Univ. Grenoble Alpes **UFRIMAG**
Master 1 **Année 2016-2017**

SLPC

Devoir surveillé du mercredi 9 novembre 2016

Consignes et informations :
- Durée : 1 heure 15 minutes
- Tous les documents du cours et du TD sont autorisés.
- Tous les dispositifs électroniques sont interdits.
- Les exercices sont **indépendants**.
- Le barème est donné à titre indicatif.

Partie I Sémantique opérationnelle (10 points)

On considère la syntaxe des expressions arithmétiques vue en cours en l'étendant avec deux nouveaux opérateurs : un opérateur de *pré-incrémentation* et un opérateur de *post-incrémentation*. La nouvelle syntaxe des expressions arithmétiques est la suivante :

$$a \quad ::= \quad n \mid x \mid a_1 + a_2 \mid ++x \mid x++$$

On note **Aexp'** l'ensemble de ces expressions arithmétiques. Informellement, la valeur de l'expression $x++$ est la valeur de la variable x, puis on associe à x l'ancienne valeur augmentée de 1. De la même manière, la valeur de l'expression $++x$ est obtenue en augmentant de 1 la valeur associée à x, puis retournant la nouvelle valeur. Par exemple, dans un état où x vaut 3, $x++$ vaut 3 et la nouvelle valeur de x est 4 tandis que $++x$ vaut 4 et la nouvelle valeur de x est 4.

On constate que l'évaluation des expressions peut modifier l'état.

Les configurations pour les expressions arithmétiques seront données par $(\textbf{Aexp'} \times \textbf{State}) \cup (\mathbb{Z} \times \textbf{State})$ (on code la valeur de l'expression, et l'état obtenu par effet de bord). La forme des règles est la suivante : $(a, \sigma) \longrightarrow (v, \sigma)$

Ne pas utiliser la fonction $\mathcal{A}$ vue en cours

Exercice 1

1. Donner la sémantique opérationnelle des expressions arithmétiques en supposant que les opérandes d'un opérateur binaire sont évalués de gauche à droite.

2. Que vaut l'expression $x + (x++)$ où x vaut initialement 3 ? Vérifier sur cet exemple la cohérence de vos règles.

3. On modifie les commandes en considérant la syntaxe suivante :

   $$S \quad := \quad x := a \mid x +:= a \mid S_1; S_2$$

   Informellement, pour évaluer la commande $x +:= a$ dans la mémoire σ,

— on évalue tout d'abord a, on obtient ainsi une valeur v et une nouvelle mémoire σ',
— puis on associe à x la valeur $v + \sigma(x)$.
Donner la sémantique opérationnelle pour les constructions $x := a$ et $x +:= a$.

4. Vérifier sur $x +:= ++y$ et sur $x +:= ++x$ avec $\sigma = [x \mapsto 3, y \mapsto 5]$

5. Pour chaque couple d'expressions ou de commandes, dire en le démontrant, si elles sont sémantiquement équivalentes.
— $x+ := a$ et $x := x + a$ pour $a \in \mathbf{Aexp}'$.
— $x+ := a$ et $x := a + x$ pour $a \in \mathbf{Aexp}'$.

Partie II : Types (5 points)

Dans cet exercice, on s'intéresse aux expressions du langage `While` vu en cours. L'idée est de considérer comme un système de types la règle des signes. Celle-ci indique par exemple que la somme de deux entiers positifs est un entier positif, que le produit de deux entiers négatifs est positif, etc. Par contre, la différence de deux entiers positifs est un entier. Nous rappelons tout d'abord la syntaxe du langage vu en cours, légèrement modifiée :

$$\begin{array}{lcl} a & := & p \mid n \mid x \mid a_1 + a_2 \mid a_1 * a_2 \mid a_1 - a_2 \\ b & := & \texttt{true} \mid \texttt{false} \mid a_1 = a_2 \mid a_1 \leq a_2 \mid \neg b \mid b_1 \wedge b_2 \end{array}$$

Nous avons donc deux catégories syntaxiques *Aexp* et *Bexp* respectivement pour les expressions arithmétiques et les expressions booléennes.

La modification consiste à avoir remplacé une dénotation d'entier par une dénotation d'entier positif ou nul, notée p et une dénotation d'entier strictement négatif, notée n. On va définir 3 types `Pos, Neg, Int` qui désignent respectivement les entiers positifs ou nul, les entiers strictement négatifs et les entiers relatifs. Nous avons un ordre naturel sur les types $\texttt{Pos} < \texttt{Int}$ et $\texttt{Neg} < \texttt{Int}$.

On considère l'environnement de types qui à chaque variable associe un type :

$$\texttt{Env} = \texttt{Var} \longrightarrow \{\texttt{Pos}, \texttt{Neg}, \texttt{Int}\}.$$

Les jugements sont de la forme
— $\Gamma \vdash a : t$ avec $\Gamma \in \mathit{Env}$, $a \in \mathit{Aexp}$ et $t \in \{\texttt{Pos}, \texttt{Neg}, \texttt{Int}\}$ pour les expressions arithmétiques,
— $\Gamma \vdash b : t$ avec $\Gamma \in \mathit{Env}$, $b \in \mathit{Bexp}$ et $t \in \{\texttt{Bool}\}$ pour les expressions booléennes.
On note Γ un élément de l'environnement et $\tau, \tau_1, \ldots$ un type.

Sous-Partie A : Expressions arithmétiques

On donne ci-dessous les règles pour les constantes, les variables et certaines règles pour l'addition.

$$\Gamma \vdash p : \texttt{Pos} \qquad \Gamma \vdash n : \texttt{Neg} \qquad \Gamma \vdash x : \Gamma(x)$$

$$\frac{\Gamma \vdash a_1 : \texttt{Pos} \quad \Gamma \vdash a_2 : \texttt{Pos}}{\Gamma \vdash a_1 + a_2 : \texttt{Pos}} \qquad \frac{\Gamma \vdash a_1 : \texttt{Neg} \quad \Gamma \vdash a_2 : \texttt{Neg}}{\Gamma \vdash a_1 + a_2 : \texttt{Neg}}$$

$$\frac{\Gamma \vdash a_1 : \texttt{Neg} \quad \Gamma \vdash a_2 : \texttt{Pos}}{\Gamma \vdash a_1 + a_2 : \texttt{Int}}$$

On constate, à la lecture de ces règles que l'addition de deux entiers positifs (respectivement négatifs) a pour résultat un entier positif (respectivement négatif) mais que l'addition d'un positif et d'un négatif est un entier (cela traduit en quelque sorte une perte d'information).

Exercice 2

Donner les règles pour la soustraction et la multiplication.

Sous-Partie B : Expressions booléennes

Pour les expressions booléennes, on considère la syntaxe suivante :

$$b \quad ::= \texttt{true} \mid \texttt{false} \mid a_1 = a_2 \mid a_1 \leq a_2 \mid \neg b \mid b_1 \wedge b_2$$

Les jugements pour les expressions booléennes seront de la forme $\Gamma \vdash b : t$ avec $\Gamma \in Env$, $b \in Bexp$ et $t \in \{\texttt{Bool}, \texttt{True}, \texttt{False}\}$.

Exercice 3

1. Donner les règles pour l'égalité.
2. Donner les règles pour l'inégalité.

Partie III : optimisation (5 points)

On considère le graphe de flot de contrôle de la figure 1. Il est demandé d'utiliser la numérotation des blocs comme indiqué sur la figure.

Exercice 4

1. Calculer les ensembles $Gen(b)$ et $Kill(b)$ pour chaque bloc de base b.
2. Calculez les ensembles $In(b)$ et $Out(b)$ pour chaque bloc de base b.
3. Supprimez les calculs redondants.

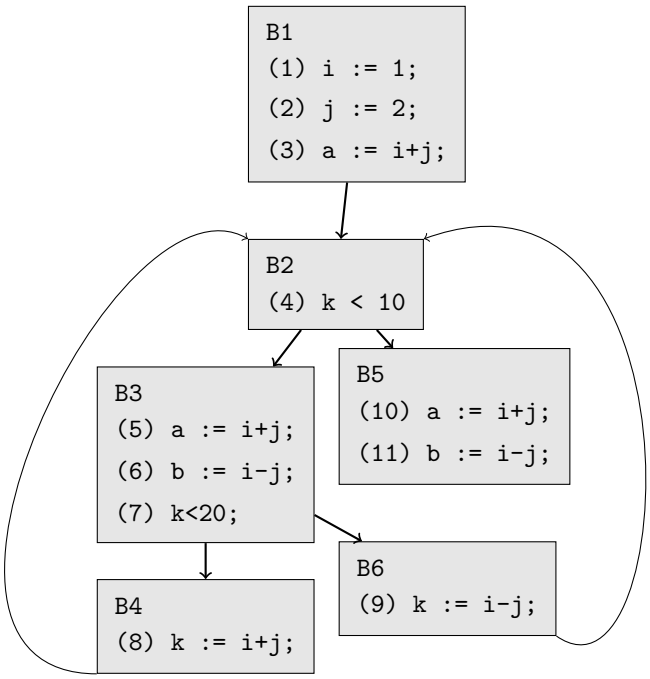

FIGURE 1 – Graphe de flot de contrôle initial