

## Devoir de maison

### Exercice 1

Nous ajoutons aux expressions arithmétiques vues en cours l'expression  $a_1 \div a_2$ , la division de  $a_1$  par  $a_2$  dans  $\mathbb{Z}$ ; c.a.d.  $7 \div 3 = 2$ . Nous voulons décrire une sémantique où cette division n'est pas définie quand  $a_2$  s'évalue à zéro.

- Modifier la fonction  $\mathcal{A}$  pour tenir compte de cette nouvelle expression.
- Comme l'évaluation des expressions peut mener à des erreurs, nous voulons prendre ceci en compte dans la sémantique opérationnelle naturelle des commandes. Nous voulons considérer deux façons de faire :
  - Nous ajoutons un état particulier que nous appelons bottom  $\perp$ . Nous voulons que l'évaluation d'une division par zéro nous mène à l'état  $\perp$  et que l'exécution de toute commande dans l'état  $\perp$  mène à l'état  $\perp$ . Donner une sémantique qui respecte ce qui vient d'être décrit.
  - Nous voulons que l'exécution de toute commande qui provoque l'évaluation d'une division par zéro puisse nous mener à n'importe quel état. Donner une sémantique qui respecte ce qui vient d'être décrit. Cette sémantique est-elle déterministe?

Quelle sémantique vous paraît la plus adéquate dans quelle situation?

### Exercice 2

On considère le langage **While** vu en cours. On ajoute la commande suivante:

$$x, y := a_1, a_2$$

C'est l'affectation simultanée de deux variables.

1. Donner un axiome ou une règle qui définit la sémantique opérationnelle naturelle de l'affectation simultanée.
2. Donner un exemple d'expressions  $a_1$  et  $a_2$  tels que les deux programmes suivants se comportent différemment :
  - $x, y := a_1, a_2$
  - $x := a_1; y := a_2$ .

Démontrer que les programmes donnés ont une sémantique différente.

3. Nous nous intéressons maintenant aux programmes suivants :

- $S; x, y := a_1, a_2$
- $S; x := a_1; y := a_2.$

Donner une condition sémantique (qui peut donc faire référence aux états) suffisante et nécessaire pour que ces deux programmes aient la même sémantique. Cette condition est-elle décidable?

4. Donner une condition syntaxique suffisante pour que ces deux programmes aient la même sémantique. Votre condition est-elle décidable?

### Exercice 3

Nous reprenons le langage **While** du cours que nous enrichissons avec la déclaration de pointeur.

Soit  $P$  un ensemble dénombrable d'identificateur disjoint de **Var**. Une variable  $p \in P$  a comme valeur une adresse dans **Adr**. L'état de la machine à l'exécution est décrit par un triplet  $(\sigma, \sigma_p, m)$  où :

1.  $\sigma : \mathbf{Var} \rightarrow \mathbb{Z}$  associe à une variable dans **Var** un entier relatif (comme dans le langage **While**),
2.  $\sigma_p : P \rightarrow \mathbf{Adr}$  associe à une variable dans  $P$  une adresse et
3.  $m : \mathbf{Adr} \rightarrow \mathbb{Z} \uplus \mathbf{Adr}$ .

On augmente les expressions les expressions  $p$  et  $a \uparrow$  obtenant :

$$a ::= x \mid n \mid a_1 \text{op} a_2 \mid p \mid a \uparrow .$$

Intuitivement,  $a \uparrow$  est la valeur mémorisée dans l'adresse associée à  $a'$ , si  $a'$  s'évalue à une adresse; et indéfinie sinon.

- **Donner la sémantique des expressions enrichies.**

On augmente les commandes par la commande :

$$Stm ::= \dots \mid a' := a$$

- Sous quelle condition la commande  $a' := a$  est bien typée.
- **Décrire intuitivement la différence entre les commandes suivantes** :  $p := p' \uparrow$ ,  $p := p'$ ,  $p := p' \uparrow \uparrow$ .
- **Donner la sémantique opérationnelle des commandes enrichies.**
- L'axiome de l'affectation pour le langage **While** reste-t-il correcte? Justifier votre réponse.