Programmation 1 – TP n° 5
Sémantiques Opérationnelles

Remarque. Les questions marquées d'une étoile (*) peuvent être *difficiles*, et sont donc à faire à la fin du TD.

1 Introduction

Théorème (Knaster-Tarski). Si $(\mathcal{T}, \sqsubseteq)$ est un treillis complet et $f : \mathcal{T} \to \mathcal{T}$ est croissante alors il existe $m \in \mathcal{T}$ tel que m soit le plus petit pré-point fixe de f.

Q 1.1 Définir de manière formelle les expressions *croissante*, *plus petit* et *pré-point fixe*.

Lemme. Si X est un ensemble, $\mathcal{P}(X)$ muni de $\subseteq$ est un treillis complet.

On définit inductivement un ensemble $\mathbb{N}$ à partir de la signature $\Sigma = \{Zero_0, Succ_1\}$ et des règles :

$$\frac{}{Zero \in \mathbb{N}} \qquad \frac{n \in \mathbb{N}}{Succ(n) \in \mathbb{N}}$$

Plus précisément, l'ensemble $\mathbb{N}$ est donc défini comme le plus petit pré-point fixe de la fonction $F : X \mapsto \{Zero\} \cup \{Succ(x) \mid \forall x \in X\}$.

Q 1.2 Montrez que F est croissante.

On définit l'ensemble $Plus \subseteq \mathbb{N} \times \mathbb{N} \times \mathbb{N}$ de façon inductive par les règles suivantes :

$$\frac{n \in \mathbb{N}}{(Zero, n, n) \in Plus} \qquad \frac{(n, p, q) \in Plus}{(Succ(n), p, Succ(q)) \in Plus}$$

Soit Y l'ensemble $\{x \in \mathbb{N} \mid (x, Zero, x) \in Plus\}$.

Q 1.3 Est-ce que $F(Y) \subseteq Y$? Que cela implique-t-il ?
Que venez vous de démontrer ?

Enfin, on définit un sous-ensemble Inf de $\mathbb{N} \times \mathbb{N}$ par :

$$\frac{n \in \mathbb{N}}{(Zero, n) \in Inf} \qquad \frac{(n, p) \in Inf}{(Succ(n), Succ(p)) \in Inf}$$

Q 1.4 Démontrez que pour tous entiers positifs a, b et c, si $a + b = c$ alors $a \leqslant c$.

2 Sémantiques Opérationnelles

Voici la grammaire de IMP :
- registres $x, y, \ldots$
- expressions arithmétiques
  $a \quad ::= \quad 0, 1, 2, \ldots \mid a + a \mid a * a \mid a - a \mid a/a \mid x$
- expressions booléennes
  $b \quad ::= \quad \mathsf{true} \mid \mathsf{false} \mid b \vee b \mid \neg b \mid a > a \mid a = a$
- commandes (programmes)
  $c \quad ::= \quad \texttt{skip} \mid \texttt{if}\ b\ \texttt{then}\ c\ \texttt{else}\ c \mid x{:=}a \mid c; c \mid \texttt{while}\ b\ \texttt{do}\ c$

On définit les valeurs de IMP :
- valeurs arithmétiques
  $v_a \quad ::= \quad 0, 1, 2, \ldots$
- valeurs booléennes
  $v_b \quad ::= \quad \mathsf{true} \mid \mathsf{false}$

IMP est donc un langage impératif.

Définitions
- On définit un état mémoire (dénoté par $\sigma, \sigma', \ldots$) comme une fonction totale des registres dans les valeurs arithmétiques.
  - si σ est un état mémoire, x un registre et v une valeur arithmétique, $\sigma[x \mapsto v]$ dénote l'état mémoire obtenu en affectant v à x dans l'état mémoire σ
  - **init** est l'état mémoire initial, qui associe l'expression 0 à chaque registre.
  - On utilisera la notation $[x_1 \mapsto v_1, \ldots, x_n \mapsto v_n]$ pour $\mathbf{init}[x \mapsto v_1] \ldots [x \mapsto v_n]$
- Si e est une expression arithmétique et σ un état mémoire, on définit $[\![e]\!]^E_\sigma$ comme l'évaluation de l'expression e dans l'état mémoire σ.
- Si b est une expression booléenne et σ un état mémoire, on définit $[\![b]\!]^B_\sigma$ comme l'évaluation de l'expression booléenne b dans l'état mémoire σ.
- On pourrait donner des règles pour ces deux fonctions, on va rester pudique cette fois en considèrant qu'elle "font ce qu'on veut".

2.1 Sémantique Opérationnelle à grands pas (naturelle)

Pour écrire la sémantique opérationnelle à grands pas on définit le prédicat $(c, \sigma) \Rightarrow \sigma'$ qui signifie que l'éxécution de la commande c dans l'état mémoire σ aboutit à l'état mémoire σ'.

Q 2.1 Donner les règles de sémantique naturelle pour IMP.

Q 2.2 Prouver :

$$(t := x; x := y; y := t, [x \mapsto 1, y \mapsto 2]) \Rightarrow [x \mapsto 2, y \mapsto 1, t \mapsto 1]$$

$$(f := 1; \texttt{while}\ x > 0\ \texttt{do}\ (f := f * x; x := x - 1), [x \mapsto 4]) \Rightarrow [x \mapsto 0, f \mapsto 13]$$

Q 2.3 (*) Enoncer, puis prouver le déterminimse de la sémantique opérationnelle à grands pas.

Q 2.4 (*) Montrer l'associativité du séquencement.

2.2 Sémantique Opérationnelle à petits pas (SOS)

Pour écrire la sémantique opérationnelle à petits pas on définit le prédicat $(c, \sigma) \to (c', \sigma')$ qui signifie qu'un pas d'exécution de la commande c dans l'état mémoire σ aboutit à la commande c' et l'état mémoire σ'.

On utilise $\bullet$ pour dénoter la fin d'exécution.

Q 2.5 Donner les règles de la sémantique opérationnelle à petits pas de IMP.

Q 2.6 Prouver :

$$(t := x; x := y; y := t, [x \mapsto 1, y \mapsto 2]) \to^\star (\bullet, [x \mapsto 2, y \mapsto 1, t \mapsto 1])$$

$$(f := 1; \texttt{while}\ x > 0\ \texttt{do}\ (f := f * x; x := x - 1), [x \mapsto 4]) \to^\star (\bullet, [x \mapsto 0, f \mapsto 28])$$

Q 2.7 (*) Enoncer, puis prouver le déterminimse de la sémantique opérationnelle à petits pas.

Q 2.8 (*) Montrer l'associativité du séquencement.

2.3 Equivalence de programmes

Q 2.9 Définir une équivalence opérationnelle de programmes en utilisant la sémantique naturelle. Définir pour cela une fonction des programmes vers les relations partielles entre états mémoires.

Q 2.10 Proposer une équivalence opérationnelle en petits pas. Discuter sur les programmes équivalents pour une sémantique et pas pour l'autre.

2.4 Finalement

Q 2.11 Enoncer l'équivalence entre les deux sémantiques opérationnelles d'IMP.

Q 2.12 Prouver cette équivalence.