

SPECIFICATION PAR MODELES ET DEVELOPPEMENT RIGOUREUX

Durée : 3 heures.

Documents : tous documents autorisés.

PARTIE 1 : Etude des transformateurs d’expressions (9 points).

On utilise le mécanisme de substitution sur les *expressions*. Par exemple, on a :

$$[x := x + 1]x + 10 = x + 1 + 10 = x + 11$$

On peut voir la substitution simple sur une expression $[x := F] E$ comme le calcul de la valeur de E **après** l’affectation $x := F$. On envisage d’étendre cette idée aux substitutions généralisées. On note $S \diamond E$ les valeurs possibles de E après la substitution S . En effet, dans le cas où la substitution S est non déterministe, le résultat d’une telle substitution donne un “*paquet*” de valeurs. Par exemple, $(x := 1 \parallel x := 2) \diamond x$ a comme résultat le paquet 1, 2. Un paquet peut donc être constitué d’une ou de plusieurs valeurs. Un paquet vide est noté \emptyset . On note $A : B$ le fait que le paquet A est un sous-paquet du paquet B . Cela donne $2, 3 : 1, 2, 3, 4$ ou $2 : 1, 2, 3$ ou encore $2 : 2$, mais on n’a pas $2, 3 : 1, 2, 4$. Si nécessaire, on peut utiliser les délimiteurs \lceil et \lfloor autour d’un paquet pour éviter les ambiguïtés de notation. Les paquets n’ont pas de structure, c’est-à-dire $\lceil 2, 3 \rfloor, 4$ est le même paquet que $2, 3, 4$. On a aussi la propriété d’absorption: $2, 2, 3 = 2, 3$.

Les opérateurs d’union, intersection et différence ensembliste peuvent s’appliquer aux paquets, avec une signification évidente. Tous les éléments d’un paquet sont de même type et le type d’un paquet est le type de ses éléments. L’application d’une fonction à un paquet distribue sur tous les éléments du paquet: $f(\lceil x, y \rceil) = \lceil f(x), f(y) \rfloor$. Cette propriété s’applique aussi aux opérateurs binaires: $\lceil 1, 2 \rfloor + \lceil 3, 4 \rfloor = 4, 5, 6$.

Un paquet en compréhension est noté $\S x \cdot (P \mid E)$ où x est une liste de variables, P est un prédicat et E une expression. La signification est le paquet des valeurs de E pour tous les x qui satisfont P . Par exemple, $\S x \cdot (0 \leq x \wedge x \leq 3 \mid 2 * x)$ est le paquet 0, 2, 4, 6.

Dans ce contexte, une “valeur” associée à une variable est généralement un paquet. La substitution de x par un paquet B est définie par :

$$\begin{aligned} [x := B] E &= \S b \cdot (b : B \mid [x := b] E) && \text{substitution dans une expression} \\ [x := B] P &= \forall b \cdot (b : B \Rightarrow [x := b] P) && \text{substitution dans un prédicat} \end{aligned}$$

On dispose des résultats suivants sur les paquets :

- | | | |
|----|---|---------------------------------|
| L1 | $\forall x \cdot (x : A \Rightarrow x : B) = A : B$ | notion de sous-paquet |
| L2 | $\S x \cdot (P \vee Q \mid E) = \S x \cdot (P \mid E) \cup \S x \cdot (Q \mid E)$ | union de paquets |
| L3 | $\S x, y \cdot (P \wedge Q \mid E) = \S x \cdot (P \mid \S y \cdot (Q \mid E))$ | si y n’est pas libre dans P |
| L4 | $\S x \cdot (\exists y \cdot P(x, y) \mid E) = \S x, y \cdot (P(x, y) \mid E)$ | si y n’est pas libre dans E |

La notation $\S x \cdot P$ signifie $\S x \cdot (P \mid x)$, c'est-à-dire le paquet des valeurs qui satisfont P . Dans le cas de l'appartenance à un paquet, on a les résultats :

- L5 $y : \S x \cdot (P \mid E) = \exists x \cdot (P \wedge y : E)$ si y est un élément
 L6 $y : \S x \cdot P = [x := y]P$ si y est un élément (cas particulier de L5)

▷ **Question 1** (5 points)

Dans le cas où S termine, la forme générale du calcul des valeurs possibles de E après S est donnée par :

$$S \diamond E = \S x' \cdot (\text{prd}_x(S) \mid [x := x'] E)$$

où $\text{prd}_x(S)$ est le prédicat avant-après de S pour les variables x . D'après cette définition, donner les formes inductives du calcul pour :

- | | |
|---------------------------------|-----------------------------------|
| 1. $\text{skip} \diamond E$ | substitution vide |
| 2. $(x := F) \diamond E$ | affectation (substitution) simple |
| 3. $(P \implies S) \diamond E$ | substitution gardée |
| 4. $(S \parallel T) \diamond E$ | choix borné |
| 5. $(x \in A) \diamond E$ | choix dans un ensemble |
| 6. $(@z \cdot S) \diamond E$ | choix non borné |
| 7. $(S ; T) \diamond E$ | séquence de substitutions |

Indications:

- pour la formule 3, on examinera séparément le cas où P est vrai et le cas où P est faux
- pour 6, on utilisera la formule : $\text{prd}_x(@z \cdot S) \Leftrightarrow \exists z, z' \cdot (\text{prd}_{x,z}(S))$ avec $z, z' \setminus S, x, x'$
- pour 7, on rappelle que l'on se place dans le cas où la terminaison est assurée.

▷ **Question 2** (2 points)

Démontrer $\text{prd}_x(S) \Leftrightarrow x' : S \diamond x$

▷ **Question 3** (2 points)

Démontrer $\text{trm}(S) \Rightarrow ([S]Q \Leftrightarrow (S \diamond x) : \S x \cdot Q)$.

PARTIE 2 : Modélisation (11 points).

On s'intéresse à un entrepôt de produits industriels. L'entrepôt est composé de bâtiments (ensemble BAT) qui peuvent contenir des produits (ensemble $PROD$). Les bâtiments contiennent des produits et il y a des produits incompatibles. Deux produits incompatibles ne doivent pas être entreposés dans le même bâtiment. La déclaration de ces relations est :

- $\text{contient} \in BAT \leftrightarrow PROD$ les bâtiments contiennent des produits
 $\text{incomp} \in PROD \leftrightarrow PROD$ les produits incompatibles

▷ **Question 4** (2 point)

Décrire les propriétés suivantes :

1. un produit n'est pas incompatible avec lui-même

2. si un produit A est incompatible avec B , alors B est incompatible avec A .

▷ **Question 5** (2 points)

A l'aide des relations *contient* et *incomp*, définir la relation $interdit \in BAT \leftrightarrow PROD$ telle que $(bb \mapsto pp) \in interdit$ signifie que le produit pp est interdit dans le bâtiment bb . Exprimer alors la propriété de sûreté:

P1: Il n'y a pas de produits incompatibles dans un même bâtiment.

▷ **Question 6** (3 point)

Soit l'opération $stocker(bb, pp)$ qui consiste à mettre un produit pp dans le bâtiment bb . Donner sa spécification en B. Montrer comment la propriété P1 est préservée par cette opération.

▷ **Question 7** (1 point)

Soit l'opération $trouver_batiment$ qui cherche où entreposer un produit pp . Cette opération est spécifiée par :

```
bat ← trouver_batiment(pp) =
  PRE
    pp ∈ PROD
  THEN
    ANY bb WHERE bb ∈ BAT ∧ bb ↦ pp ∉ interdit THEN
      bat := bb
    END
  END
```

Etudier la faisabilité de cette opération.

▷ **Question 8** (3 point)

On raffine le problème en utilisant un tableau *stock* qui à chaque couple bâtiment-produit, associe une information (ensemble *INFO*):

- *oui*: le produit est dans le bâtiment
- *non*: le produit est interdit dans le bâtiment
- *pos*: le produit n'est pas dans le bâtiment, mais il n'est pas interdit.

Cet invariant de liaison s'écrit :

$$\begin{aligned} & stock \in BAT \times PROD \rightarrow INFO \wedge \\ & contient = stock^{-1}[\{oui\}] \wedge \\ & interdit = stock^{-1}[\{non\}] \end{aligned}$$

L'opération $trouver_batiment$ raffinée est :

```
bat ← trouver_batiment(pp) =
  ANY cc WHERE cc ∈ BAT ∧ stock(cc, pp) ≠ non THEN
    bat := cc
  END
```

Donner l'obligation de preuve de cette opération dans le raffinement, en supposant sa faisabilité. Montrer alors qu'elle peut être prouvée.