(PSI*)

Devoir surveillé d'informatique

(jeudi 16 décembre 2004)

L'utilisation des calculatrices n'est pas autorisée pour cette épreuve. Le langage de programmation choisi par le candidat doit être spécifié en tête de la copie.

* * *

Coloriage

On attachera une grande importance à la concision, à la clarté, et à la précision de la rédaction.

Le temps d'exécution $T(f)$ d'une fonction f est le nombre d'opérations élémentaires (addition, soustraction, multiplication, division, affectation, etc.) nécessaire au calcul de f. Lorsque ce temps d'exécution dépend d'un paramètre n, il sera noté $T_n(f)$. On dit que la fonction f s'exécute:

- en temps linéaire en n, s'il existe $K > 0$ tel que pour tout n, $T_n(f) \leqslant Kn$.;
- en temps quadratique en n, s'il existe $K > 0$ tel que pour tout n, $T_n(f) \leqslant Kn^2$.

Cercles d'amis

On recherche les cercles d'amis dans un ensemble $C = \{c_0, c_1, \ldots, c_{n-1}\}$ de n personnes ($n > 0$). À un moment donné, c_{i_1} et c_{j_1} sont amis; c_{i_2} et c_{j_2} sont amis; $\ldots c_{i_s}$ et c_{j_s} sont amis. On notera par $\sim$ la relation d'amitié. On a donc $c_{i_1} \sim c_{j_1}$, $c_{i_2} \sim c_{j_2}$, $\ldots c_{i_s} \sim c_{j_s}$, ($s > 0$).

La relation d'*amitié indirecte* $\equiv$ est définie selon le principe que « les amis de mes amis sont mes amis ». On pose donc:

$$c_i \equiv c_j \text{ si et seulement si } c_i = c_{k_0} \sim c_{k_1} \sim c_{k_2} \ldots \sim c_{k_\ell} = c_j \ (\ell \geqslant 0).$$

Pour $\ell = 0$, on a donc $c_i \equiv c_i$. Il en résulte que c_i et c_j sont des amis indirects s'ils sont amis ou s'ils ont un ami indirect en commun. On dira aussi que c_i et c_j appartiennent à un même cercle d'amis.

Ainsi en prenant $n = 7$, $s = 4$ et $c_2 \sim c_1,\ c_3 \sim c_6,\ c_4 \sim c_5,\ c_2 \sim c_6$, les cercles d'amis sont $\{c_0\}$, $\{c_1, c_2, c_3, c_6\}$, $\{c_4, c_5\}$. Il y a trois cercles d'amis: c_0 n'a d'autre ami que lui-même; $c_1,\ c_2,\ c_3,\ c_6$ sont des amis indirects; c_4 et c_5 sont aussi amis.

On cherche à répondre à la question suivante: étant donné un couple (i, j), a-t-on $c_i \equiv c_j$?

Dans chaque cercle d'amis, on élit un représentant (unique). La relation d'amitié indirecte $\equiv$ est représentée par un tableau E de n éléments vérifiant la propriété suivante: en partant de toute entrée i dans E, la suite définie par $i_0 = i$ et $i_{k+1} = E[i_k]$ pour $k \geqslant 0$ converge vers un ℓ vérifiant $\ell = E[\ell]$ qui est le représentant unique c_ℓ du cercle d'amis de c_i.

Ainsi dans l'exemple précédent, en prenant c_0, c_1 et c_5 comme représentants de $\{c_0\}$, $\{c_1, c_2, c_3, c_6\}$ et $\{c_4, c_5\}$, le tableau $E = \langle 0, 1, 1, 6, 5, 5, 2\rangle$ peut représenter la relation d'amitié indirecte $\equiv$. Mais on aurait pu aussi prendre $E = \langle 0, 1, 3, 1, 5, 5, 3\rangle$; etc.

Question 1 Donner la valeur du tableau E quand $s = 0$. Donner ses valeurs possibles quand $s = 1$ et $c_i \sim c_j$ pour deux valeurs i et j données.

Pour déterminer le représentant c_ℓ du cercle d'amis de c_i, on explore le tableau E à partir de l'indice i jusqu'à trouver ℓ tel que $E[\ell] = \ell$. Dans l'exemple, quand $E = \langle 0, 1, 1, 6, 5, 5, 2\rangle$, on trouve le représentant du cercle d'amis de c_6, en cherchant celui de c_2 puisque $E[6] = 2$, puis celui de c_1 puisque $E[2] = 1$, pour finalement trouver c_1 puisque $E[1] = 1$.

Question 2 Écrire une fonction `representant`(E, i) retournant, en temps linéaire par rapport à n, l'indice ℓ du représentant c_ℓ du cercle d'amis de c_i.

Question 3 Écrire une fonction `sontAmis`(E, i, j) qui retourne la valeur *vrai* si $ci \equiv c_j$ et *faux* sinon. (Dans les langages de programmation où les valeurs booléennes n'existent pas, on rendra l'entier 0 pour la valeur *faux* et 1 pour la valeur *vrai*).

On veut maintenant modifier la relation $\equiv$ en ajoutant la nouvelle relation $c_{i_{s+1}} \sim c_{j_{s+1}}$. Deux nouveaux amis viennent de se déclarer; leurs cercles d'amis respectifs peuvent donc fusionner. On doit modifier le tableau E tout en préservant sa propriété d'indiquer les représentants. On ajoute une relation entre le représentant c_ℓ de $c_{i_{s+1}}$ et le représentant $c_{\ell'}$ de $c_{j_{s+1}}$. On peut donc poser soit $E[\ell] = \ell'$ soit $E[\ell'] = \ell$.

Question 4 Écrire une fonction `ajout`(E, i, j) qui modifie le tableau E représentant la relation $\equiv$ après l'ajout de la nouvelle relation $c_i \sim c_j$. Donner un ordre de grandeur du temps d'exécution de cette fonction par rapport à n.

Question 5 Montrer sur un exemple que le choix laissé arbitraire dans la modification du tableau E peut influer sur le temps d'exécution de la fonction `representant`. Donner une piste pour garantir un meilleur temps d'exécution.

Dans la suite, on dira que c_i et c_j sont « équivalents » si $c_i \equiv c_j$.

Coloriage d'objets

On cherche à déterminer les p objets contenus dans *une image numérique noir et blanc* T représentée par une matrice $R \times R$. Typiquement, $R = 1024$ ou $R = 2048$. Chaque élément (i, j) de l'image ($0 \leqslant i < R$, $0 \leqslant j < R$), encore appelé *pixel*, est un entier valant 0 (blanc) ou 1 (noir). La couleur du fond de l'image est supposée blanche; un pixel valant 1 appartient à un des p objets. Le but de cette partie consiste à colorier chaque pixel de T par le numéro k ($1 \leqslant k \leqslant p$) de l'objet auquel il appartient. On réserve la valeur 0 pour les pixels du fond de l'image, c'est-à-dire valant déjà la valeur 0 dans l'image initiale. Par convention, la première ligne de la matrice est en haut de l'image (nord) et la première colonne est à gauche (ouest).

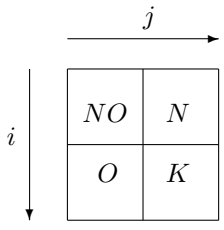
Dans l'image suivante, il y a 5 objets dans l'image de gauche qui devient après coloriage l'image de droite:

00000000011010000000	00000000011020000000
00111000011010001100	00333000011020004400
11001100100010001100	33003300500020004400
11111000100010000000	33333000500020000000
01000000100011011100	03000000500022022200
00000000000000110000	00000000000000220000
...	...

Question 6 Écrire une fonction `couleurNO`(T, i, j) qui retourne la couleur du pixel $t_{i-1,j-1}$ au nord-ouest du pixel $t_{i,j}$ pour i, j vérifiant $1 \leqslant i < R$ et $1 \leqslant j < R$. Cette fonction retourne aussi la couleur du fond lorsque $i = 0$ ou $j = 0$. Écrire de même les fonctions `couleurN`(T, i, j) et `couleurO`(T, i, j) qui retournent respectivement les couleurs du pixel au nord et à l'ouest du pixel $t_{i,j}$.

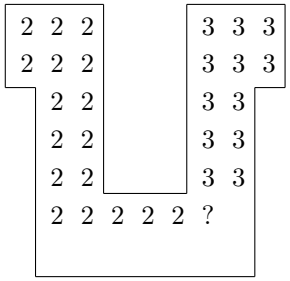
La reconnaissance des différents objets d'une image T se fait en deux phases. D'abord, on identifie les objets figurant sur l'image. Par la suite, on procède au coloriage de tous les pixels. Par convention, on dira que deux pixels (valant 1) appartiennent à un même objet s'ils sont connectés sur des axes nord/sud, est/ouest ou nord-ouest/sud-est (Attention, ils sont dans des objets différents s'ils sont uniquement connectés nord-est/sud-ouest). Plus exactement, les pixels $t_{i,j}$ et $t_{i',j'}$ sont dans un même objet si $t_{i,j} = t_{i',j'} = 1$ et on a soit $i = i'$ et $|j - j'| \leqslant 1$, soit $j = j'$ et $|i - i'| \leqslant 1$, soit $(i - i')(j - j') = 1$.

La phase d'identification des objets se fait par modification progressive de la matrice T en déplaçant de gauche à droite et de haut en bas une fenêtre de quatre pixels. On suppose que les pixels NO au nord-ouest, N au nord, et O à l'ouest du pixel courant K en bas à droite de la fenêtre sont déjà identifiés. Soient $\underline{no}$, $\underline{n}$, $\underline{o}$ leurs couleurs respectives.



Pour identifier le pixel non blanc K, on applique les règles suivantes, l'une après l'autre, en s'arrêtant lorsqu'une condition est satisfaite:

1. Si $\underline{no} \neq 0$, on colorie K avec la couleur de $\underline{no}$.
2. Si une seule des deux couleurs $\underline{o}$ et $\underline{n}$ n'est pas nulle, alors K est colorié avec cette couleur non nulle.
3. Si $\underline{o} = \underline{n} = 0$, alors K n'appartient à aucun objet reconnu jusque là et on lui attribue une nouvelle couleur.
4. Si $\underline{o} = \underline{n} \neq 0$, alors K est colorié avec cette couleur.
5. Si $\underline{o} \neq 0$, $n \neq 0$ et $\underline{o} \neq \underline{n}$, on ajoute l'équation $\underline{o} \sim \underline{n}$ pour dire que les deux couleurs sont équivalentes et on colorie K avec une de ces deux couleurs.



Ainsi, dans l'exemple ci-dessus, les couleurs 2 et 3 sont équivalentes.

La variable globale nc aura comme valeur le nombre de couleurs nécessaires pour identifier les objets de T. On suppose que cette valeur est bornée par une constante C pas trop grande, et on gère les équivalences entre couleurs avec un tableau E de longueur C selon les méthodes de la première partie.

Question 7 Écrire une fonction `marquage`(T) qui identifie les objets de la matrice de pixels T selon la méthode décrite précédemment, tout en modifiant les valeurs de la variable nc et du tableau E global représentant les équivalences entre les couleurs attribuées aux pixels de T. Donner un ordre de grandeur de son temps d'exécution en fonction de R et C.

La deuxième phase de l'algorithme consiste à re-parcourir l'image de haut en bas et de gauche à droite en coloriant avec une même couleur tous les pixels appartenant à un même objet. Si l'image contient p objets, les couleurs utilisées seront prises entre 1 et p (tout en gardant 0 pour la couleur du fond). Le tableau E servira à résoudre les équivalences entre couleurs trouvées pendant la première phase pour transformer toutes les couleurs équivalentes en une seule couleur.

Question 8 Écrire une fonction `diffusion`(T) qui effectue la deuxième phase en coloriant avec une même couleur les points ayant des couleurs équivalentes. Donner un ordre de grandeur de son temps d'exécution en fonction de R et C. En déduire la fonction `colorier`(T) effectuant le coloriage de T.

* * *