UFR IM²AG

UNIVERSITÉ Grenoble Alpes

L3 Informatique
Année 2017–2018

UE ALGO5 — TD2 — Séance 4 : Drapeau arc-en-ciel

On s'intéresse à la généralisation du problème du drapeau hollandais. Les données sont un tableau de couleurs $C[1..P]$, et un tableau $T[1..N]$ d'éléments de C. Le problème consiste à réorganiser le tableau par tranche de couleurs, ordonnées de la même manière que dans C, sans utiliser un algorithme de tri classique.

Le but de cette séance est de faire pratiquer l'écriture d'invariants d'itération, et d'appuyer la réalisation d'un algorithme sur l'invariant écrit. Il faut donc mettre l'accent sur l'exercice 2, faire écrire l'invariant de la manière la plus précise possible (si possible avec dessins + notation formelle), et ne laisser passer les étudiants à la réalisation complète qu'une fois le schéma et l'invariants complètement écrits.

Normalement, le problème est assez simple pour pouvoir écrire la spécification et les invariants de manière formelle.

Exercice 1.

Q 1. Compléter la spécification de la procédure DrapeauAEC :

```
1 Couleur : type abstrait

3 DrapeauAEC(C,P,T,N)
  { Données :
5     P, N : entiers
      C : tableau sur [1..P] de couleurs
7     T : tableau sur [1..N] de couleurs
      Pré−condition : ...
9     Post−condition : ...
  }
```

Corrigé —

Pré-condition: il faut expliciter et formaliser le fait que :
—toutes les couleurs de C sont différentes (1)
—toutes les valeurs de T sont dans C (2)

$$\forall i \in [1,P], \forall j \in [1,P], i \neq j \Rightarrow C[i] \neq C[j] \quad (1)$$
$$\forall i \in [1,N], \exists j \in [1,P], T[i] = C[j] \quad (2)$$

Post-condition: il est temps de réfléchir à ce que va construire l'algorithme. Pour le drapeau hollandais, la post-condition peut s'exprimer à l'aide de 2 indices (pour délimiter 3 tranches dans le tableau). Ici il faut utiliser $P-1$ indices.

Il existe $i(1), ..., i(P)$ tels que :

$i(0)$	$i(1)$	$i(2)$	$i(P-1)$ N $i(P)$
$= C[1]$	$= C[2]$		$= C[P]$

D'où :

$\exists i(0), ..., i(P)$ t.q.

—$\forall j \in [0, P-1], 1 \leq i(j) \leq i(j+1)$
—$i(0) = 1$
—$i(P) = N+1$
—$\forall j \in [1, N], \exists k \in [1, P], i(k-1) \leq j \leq i(k)$ et $T[j] = C[k]$
... et bien sûr, le tableau T est une permutation du tableau initial (vu en TD...).

Q 2. Écrire le schéma et l'invariant de la boucle principale de l'algorithme.

Corrigé —

On utilise donc un tableau I[0..P] pour stocker les indices délimitant les tranches du tableau.

On peut se baser sur la post-condition pour construire l'invariant de la boucle principale :

$I[0]$	$I[1]$	$I[2]$	$I[P-1]$	$I[P]$ N
$= C[1]$	$= C[2]$		$= C[P]$	non traités

Le corps de l'itération consiste donc à insérer l'élément à l'indice $I[P]$ à la bonne place. Comme pour le drapeau hollandais, il n'est pas nécessaire de décaler tous les éléments : on peut faire des permutations successives de l'élément à placer avec les éléments situés aux indices $I[j]$.

On peut donc exprimer l'invariant de l'itération interne (celle qui va placer l'élément x) :

$I[0]$	$I[1]$	$I[2]$	$I[j-1]$	$I[j]$		$I[P-1]$	$I[P]$ N
$= C[1]$	$= C[2]$		$= C[J]$	x		$= C[P]$	non traités

Avec $\exists k \in [1, P]$ t.q. $x = C[k]$ et $k \leq j$.

Q 3. Écrire l'algorithme complet.

Corrigé —

```
DrapeauAEC(C, P, T, N)
    I : tableau sur [0..P] d'entiers
    pour j de 0 à P faire
        I[j] ← 1
    fin
    tant que I[P] ≠ N + 1 faire
        I[P] ← I[P] + 1
        // Décalages successifs
        j ← P
        tant que T[I[j]] ≠ C[j] faire
            // Échange de x avec le premier élément de la tranche courante
            tmp ← T[I[j]]
            T[I[j]] ← T[I[j − 1]]
            T[I[j − 1]] ← tmp

            I[j − 1] ← I[j − 1] + 1
        fin
    fin
```
