \cup \{-\infty, +\infty\}$, et $(-\infty) - 1 = (-\infty)$, $(+\infty) + 1 = +\infty$.)

On vérifie aisément que α et γ sont deux fonctions monotones, lorsque $\mathbb{P}(\mathbb{Z})$ est ordonné par l'ordre d'inclusion $\subseteq$. On peut aussi vérifier que $\alpha(\gamma(v)) = v$ pour tout $v \in \mathcal{I}$ et que $E \subseteq \gamma(\alpha(E))$ pour toute partie E de $\mathbb{Z}$.

11. En déduire que, pour tout $v \in \mathcal{I}$, pour tout $E \in \mathbb{P}(\mathbb{Z})$, $\alpha(E) \leq v$ ssi $E \subseteq \gamma(v)$.
    Si l'une de ces inégalités équivalentes est vérifiée, on dira que v *représente correctement* l'ensemble E.

On généralise cela aux environnements : un environnement quotient η représente correctement un ensemble $\mathcal{E}$ d'environnements (réels) si et seulement si $\eta(x)$ représente correctement $\{\rho(x) \mid \rho \in \mathcal{E}\}$ pour toute variable x.

On admettra que si η représente correctement $\mathcal{E}$, alors pour toute expression e, $Q[\![e]\!]\eta$ représente correctement $\{[\![e]\!]\rho \mid \rho \in \mathcal{E}\}$.

12. Soit c une commande, et supposons que pour tout environnement quotient η' et tout ensemble $\mathcal{E}'$ d'environnements (réels) tels que η' représente correctement $\mathcal{E}'$, $Q[\![c]\!]\eta'$ représente correctement $X[\![c]\!]\mathcal{E}'$.

    Montrer que, pour tous η et $\mathcal{E}$, si η représente correctement $\mathcal{E}$, alors $Q[\![\texttt{while}\, e\, \texttt{do}\, c]\!]\eta$ représente correctement $X[\![\texttt{while}\, e\, \texttt{do}\, c]\!]\mathcal{E}$.

13. La similarité entre la définition de $Q[\![c]\!]\rho$ est les équations caractérisant $X[\![c]\!]\mathcal{E}$ nous permettent de démontrer que pour tous η et $\mathcal{E}$, si η représente correctement $\mathcal{E}$, alors $Q[\![c]\!]\eta$ représente correctement $X[\![c]\!]\mathcal{E}$. Nous avons déjà traité le cas de la boucle `while` à la question 12. Cette démonstration s'effectue par récurrence, mais sur quoi ?

14. Poursuivons la question précédente. Le lemme à prouver dans le cas de la boucle `while` a été décrit à la question 12. Dans le cas de l'affectation, quel lemme devons-nous prouver ?

15. Prouvez ce lemme.

Les cas des autres constructions du langages ne seront pas traitées ici. Tout fonctionne : si η représente correctement $\mathcal{E}$, alors $Q[\![c]\!]\eta$ représente correctement $X[\![c]\!]\mathcal{E}$.

16. On peut penser calculer $Q[\![c]\!]\eta$ par récurrence sur la structure de c, en suivant les équations (1) à (5). Pour cette dernière, on peut calculer $\mathrm{lfp}_\eta(F_{e,c})$ par *itérations de Kleene* : comme la borne supérieure de η, $F_{e,c}(\eta)$, $F^2_{e,c}(\eta)$, etc., en calculant $\eta_0 = \eta$, $\eta_{n+1} = F_{e,c}(\eta_n)$, et en s'arrêtant à la première étape n où $\eta_{n+1} = \eta_n$. Montrer que ceci ne termine pas, en exhibant un programme $\texttt{while}\, e\, \texttt{do}\, c$ adéquat.

17. Pour corriger ce problème, je propose de remplacer $F_{e,c}$ dans (5) par $F'_{e,c}$ (autrement dit je remplace la définition de $Q[\![\texttt{while}\, e\, \texttt{do}\, c]\!]\eta$ par $\mathrm{lfp}_\eta(F'_{e,c})$), définie comme suit :

$$F'_{e,c}(\eta') \quad = \quad \begin{cases} \eta' & \text{si } Q[\![e]\!]\eta' = \bot \text{ ou } Q[\![e]\!]\eta' = 0 \\ \eta' \nabla Q[\![c]\!]\eta' & \text{sinon.} \end{cases} ,$$

    où l'opération ∇ est définie par : $\bot \nabla v = v$, $v \nabla \bot = v$, $(a,b)\nabla(c,d)$ est égal à (a',b') où $a' = a$ si $c \geq a$, $a' = -\infty$ sinon, et $b' = b$ si $d \leq b$, $b' = +\infty$ sinon.

    Pourquoi le calcul de $\mathrm{lfp}_\eta(F'_{e,c})$ par itérations de Kleene termine-t-il ? Pourquoi, avec la nouvelle définition, $Q[\![c]\!]\eta$ représente-t-il toujours correctement $X[\![c]\!]\mathcal{E}$ pour toute commande c ?

C'est un exemple simple d'*interprétation abstraite*, une technique d'analyse statique de programmes, donnant des informations sur les valeurs calculées par le programme sans avoir à l'exécuter.