927. Exemples de preuve d'algorithme : correction, terminaison.

Définition 1. Soit $f : E \to S$. On dit qu'un algorithme $\mathcal{A}$ *réalise* la fonction f si

1. pour toute entrée $e \in E$, $\mathcal{A}$ se termine (*terminaison*) ;
2. lorsque $\mathcal{A}$ se termine pour une entrée $e \in E$, sa sortie s est telle que $s = f(e)$ (*correction partielle*).

1 Généralités

1.1 Correction partielle

Définition 2. On appelle *invariant de boucle* une assertion portant sur le contexte d'une boucle qui est vraie à chaque passage.

Algorithme d'exponentiation rapide

```
procédure PUISSANCE(a, n)  ▷ On note A et N = \overline{n_ℓ ... n_0}^2 leurs valeurs initiales
    p ← 1
    tant que n > 0 faire        ▷ À l'étape k ∈ [[0, ℓ]], p = A^{\overline{n_{k-1} ... n_0}^2}, a = A^{1<<k}
        si n est impair alors   ▷ et n = \overline{n_ℓ ... n_k}^2
            p ← p × a
        a ← a × a
        n ← ⌊n/2⌋
    renvoyer p                  ▷ À la sortie de la boucle, p = A^N
```

Algorithme de calcul du PGCD

```
procédure PGCD(a, b)                ▷ Invariant : A ∧ B | a et A ∧ B | b
    si b = 0 alors          ▷ où A et B sont les valeurs initiales respectives de a et b
        renvoyer a
    sinon
        renvoyer PGCD(b, a mod b)
```

Développement 1. Soit u une suite à valeurs dans un ensemble fini. L'algorithme du lièvre et de la tortue détermine le rang r et la période T de la suite en temps $O(r + T)$ et en espace logarithmique.

1.2 Terminaison

Exemple 3. Un programme sans boucle **tant que** ni appel récursif termine.

Théorème 4. Le problème de l'arrêt est indécidable.

Conjecture de Syracuse

```
procédure SYRACUSE(n)
    tant que n > 1 faire
        si n est pair alors
            n ← n/2
        sinon
            n ← 3n + 1
    renvoyer 1
```

Définition 5. Une relation binaire $\prec$ sur un ensemble E est dite *bien fondée* s'il n'existe pas de suite infinie de E décroissante pour $\prec$. $(E, \prec)$ est alors dit *bien fondé*.

Exemple 6. $(\mathbb{N}, <)$ est un ensemble bien fondé.

Application 7. L'algorithme d'exponentiation rapide termine.

Proposition 8. Une relation bien fondée est irréflexive, sa fermeture transitive est bien fondée et sa fermeture transitive et réflexive est un ordre partiel.

Proposition 9. Si $(E, \prec_E)$ et $(F, \prec_F)$ sont des ensembles bien fondés alors $(E \times F, \prec)$ est bien fondé avec $\prec$ défini par $(m, n) \prec (m', n')$ si et seulement si $m \prec_E m'$ ou ($m = m'$ et $n \prec_F n'$).

Exemple 10. $(\mathbb{N}^2, <_{lex})$ est un ensemble bien fondé.

Application 11. L'algorithme PGCD termine.

Développement 2. L'algorithme de Knuth-Morris-Pratt détermine les bords maximaux d'une chaîne de caractères en temps et espace linéaire en la longueur de la chaîne.

2 Sémantique formelle

Définition 12. On dit qu'un énoncé $\{Pre\}$ `prog` $\{Post\}$ est valide ce qu'on note $\models \{Pre\}$ `prog` $\{Post\}$ si pour toute interprétation I :

$$\forall s \in [\![Pre]\!]_I, s \cdot \texttt{prog} \neq \bot \Rightarrow s \cdot \texttt{prog} \in [\![Post]\!]_I.$$

Définition 13. Les axiomes et règles de déduction pour la logique de Hoare sont les suivants.

$$\frac{\text{« P non modifiée par } \texttt{prog} \text{ »}}{\{P\}\ \texttt{prog}\ \{P\}} \qquad \frac{}{\{P[E/V]\}\ V \leftarrow E\ \{P\}} \qquad \frac{\{P\}S\{Q\}, \{Q\}T\{R\}}{\{P\}\ S;T\ \{R\}}$$

$$\frac{\{A \wedge c\}\ \texttt{prog}\ \{B\} \quad \{A \wedge \neg c\}\ \texttt{prog}'\ \{B\}}{\{A\}\ \textbf{si}\ c\ \textbf{alors}\ \texttt{prog}\ \textbf{sinon}\ \texttt{prog}'\ \{B\}} \qquad \frac{\{A \wedge c\}\ \texttt{prog}\ \{A\}}{\{A\}\ \textbf{tant que}\ c\ \textbf{faire}\ \texttt{prog}\ \{A \wedge \neg c\}}$$

$$\frac{\models (A \Rightarrow A') \quad \{A'\}\ \texttt{prog}\ \{B'\} \quad \models (B' \Rightarrow B)}{\{A\}\ \texttt{prog}\ \{B\}} \qquad \frac{\{A\}\ \texttt{prog}\ \{B\} \quad \{A'\}\ \texttt{prog}\ \{B'\}}{\{A \wedge A'\}\ \texttt{prog}\ \{B \wedge B'\}}$$

Algorithme d'Euclide étendu

procédure BÉZOUT(a, b) ▷ On note A et B les valeurs initiales de a et b
  $\{b \geqslant 0 \wedge a = A \wedge b = B\}$ ▷ et $\Delta = A \wedge B$
  $u, v, x, y \leftarrow 1, 0, 0, 1$ ▷ $\{d|n\} \equiv \{\exists k, k \geqslant 1 \wedge k \leqslant n \wedge n = kd\}$
  $\{b \geqslant 0 \wedge Au + Bv = a \wedge Ax + By = b \wedge \Delta|a \wedge \Delta|b\}$
  tant que $b \neq 0$ **faire**
    $\{b \geqslant 0 \wedge Au + Bv = a \wedge Ax + By = b \wedge \Delta|a \wedge \Delta|b\}$
    $q, r = \lfloor a/b \rfloor, a \bmod b$
    $\{b \geqslant 0 \wedge Au + Bv = a \wedge Ax + By = b \wedge a = qb + r \wedge \Delta|b \wedge \Delta|r\}$
    $m, n = u - xq, v - yq$
    $\{b \geqslant 0 \wedge Ax + By = b \wedge Am + Bn = r \wedge \Delta|b \wedge \Delta|r\}$
    $a, b, u, v, x, y \leftarrow b, r, x, y, m, n$
    $\{b \geqslant 0 \wedge Au + Bv = a \wedge Ax + By = b \wedge \Delta|a \wedge \Delta|b\}$
  $\{b = 0 \wedge Au + Bv = a \wedge \Delta|a \wedge \Delta|b\} \equiv \boxed{\{a = \Delta \wedge Au + Bv = \Delta\}}$
  renvoyer a, u, v

Proposition 14. Ce système de déduction est cohérent et incomplet.

Développements

Algorithme de Knuth-Morris-Pratt
Algorithme du lièvre et de la tortue

Références

Cormen (et un peu Demazure aussi)
Winskel

Algorithme du lièvre et de la tortue

procédure FLOYD(u_0, f)
  lièvre $\leftarrow f(f(u_0))$
  tortue $\leftarrow f(u_0)$
  tant que *lièvre* $\neq$ *tortue* **faire**
    lièvre $\leftarrow f(f(\textit{lièvre}))$
    tortue $\leftarrow f(\textit{tortue})$
  tortue $\leftarrow f(\textit{tortue})$
  $T \leftarrow 1$
  tant que *lièvre* $\neq$ *tortue* **faire**
    tortue $\leftarrow f(\textit{tortue})$
    $T \leftarrow T + 1$
  lièvre $\leftarrow u_0$
  $r \leftarrow 0$
  tant que *lièvre* $\neq$ *tortue* **faire**
    lièvre $\leftarrow f(\textit{lièvre})$
    tortue $\leftarrow f(\textit{tortue})$
    $r \leftarrow r + 1$
  renvoyer T, r

Algorithme de Knuth-Morris-Pratt

procédure KMP(s) ▷ s est de longueur n et $s = s_0 \cdots s_{n-1}$
  $(\ell_0, \ldots, \ell_n) \leftarrow (0, \ldots, 0)$
  pour $i = 1, \ldots, n$ **faire**
    $k \leftarrow i - 1$
    tant que $s_{\ell_k} \neq s_{i-1}$ et $k > 0$ **faire**
      $k \leftarrow \ell_k$
    si $s_{\ell_k} = s_{i-1}$ **alors**
      $\ell_i \leftarrow \ell_k + 1$
    sinon
      $\ell_i \leftarrow 0$
  renvoyer $(\ell_i)_{0 \leqslant i \leqslant n}$