Université Lille1 Sciences et Technologies

Info 204 - ASD

2013–2014
Licence STS - Semestre 4

DS1 - documents de cours, TD, TP autorisés

Note : lorsque, dans une question, est demandée la complexité, c'est une fonction de la taille de la donnée qui est attendue. Lorsque c'est le comportement asymptotique, la réponse attendue est soit une notation $\mathcal{O}$, soit une notation Θ soit une notation Ω.

Vous pourrez utiliser toute fonction vue en cours, TD, TP à la condition expresse de spécifier correctement les entrées et les sorties.

Exercice 1 : Questions de cours [4 points]

Q 1.1 [2 points] Un algorithme s'exécute dans le meilleur des cas en réalisant $c_m(n) = 2n + 1$ opérations et dans le pire des cas en $c_p(n) = 2^n + 4n - 2$ opérations pour une donnée de taille n. Indiquez et prouvez son comportement asymptotique.

Q 1.2 [2 points] Le nombre d'appels récursifs d'un algorithme récursif est donné par la formule :

$$c(n) = \begin{cases} 1 & \text{si } n = 0 \text{ ou } 1 \\ 16c\left(\frac{n}{4}\right) + n^3 & \text{sinon} \end{cases}$$

Indiquez et prouvez son comportement asymptotique.

Exercice 2 : Calculer une complexité et améliorer un algorithme [3 points]

Dans cet exercice on considère des tableaux d'entiers. On dit qu'un élément e est majoritaire dans un tableau t de longueur n si t contient plus de $\frac{n}{2}$ occurrences de e. Pour simplifier, on supposera traiter le cas particulier où il existe toujours un élément majoritaire. On propose la fonction `element_majoritaire` qui calcule l'élément majoritaire de t en utilisant la fonction `decompte`.

```
let decompte t e =
  let n = Array.length t
  and nb = ref 0
  in
  for i = 0 to n-1 do
    if t.(i) = e then nb := !nb + 1;
  done;
  !nb
```

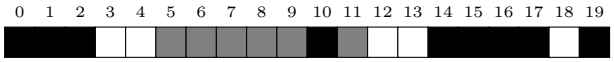
```
let element_majoritaire t =
  let n = Array.length t
  and max = ref 0
  and valmax = ref 0
  in
  for i = 0 to n-1 do
    let nb = decompte t t.(i)
    in
    if nb > !max then begin
      max := nb;
      valmax :=  t.(i);
    end
  done;
  !valmax
```

Q 2.1 [1 point] Quelle est l'opération à prendre en compte pour le calcul de la complexité en temps ?

Q 2.2 [2 points] Quelle est la complexité en temps ? Justifiez.

Exercice 3 : Concevoir un algorithme d'une complexité donnée [6 points]

On s'intéresse à un problème d'analyse d'images : « Etant donné une ligne de pixels de différentes couleurs, on souhaite identifier la position et la longueur du plus long segment monochrome dans la ligne. » Par exemple, étant donné la ligne suivante :



notre algorithme indiquera que le plus long segment monochrome débute en position 5 et est de longueur 5. Pour simplifier, on supposera qu'une ligne de pixels est représentée par un tableau d'entiers, chaque entier codant pour une couleur. La ligne de pixels ci-dessus sera par exemple représentée par :

```
[| 56;56;56;0;0;34;34;34;34;34;56;34;0;0;56;56;56;56;0;56 |]
```

Q 3.1 [2 points] Proposez une fonction `plsm` écrite en CAML qui résolve ce problème dont la complexité en temps soit **linéaire** (on compte le nombre de comparaisons de couleurs). La fonction prend en entrée un tableau d'entiers t et retourne un couple avec la position p et la longueur ℓ tel que ℓ est la longueur du plus long segment monochrome et p la position à laquelle il débute.

Q 3.2 [1 point] Justifiez la complexité de votre algorithme.

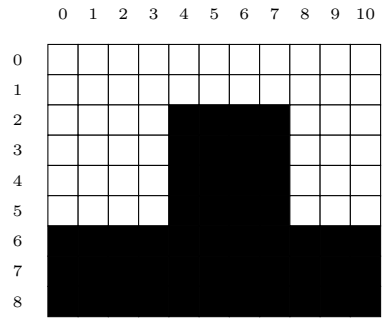
On souhaite maintenant obtenir les couleurs dominantes de la ligne. Le nombre de couleurs dominantes que l'on souhaite extraire dépend de l'application réalisée. Toujours sur l'exemple ci-dessus, l'extraction d'une couleur dominante retournera 56 et l'extraction des deux couleurs dominantes retournera 56 et 34.

Q 3.3 [2 points] Ecrivez une fonction `couleurs_dominantes` en CAML qui prend en entrée une ligne de pixels t, un entier k et retourne un tableau avec les k couleurs dominantes de t (on supposera que le nombre de couleurs de t est toujours supérieur ou égal à k). La complexité de votre fonction devra être en $\mathcal{O}(n \log n)$ (on compte toujours le nombre de comparaisons de couleurs).

Q 3.4 [1 point] Justifiez la complexité de votre algorithme.

Exercice 4 : Mettre en œuvre un algorithme de programmation dynamique [7 points]

On s'intéresse à nouveau dans cet exercice à un problème d'analyse d'image. On considère cette fois des images en noir et blanc et le problème qu'on veut traiter est la détection du plus grand carré de cases noires dans l'image. Par exemple, pour l'image suivante :



l'algorithme indiquera que le plus grand carré noir se situe en case (4,2) [1] et est de longueur 4.

On propose la récursion suivante pour résoudre le problème. On note $\ell(i,j)$ la longueur du plus grand carré noir se terminant en position $(i-1, j-1)$ dans l'image. On a alors :

$$\ell(i,j) = \begin{cases} 0 & \text{si la case de l'image en position } (i-1,j-1) \text{ est blanche} \\ 1 + \min(\ell(i-1,j-1), \ell(i-1,j), \ell(i,j-1)) & \text{sinon} \end{cases}$$

1. D'autres coordonnées sont possibles – (4,3), 4,4), (4,5) – , c'est simplement celles que mon algorithme calcule

et

$$\ell(i,0) = \ell(0,j) = 0 \text{ pour } 0 \leq i \leq n, 0 \leq j \leq m$$

L'obtention du résultat, c'est-à-dire la plus grande longueur L, consiste à sélectionner le couple (i,j) maximisant $l(i,j)$:

$$L = \max_{\substack{1 \leq i \leq n \\ 1 \leq j \leq m}} \ell(i,j)$$

si on considère une image de taille $n \times m$.

Nous allons donc mettre en œuvre une résolution par programmation dynamique. Pour simplifier, on supposera qu'une image est donnée par un tableau à deux dimensions dont les cases contiennent soit 0 soit 1, 0 indiquant les cases blanches, et 1 les cases noires. On manipulera trois variables déclarées comme ci-dessous :

```
let img =
  [|
    [| 0;0;0;0;0;0;1;1;1 |];
    [| 0;0;0;0;0;0;1;1;1 |];
    [| 0;0;0;0;0;0;1;1;1 |];
    [| 0;0;0;0;0;0;1;1;1 |];
    [| 0;0;1;1;1;1;1;1;1 |];
    [| 0;0;1;1;1;1;1;1;1 |];
    [| 0;0;1;1;1;1;1;1;1 |];
    [| 0;0;1;1;1;1;1;1;1 |];
    [| 0;0;0;0;0;0;1;1;1 |];
    [| 0;0;0;0;0;0;1;1;1 |];
    [| 0;0;0;0;0;0;1;1;1 |]
  |]
;;
let n = Array.length img;;
let m = (Array.length img.(0));;
```

Q 4.1 [1 point] Quelle est la taille de la table de programmation dynamique ?

Q 4.2 [1 point] Donnez le code CAML qui réalise l'initialisation de la table.

Q 4.3 [2 points] Donnez le code CAML qui réalise le remplissage de la table.

Q 4.4 [1.5 point] Donnez le code CAML qui extrait le résultat.

Q 4.5 [1.5 point] Quel est le comportement asymptotique en temps ? Justifez clairement.