Réf: Winskel / Carton / Cormen + Apt-Olderog (non utilisé ici)

927

Exemples de preuves d'algorithmes : correction, terminaison

exemplaire Lauriane

Il est facile de vérifier que la plupart des instructions font ce qu'on attend, mais les boucles sont plus compliquées à étudier et nécessitent des outils précis.

I. Preuves informelles

1/ étude de la correction

Déf: Un *invariant de boucle* est une fonction booléenne I qui, si elle est vraie avant l'exécution du corps de la boucle, sera encore vraie après l'exécution du corps de la boucle.

Ex: TRI-INSERTION (A)

```
for j = 2 to A.length
    key = A[j]
    i = j-1
    while i>0 and A[i] > key
        A[i+1] = A[i]
        i = i-1
    A[i+1] = key
```

L'assertion I suivante est un invariant de boucle :

I: Le sous-tableau $A[1 \cdots j-1]$ contient les éléments initiallement présents dans $A[1 \cdots j-1]$, mais en ordre croissant.

Rq: Certains algorithmes incorrects peuvent quand même être intéressants (p. ex. le test de primalité de Solovay-Strassen).

2/ étude de la terminaison

Rq: Le problème de savoir si un programme termine en temps fini est indécidable, mais il existe quand même des outils qui permettent de prouver qu'un programme termine.

Déf: Un ensemble $(E,<)$ est *bien fondé* si la relation binaire $<$ est telle qu'il n'existe aucune suite infinie décroissante.

Ex: - $E = \mathbb{N}^2$ muni de l'ordre strict associé à l'ordre lexicographique.

- $E = \Sigma^*$ avec $m_1 < m_2$ si $|m_1| < |m_2|$.

Déf: Un *variant de boucle* est une quantité $V \in (E,<)$ ensemble bien fondé telle que V décroît à chaque itération de la boucle (ou à chaque appel récursif si le programme est récursif).

3/ Un exemple complet : l'algorithme de Hopcroft

L'algorithme de Hopcroft sert à construire à partir d'un automate déterministe $\mathcal{A} = (Q, A, E, I, F)$ l'automate minimal reconnaissant le même langage que $\mathcal{A}$.

```
HOPCROFT(A):
  P = (F, Q\F)
  S = {(min(F, Q\F), a) | a ∈ A}
  while S ≠ ∅ do
    (C,a) = un élément de S
    S = S \ {(C,a)}
    forall B ∈ P coupé en B1, B2 par (C,a) do
      remplacer B par B1, B2 dans P
      forall b ∈ A do
        if (B,b) ∈ S then
          remplacer (B,b) par (B1,b), (B2,b) dans S
        else
          ajouter (min(B1,B2),b) à S
```

(DEV)

II Preuves formelles

1/ Langage IMP

Déf : On définit :

$A_{exp} ::= a \mid X \mid a_1 + a_2 \mid -a_1 \mid a_1 \times a_2$

au sens où A_{exp} est le plus petit ensemble contenant les constantes $a \in \mathbb{Z}$, les variables X et étant stable par somme, produit et par passage à l'opposé. (expressions arithmétiques)

On définit aussi les expressions booléennes et les commandes :

$B_{exp} ::= b \mid X \mid \neg b_1 \mid b_1 \vee b_2 \mid b_1 \wedge b_2 \mid a_1 \leq a_2 \mid a_1 = a_2$

$Com ::= skip \mid X := a \mid \text{if } b_1 \text{ then } c_1 \text{ else } c_2 \mid \text{while } b_1 \text{ do } c_1 \mid c_1 ; c_2$

Un programme en IMP est un élément de Com

2/ Sémantique dénotationnelle

Déf : On appelle état une application σ qui associe à chaque variable une valeur. On note Σ l'ensemble des états.

On définit alors par induction les fonctions d'évaluations :

$\mathcal{A} : A_{exp} \longrightarrow (\Sigma \rightarrow \mathbb{Z})$

$\mathcal{B} : B_{exp} \longrightarrow (\Sigma \rightarrow \{V, F\})$

$\mathcal{C} : Com \longrightarrow (\Sigma \rightarrow \Sigma)$

On omettra les parenthèses en évaluant (p. ex $\mathcal{A}[\![a]\!]\sigma$).

$\mathcal{A}$ est définie par : $\mathcal{A}[\![a]\!]\sigma = a$ si a est une constante,

$\mathcal{A}[\![X]\!]\sigma = \sigma(X)$ si X est une variable,

$\mathcal{A}[\![a_1 + a_2]\!]\sigma = \mathcal{A}[\![a_1]\!]\sigma + \mathcal{A}[\![a_2]\!]\sigma$, etc...

De même, $B[\![b]\!]\sigma = b$ si b constante, etc...

$$B[\![a_1 \leq a_2]\!]\sigma = (A[\![a_1]\!]\sigma \leq A[\![a_2]\!]\sigma)$$

et on a :

$$C[\![skip]\!]\sigma = \sigma,$$

$$C[\![X := a]\!]\sigma = \sigma[A[\![a]\!]/X] := \left| \begin{array}{l} X \to A[\![a]\!] \\ Y \to \sigma(Y) \end{array}\right.$$

$$C[\![\text{if } b \text{ then } c_1 \text{ else } c_2]\!]\sigma = \begin{cases} C[\![c_1]\!]\sigma & \text{si } B[\![b]\!]\sigma \\ C[\![c_2]\!]\sigma & \text{sinon} \end{cases}$$

$$C[\![c_1 ; c_2]\!]\sigma = C[\![c_2]\!] \circ C[\![c_1]\!]\sigma$$

$$C[\![\text{while } b \text{ do } c]\!] = \lim \Theta_n$$

où Θ_0 est définie nulle part et $\Theta_{n+1}(\sigma) = \left| \begin{array}{ll} \sigma & \text{si } \neg B[\![b]\!]\sigma \\ \Theta_n \circ C[\![c]\!]\sigma & \text{sinon} \end{array}\right.$

La limite est définie au sens où le domaine de définition de Θ_n est croissant et si $\Theta_n(\sigma)$ existe, $\Theta_{n+1}(\sigma) = \Theta_n(\sigma)$.

Prop: Si on note $w =$ while b do c, alors on a :

$$C[\![w]\!] = C[\![\text{if } b \text{ then } c ; w \text{ else skip}]\!]$$

3/ Règles de Hoare

Déf: On définit l'ensemble des assertions en rajoutant des variables entières i à Aexp et en rajoutant les expressions booléennes $\forall i . A$ et $\exists i . A$:

$$\text{Assn} ::= b \mid X \mid A_1 \wedge A_2 \mid A_1 \vee A_2 \mid \neg A_1 \mid a_1 \leq a_2 \mid a_1 = a_2 \mid \forall i . A \mid \exists i . A$$

On définit alors par induction sur Assn le symbole $\models^I$ pour toute interprétation I (p.ex. $\sigma \models^I \forall i . A$ si $\sigma \models^{I[n/i]} A$ pour tout n).

On omettra dans la suite les interprétations I pour simplifier les notations.

Déf: Un triplet de Hoare est un triplet $\{A\}c\{B\}$ où $A, B \in$ Assn et $c \in$ Com.

On a $\sigma \models \{A\}c\{B\}$ si $(\sigma \models A) \Rightarrow (C[\![c]\!]\sigma \models B)$

Déf: Les règles de Hoare sont les suivantes :

$$\{A\} \text{ skip } \{A\} \qquad \{B[a/X]\} X := a \{B\}$$

$$\frac{\{A\} c_1 \{C\} \quad \{C\} c_2 \{B\}}{\{A\} c_1 ; c_2 \{B\}} \qquad \frac{\{A \wedge b\} c_1 \{B\} \quad \{A \wedge \neg b\} c_2 \{B\}}{\{A\} \text{ if } b \text{ then } c_1 \text{ else } c_2 \{B\}}$$

$$\frac{\{A \wedge b\} c \{A\}}{\{A\} \text{ while } b \text{ do } c \{A \wedge \neg b\}} \qquad \frac{\models (A \Rightarrow A') \quad \{A'\} c \{B'\} \quad \models (B' \Rightarrow B)}{\{A\} c \{B\}}$$

On écrit $\vdash \{A\}c\{B\}$ si on peut déduire $\{A\}c\{B\}$ des règles de Hoare.

Ex: Calcul de la factorielle (DEV ?) $\leftarrow$ mieux a priori un autre exemple

Th: Si $\vdash \{A\}c\{B\}$, alors $\models \{A\}c\{B\}$ (DEV)

4) Plus faibles préconditions

Déf: On appelle plus faible précondition du couple $(c, B) \in \text{Com} \times \text{Assn}$

$$WP(c, B) = \{\sigma \in \Sigma \mid C[\![c]\!]\sigma \models B\}.$$

Prop: $\models \{A\}c\{B\}$ si et seulement si $A \subset WP(c, B)$.

pas forcément utile