327 : Exemples de preuves d'algorithmes : correction & terminaison !

L'objectif de ce cours est de présenter quelques méthodes de preuves d'algorithmes.

On dit qu'un algo. **termine** s'il s'exécute en un temps fini et qu'il est correct si il réalise sa spécification.

I Preuves informelles [Cormen]

Définir spécification !

1. Algorithmes impératifs : correction

définition 1 : Invariant de boucle

Un invariant de boucle est un prédicat vrai à chaque passe de la boucle.

La méthode de l'invariant de boucle permet de valider le fonctionnement d'une boucle.

Dans le cas de boucles imbriquées, on adopte une approche bottom-left.

exemples : • l'exponentiation rapide

```
pow (a, n) =
  p <- 1
  while n > 0
    si n impair
      p <- p × a
      n <- n - 1
    sinon
      n <- ⌊n/2⌋
      p <- p * p
  renvoyer p
```

avec $n_k = \overline{n_l - n_0}^2$,

à l'étape $k \in [\![0, l]\!]$,

$p = a^{\overline{n_{k-1} \cdots n_0}^2}$

à la fin $p = a^n$.

• l'algorithme d'Euclide renvoit le pgcd de deux entiers.

2. Correction d'algorithmes récursifs

définition 2 : Relation bien fondée

Une relation binaire $\prec$ sur un ensemble E est dite bien fondée s'il n'existe pas de suite infinie décroissante.

exemple : $\mathbb{N}$ muni de $<$ est bien fondé.

Théorème 1 : Induction sur un ensemble bien fondé

Soit $(E, <)$, un ensemble bien fondé et soit p un prédicat.

Si p est vérifié pour les éléments minimaux de E et si $\forall x \in E, \left[\forall y \in E, y < x, p(y)\right] \Rightarrow p(x)$ alors, $\forall x \in E, p(x)$

remarque : sur $\mathbb{N}$, on retrouve le théorème de récurrence.

Théorème 2 : Théorème de correction

Soit $f : A \to B$ récursive et soit $\phi : A \to E$ où $(E, <)$ est un ensemble bien fondé.

Soit $M = \{x \in A, \phi(x) \text{ est minimal dans } \phi(A)\}$

Soit P_f, un prédicat sur les valeurs calculées par f.

si $\begin{cases} \forall b \in M, P_f(b) \\ \forall x \in A, \left(\forall y \in A, \phi(y) < \phi(x), P_f(y)\right) \Rightarrow P_f(x) \end{cases}$

alors P_f est vérifié pour tout calcul de f.

Exemple : Le calcul du binôme de Newton via le triangle de Pascal est correct :

$A = \mathbb{N}^2$, $E = \mathbb{N}$, $\phi : A \to E$

$(n, p) \mapsto 0$ si $p > n$, p sinon

correction | Terminaison | complexité !

3. Terminaison des algorithmes impératifs

Définition 3 : Variant de boucle
Un variant de boucle est une quantité $V \in E$ où $(E, <)$ est un ensemble bien fondé.

Théorème 3 : Théorème de terminaison
Une boucle qui possède un variant décroissant termine.

exemples : l'exponentiation rapide & l'algo. d'Euclide terminent.

4. Une preuve complète : lièvre et tortue [Weinzkel]

Développement 01 :
Soit E, un ensemble fini et soit $f : E \to E$
La suite $u_{n+1} = f(u_n)$ est périodique à partir d'un certain rang : r
L'algorithme du lièvre et de la tortue détermine le rang r et la période T en temps $O(r+T) = O(E)$.

5. Terminaison : le cas récursif

Théorème 4 : Théorème de terminaison
Soit f, A, B, φ, E et M comme dans le théorème 2, si $\forall b \in M$, $f(b)$ termine et si $\forall x \in A$, la définition de $f(x)$ ne fait apparaître que des appels (en nombre fini) de $f(y)$ où $\varphi(y) < \varphi(x)$, alors $f(x)$ termine $\forall x \in A$.

6. Terminaison : cas général

Certains algorithmes ne terminent pas :
exemple :

Morris (m, n) =
si $m = 0$, renvoyer 1
sinon Morris $(m-1, \text{Morris}(m, n))$

Le problème de terminaison est indécidable dans le cas général.

demo :
Si il existe Termine : $X \mapsto$ True si X termine
False sinon
alors pour TEST : $() \mapsto$ TEST si Termine(TEST) = True
$\mapsto$ True sinon
il y a contradiction.

exemple d'algorithme dont on ne sait pas si il termine :
COLLATZ (n) : Si $n = 1$, renvoyer 1
Sinon
si n pair, COLLATZ $(n/2)$
sinon COLLATZ $(3n+1)$

II Preuves Formelles et logique de Hoare

1. Assertions et sémantique dénotationnelle

L'objectif est de définir formellement la notion de propriété.

définition 1 : Assertion
Cette notion est inductive :
- Toute expression booléenne est une assertion
- Si p et q sont des assertions, alors,
  $\neg p$, $p \vee q$, $p \wedge q$, $p \Rightarrow q$, $p \Leftrightarrow q$ aussi
- Si p : assertion et x : variable, $\forall x : p$ et $\exists x : p$: assertions

définition 2 : état de programme ; sémantique
- On appelle état d'un programme une application σ qui associe à chaque variable une valeur.
- On note Σ l'ensemble des états.
- On appelle sémantique d'une assertion l'application $\models P : \Sigma \to \{V/F\}$
  $\sigma \mapsto$ V si p est vérifié pour σ
  F sinon

rem : on note parfois $S[\![P]\!](\sigma)$ au lieu de $\sigma \models p$.

Dévs : - lièvre et tortue
- KMP
- Dijkstra
- Bellman-Ford
) informelles

A la Hoare
- Factorielle
- Complétude

Théorème
- complétude relative
- correction
) de Hoare

2. Règles de Hoare & Preuves d'Algorithmes

Le principe de la logique de Hoare est d'introduire des règles permettant de décomposer un programme pour prouver sa correction.
On écrit alors le programme {Pré} Prog {Post}.

définition 3 : Corrections partielle et totale.
- {P} S {Q} est partiellement correct si toute exécution de S démarrant dans un état P et terminant dans un état Q.
- {P} S {Q} est totalement correct si toute exécution de S démarrant en P termine en Q.

On définit les règles de Hoare :

$$\{A\}\ SKIP\ \{A\}\ ;\quad \{B[a/X]\}\ X := a\ \{B\}\ ;\quad \frac{\{A\}P_1\{B\}\ ;\ \{B\}P_2\{C\}}{\{A\}P_1 ; P_2\{C\}}$$

$$\frac{\{A \wedge b\}P_1\{B\}\ ;\ \{A \wedge \neg b\}P_2\{B\}}{\{A\}\ \text{if } b \text{ then } P_1 \text{ else } P_2\ \{B\}}\ ;\quad \frac{\{A \wedge b\}c\{A\}}{\{A\}\ \text{while } b \text{ do } c\ \{A \wedge \neg b\}}$$

$$\frac{\models (A \Rightarrow A')\quad \{A'\}c\{B'\}\quad \models (B' \Rightarrow B)}{\{A\}c\{B\}}$$

; on écrit $\vdash \{A\}c\{B\}$ quand $\{A\}c\{B\}$ est un théorème.

Théorème : Cohérence de la logique de Hoare
Si $\vdash \{A\}c\{B\}$, alors $\models \{A\}c\{B\}$

développement 02

3. Plus faibles précondition : vers une complétude relative

définition 4 : Plus faible précondition
On appelle plus faible précondition du couple (c, B) l'ensemble
$wp(c, B) = \{\sigma \in \Sigma,\ S[\![c]\!](\sigma) \models B\}$.

Théorème : Complétude relative
$\models \{A\}c\{B\} \Leftrightarrow A \subset wp(c, B)$