Option Informatique en Spé MP et MP*

Devoir surveillé du 19 mars 2003

***

Résumé

Le premier problème porte sur la programmation en Caml.

Le deuxième problème étudie la notion d'arbre recouvrant minimal (en anglais : *minimal spanning tree*) d'un graphe valué. Nous analysons deux algorithmes de construction d'un tel arbre : le premier a été publié par Kruskal en 1956, le second par Prim en 1957 (mais était déjà connu de Jarník en 1930).

Veuillez rédiger chaque partie sur une copie séparée.

Table des matières

1 Programmation **2**

2 Arbre recouvrant minimal **3**

2.1 Graphes valués : définitions et notations 3

2.2 Quelques questions d'ordre général 3

2.3 L'algorithme de Kruskal 4

2.4 L'algorithme de Prim 4

1 Programmation

▶ Chaque question fixe un *objectif* : c'est le nombre maximal de lignes que vous devez écrire. Vous compterez pour une ligne l'en-tête d'une fonction, comme :

```
let toto x y = function
```

Vous compterez chaque motif pour une ligne ; notez bien qu'un motif union, comme :

```
| [] | [true] | [_;false;_] -> ...
```

compte pour une seule ligne.

▶ L'emploi de références ou de structures de données mutables est interdit, tout comme le recours aux constructions `for ... do ... done` et `if ... then ... else ...`

Question 1 • Rédigez en Caml une fonction :

```
est_constante : 'a list -> bool
```

spécifiée comme suit : `est_constante s` indique si la liste s est constante, c'est-à-dire si tous ses membres sont égaux. Le coût, exprimé en nombre de *Cons*, doit être un $\mathcal{O}(|s|)$. Objectif : 3 lignes.

▶ Nous nous proposons de rédiger en Caml une fonction :

```
est_monotone : int list -> bool
```

spécifiée comme suit : `est_monotone s` indique si la liste s est monotone (croissante ou décroissante, au sens large). Le coût, exprimé en nombre de *Cons*, doit être un $\mathcal{O}(|s|)$.

Question 2 • Commencez par rédiger une fonction `est_croissante` qui indique si la liste transmise en argument est croissante au sens large. Objectif : 3 lignes.

Question 3 • Utilisez cette fonction (et une autre, très semblable, mais que l'on ne vous demande pas de rédiger) pour répondre à la question posée. Objectif : 2 lignes.

▶ Dans certains contextes, le programme doit traiter chaque élément dans un laps de temps borné : ce mode de fonctionnement est dit *en ligne* (en anglais, *on line*).

Question 4 • Expliquez pourquoi votre réponse à la question précédente ne satisfait pas ce critère.

Question 5 • Remettez-vous au travail pour livrer une nouvelle version de `est_monotone` satisfaisant la contrainte de fonctionnement en ligne. Objectif : 5 lignes.

▶ Soient $x = (x_i)_{0 \leqslant i < n}$ et $y = (y_j)_{0 \leqslant j < p}$ deux listes, de longueurs respectives n et p. Nous dirons que x est *extraite* de y s'il existe une injection croissante s de $[\![0, n-1]\!]$ dans $[\![0, p-1]\!]$ telle que $x_i = y_{s(i)}$ pour tout $i \in [\![0, n-1]\!]$. Par exemple, la liste $x = (8, 2, 7, 3, 4, 9)$ est extraite de la liste $y = (1, 2, 8, 0, 2, 7, 7, 7, 4, 3, 4, 3, 4, 8, 9)$, en prenant (par exemple) s définie par $s(0) = 2$, $s(1) = 4$, $s(2) = 5$, $s(3) = 9$, $s(4) = 12$ et $s(5) = 14$.

Question 6 • Rédigez en Caml une fonction :

```
est_extraite_de : 'a list * 'a list -> bool
```

spécifiée comme suit : `est_extraite_de (x,y)` indique si la liste x est *extraite* de la liste y. Le coût, exprimé en nombre de *Cons*, doit être un $\mathcal{O}(|x| + |y|)$. Objectif : 5 lignes.

2 Arbre recouvrant minimal

2.1 Graphes valués : définitions et notations

▶ Un *graphe valué* est un triplet (S, A, π) où S est un ensemble fini de *sommets*, A un ensemble d'*arêtes* (une arête est une paire $\{x, y\}$ de sommets distincts) et π une application de A dans $\mathbb{R}$: $\pi\big(\{x, y\}\big)$ est le *poids* de l'arête $\{x, y\}$. La figure 1 représente un graphe valué comportant huit sommets et onze arêtes.

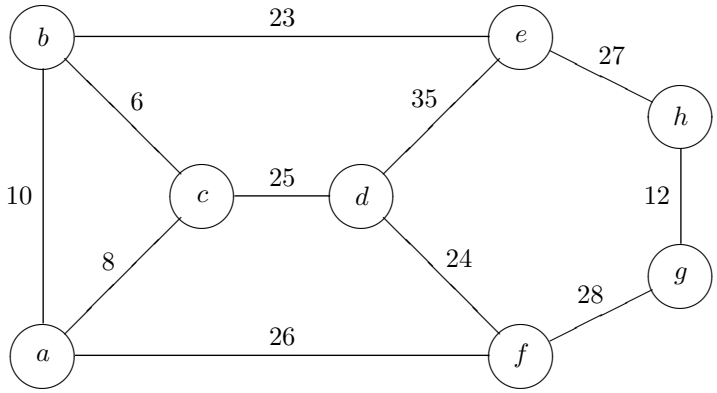

Figure 1: un graphe valué

▶ Les notions de *voisin*, de *chemin*, de *cycle* et de *connexité* sont connues. Un chemin est *élémentaire* s'il ne repasse pas deux fois par un même sommet. Un graphe est *acyclique* s'il ne comporte aucun cycle. Tous les graphes considérés dans la suite sont connexes et non orientés ; on note $|S|$ le nombre de sommets et $|A|$ le nombre d'arêtes.

▶ $G' = (S', A')$ est un *sous-graphe* de $G = (S, A)$ si $S' \subset S$ et $A' \subset A$; c'est un *graphe partiel* si de plus $S' = S$. Soit S' une partie de S, et A' l'ensemble des arêtes de G reliant deux sommets appartenant à S' ; nous dirons que (S', A') est le sous-graphe de G *induit* par S'.

▶ Rappel : soit $G = (S, A)$ un graphe à n sommets ; les assertions suivantes sont deux à deux équivalentes, et définissent un *arbre* :

1. G est connexe et acyclique
2. G est connexe et comporte $n - 1$ arêtes
3. G est acyclique et comporte $n - 1$ arêtes
4. G est acyclique, mais ne le reste pas si on lui ajoute une arête
5. G est connexe, mais ne le reste pas si on lui enlève une arête
6. deux sommets quelconques de G sont reliés par un et un seul chemin élémentaire

▶ Un *arbre recouvrant* de G est un graphe partiel de G connexe et acyclique : ce qui justifie sa dénomination. L'arbre est minimal si la somme des poids de ses arêtes est minimale.

▶ Une *forêt* est une famille d'arbres deux à deux disjoints ; la définition d'une *forêt recouvrante* est évidente.

2.2 Quelques questions d'ordre général

Question 1 • Justifiez l'existence d'un arbre recouvrant minimal. Donnez un exemple *très simple* où l'on n'a pas l'unicité.

Question 2 • Montrez que si les poids des arêtes sont deux à deux distincts, il n'existe qu'un seul arbre recouvrant minimal.

Question 3 • Montrez que l'on peut supposer, sans perte de généralité, que les poids des arêtes sont tous strictement positifs.

Question 4 • L'étudiant Maurice ALANISETTE se propose d'appliquer une méthode *diviser pour régner* à la recherche d'un arbre recouvrant minimal d'un graphe valué $G = (S, A, \pi)$ donné. Il procède comme suit : si le graphe n'a qu'un sommet, le problème est résolu ; sinon, soient $S = S_1 \cup S_2$ une partition de l'ensemble des sommets, T_1 un arbre recouvrant minimal du sous-graphe de G induit par S_1, et T_2 défini de façon analogue. Parmi les arêtes reliant un sommet de S_1 et un sommet de S_2, choisissons-en une a de poids minimal ; en réunissant T_1, T_2 et cette arête, on obtient un arbre recouvrant minimal du graphe G. Que pensez-vous de cette méthode ?

▶ Lorsque j'ai posé ce problème pour la première fois (il y a six ans), la plupart des étudiants n'ont pas fait la différence entre l'algorithme de KRUSKAL et celui de PRIM. Vous lirez donc soigneusement les parties 2.3 et 2.4 pour bien comprendre les fonctionnements respectifs de ces deux algorithmes.

2.3 L'algorithme de Kruskal

▶ Nous décrivons un premier algorithme de construction d'un arbre recouvrant minimal.

Question 5 • Soit $\{x, y\}$ une arête de poids minimal du graphe valué $G = (S, A, \pi)$. Montrez que cette arête fait partie d'un arbre recouvrant minimal.

Question 6 • Soit $G_1 = (S_1, A_1)$ un sous-graphe d'un arbre recouvrant minimal T de G. On suppose S_1 contenu *strictement* dans S. Montrez qu'il existe au moins une arête $a = \{x, y\} \in G \setminus A_1$ telle que, notant $S_2 = S_1 \cup \{x, y\}$ et $A_2 = A_1 \cup \{a\}$, le graphe $G_2 = (S_2, A_2)$ soit acyclique.

Question 7 • Montrez que l'on peut choisir a de façon que $G_2 = (S_2, A_2)$ soit un sous-graphe d'un arbre recouvrant minimal de G.

Question 8 • En déduire un algorithme de construction d'un arbre recouvrant minimal. Note : on ne demande pas, ici, d'écrire un programme en Caml ; en revanche, la preuve du bon fonctionnement de l'algorithme devra apparaître sur la copie.

Question 9 • Appliquez cet algorithme au graphe de la figure 1.

Question 10 • Comment modifier l'algorithme précédent pour obtenir un arbre recouvrant de poids *maximal* ?

Question 11 • Que pensez-vous de l'algorithme suivant, proposé par l'étudiant Marcel-Paul TORTUEUX : on examine les arêtes par poids *décroissants* ; si l'arête examinée participe à un cycle, on la supprime, sinon on la garde et on passe à la suivante.

2.4 L'algorithme de Prim

▶ Nous décrivons un deuxième algorithme de construction d'un arbre recouvrant minimal.

Question 12 • Soit x un sommet du graphe valué $G = (S, A, \pi)$, et $\{x, y\}$ une arête de poids minimal parmi celles qui relient x à un autre sommet. Montrez que $\{x, y\}$ fait partie d'un arbre recouvrant minimal.

Question 13 • Soit A_1 une partie de A strictement contenue dans l'ensemble des arêtes d'un arbre recouvrant minimal T de G. Soit S_1 l'ensemble des sommets reliés par les arêtes de A_1 ; on suppose S_1 contenu *strictement* dans S. Justifiez l'existence d'une arête $a = \{u, v\}$ telle que $u \in S_1$, $v \notin S_1$, et de poids minimal pour cette propriété.

Question 14 • Notant $S_2 = S_1 \cup \{v\}$ et $A_2 = A_1 \cup \{a\}$, montrez que (S_2, A_2) est un sous-graphe d'un arbre recouvrant minimal.

Question 15 • En déduire un algorithme de construction d'un arbre recouvrant minimal. Note : on ne demande pas, ici, d'écrire un programme en Caml ; en revanche, la preuve du bon fonctionnement de l'algorithme devra apparaître sur la copie.

Question 16 • Appliquez cet algorithme au graphe de la figure 1, en prenant comme sommet initial a.

Question 17 • Que se passe-t-il lorsque l'on applique l'algorithme de PRIM à un graphe valué dont toutes les arêtes ont le même poids ?

FIN