Vestiges troglodytiques

Épreuve pratique d'algorithmique et de programmation
Concours commun des écoles normales supérieures

Durée de l'épreuve: 3 heures 30 minutes

Juillet 2008

ATTENTION !

N'oubliez en aucun cas de recopier votre u_0
à l'emplacement prévu sur votre fiche réponse

Important.

Sur votre table est indiqué un numéro u_0 qui servira d'entrée à vos programmes. Les réponses attendues sont généralement courtes et doivent être données sur la fiche réponse fournie à la fin du sujet. À la fin du sujet, vous trouverez en fait deux fiches réponses. La première est un exemple des réponses attendues pour un $\widetilde{u_0}$ particulier (précisé sur cette même fiche et que nous notons avec un tilde pour éviter toute confusion!). Cette fiche est destinée à vous aider à vérifier le résultat de vos programmes en les testant avec $\widetilde{u_0}$ au lieu de u_0. Vous indiquerez vos réponses (correspondant à votre u_0) sur la seconde et vous la remettrez à l'examinateur à la fin de l'épreuve.

En ce qui concerne la partie orale de l'examen, lorsque la description d'un algorithme est demandée, vous devez présenter son fonctionnement de façon schématique, courte et précise. Vous ne devez en aucun cas recopier le code de vos procédures!
Quand on demande la complexité en temps ou en mémoire d'un algorithme en fonction d'un paramètre n, on demande l'ordre de grandeur en fonction du paramètre, par exemple: $O(n^2)$, $O(n \log n)$,...
Il est recommandé de commencer par lancer vos programmes sur de petites valeurs des paramètres et de ***tester vos programmes sur des petits exemples que vous aurez résolus préalablement à la main ou bien à l'aide de la fiche réponse type fournie en annexe***. Enfin, il est recommandé de lire l'intégralité du sujet avant de commencer afin d'effectuer les bons choix de structures de données dès le début.

1 Introduction

Une équipe de la section 32 du CNRS («mondes anciens et médiévaux») vient de découvrir les vestiges d'une civilisation oubliée ! Leurs fouilles ont permis de mettre à jour l'entrée d'un réseau de grottes. Il semblerait que les différentes grottes soient reliées entre elles par des galeries et que ces galeries aient été creusées de façon à ce qu'il y ait un unique chemin entre chaque paire de grottes. Ce réseau de grottes forme donc un arbre et seule la grotte principale, dont l'entrée vient d'être découverte, est reliée au monde extérieur. Ce réseau est en très mauvais état et il est donc difficile à explorer. Par chance, ce peuple avait pris soin de graver une carte sur l'un des murs de l'entrée principale.
Devant le danger d'une telle exploration, les scientifiques ont décidé d'utiliser un robot explorateur. Ce robot partira de la grotte principale et visitera donc un certain nombre de grottes avant de revenir à son point de départ pour rendre compte de son exploration. Seulement, les batteries de ce robot sont limitées et il ne peut parcourir qu'une distance limitée avant d'être à court d'énergie. Votre objectif est donc d'aider nos archéologues à programmer ce robot afin qu'il explore le plus grand nombre de grottes possibles.
Il est naturel de modéliser ce problème à l'aide d'un arbre dont chacune des arêtes est étiquetée par une distance.

Définition 1 (Arbre) *Un arbre est un couple $\mathcal{T} = (V, E)$ où $V = \{1, \ldots, n\}$ est un ensemble de nœuds et $E \subset V \times V$ est une relation (aussi appelée ensemble d'arêtes) sur l'ensemble des nœuds vérifiant la propriété suivante :*

Pour tout $j \in \{2, \ldots, n\}$, il existe un unique $i \in \{1, \ldots, j-1\}$ tel que $(i, j) \in E$.

Un tel i est appelé père de j et noté $p(j)$. Pour tout $i \in V$, on appelle fils de i l'ensemble, noté $f(i)$, des j tels que $(i, j) \in E$. Seul le nœud 1 n'a pas de père : ce nœud particulier est appelé racine de l'arbre. Enfin, les nœuds n'ayant pas de fils sont appelés feuilles.
On appelle degré $d(i)$ d'un nœud i, son nombre de fils : $d(i) = card(f(i))$.
On définit la hauteur $h(i)$ d'un nœud i par la relation suivante :

$$h(i) = \begin{cases} 0 & si\ f(i) = \emptyset \\ \max_{j \in f(i)} h(j) + 1 & sinon \end{cases}$$

La hauteur d'un arbre est la hauteur de la racine.

Définition 2 (Arbre étiqueté) *On appellera arbre étiqueté un triplet $\mathcal{T} = (V, E, w)$ où (V, E) est un arbre et $w : E \mapsto \mathbb{N}^*$ indique la longueur de chaque arête.*
Pour un arbre étiqueté, on peut définir la profondeur maximale $\pi(i)$ d'un nœud i par la relation suivante :

$$\pi(i) = \begin{cases} 0 & si\ f(i) = \emptyset \\ \max_{j \in f(i)} w(i, j) + \pi(j) & sinon \end{cases}$$

De même, la profondeur totale $\sigma(i)$ d'un nœud i est définie par la relation suivante :

$$\sigma(i) = \begin{cases} 0 & si\ f(i) = \emptyset \\ \sum_{j \in f(i)} w(i, j) + \sigma(j) & sinon \end{cases}$$

2 Génération aléatoire de réseaux

Considérons la suite d'entiers (u_k) définie pour $k \geqslant 0$ par :

$$u_k = \begin{cases} \textit{votre } \mathsf{u}_0 \textit{ (à reporter sur votre fiche)} & \text{si } k = 0 \\ 15\,091 \times u_{k-1} \mod 64\,007 & \text{si } k \geqslant 1 \end{cases}$$

Question 1 *Que valent :* ***a)*** u_{10} ***b)*** u_{100} ***c)*** u_{1000}

On s'assurera de pré-calculer et stocker suffisamment de valeurs de u_n de manière à pouvoir y accéder en temps constant par la suite.

Définition 3 *Pour $n \in \mathbb{N}^*$ on note $\mathcal{T}_n = (V, E, w)$ l'arbre étiqueté défini par :*
- $V = \{1, \ldots, n\}$,
- $E = \left\{ \left(1 + \left\lfloor \frac{(j-1).u_{2j}}{64007} \right\rfloor, j\right) | j \in \{2, \ldots, n\} \right\}$,
- $w(p(j), j) = 2. \left(1 + \left\lfloor \frac{400.u_{2j+1}}{64007} \right\rfloor\right)$

Question 2 *Calculer le plus grand degré des nœuds ainsi que le nombre de feuilles des arbres suivants :* ***a)*** $\mathcal{T}_{10}$ ***b)*** $\mathcal{T}_{100}$ ***c)*** $\mathcal{T}_{200}$

Question 3 *Calculer $\pi(1)$ et $\sigma(1)$ pour les arbres suivants :* ***a)*** $\mathcal{T}_{10}$ ***b)*** $\mathcal{T}_{100}$ ***c)*** $\mathcal{T}_{200}$

Définition 4 (Tournée) *Une tournée $\mathcal{S}$est un sous-arbre (enraciné en 1) $\mathcal{S} = (V_\mathcal{S}, E_\mathcal{S})$ de $\mathcal{T} = (V, E, w)$, c'est-à-dire un arbre tel que $V_\mathcal{S} \subseteq V$ et $E_\mathcal{S} \subseteq E$.*
La longueur $l(\mathcal{S})$ d'une tournée $\mathcal{S}$est égale à la profondeur totale de la racine dans l'arbre $\mathcal{S}$. Le poids d'une tournée est le cardinal de $V_\mathcal{S}$.

Notre robot explorateur a une autonomie de d et devra rentrer dans les grottes puis ressortir. Chaque arête va donc être parcourue deux fois, d'où le facteur 2 dans la définition des arbres $\mathcal{T}_n$ afin de pouvoir directement utiliser les notions que nous venons de définir (profondeur, longueur, ...). L'objectif de la section suivante va donc porter sur le problème suivant :

> Étant donnée une distance maximale d, notre objectif dans la prochaine section va donc être de trouver une tournée $\mathcal{S}$ de longueur inférieure ou égale à d et de poids maximal.

3 Optimisation d'une tournée

Une première idée consiste à construire une tournée de façon gloutonne. En partant de la racine, l'arbre est exploré récursivement en choisissant en priorité les sous-arbres dont les racines sont les plus proches. En cas d'égalité, on choisira prioritairement le nœud dont l'indice est le plus petit.

Question 4 *En fixant $d = 1000$, calculer le poids de la tournée ainsi construite pour les arbres suivants :* ***a)*** $\mathcal{T}_{10}$ ***b)*** $\mathcal{T}_{100}$ ***c)*** $\mathcal{T}_{200}$

Définition 5 (Étoile) *À tout arbre $\mathcal{T} = (V, E, w)$, on associe l'arbre $\widehat{\mathcal{T}} = (V, \widehat{E}, \widehat{w})$ tel que $\widehat{E} = \{(1, j) | (i, j) \in E\}$ et $\widehat{w}(1, j) = w(p(j), j)$. L'arbre $\widehat{\mathcal{T}}$ est donc de hauteur 1 et a la même profondeur totale que $\mathcal{T}$.*

Définition 6 (Chaîne) *À tout arbre $\mathcal{T} = (V, E, w)$, on associe l'arbre $\widetilde{\mathcal{T}} = (V, \widetilde{E}, \widetilde{w})$ tel que $\widetilde{E} = \{(j-1, j) | (i, j) \in E\}$ et $\widetilde{w}(j-1, j) = w(p(j), j)$. L'arbre $\widetilde{\mathcal{T}}$ est donc de hauteur $card(V) - 1$ et a la même profondeur totale que $\mathcal{T}$.*

Question à développer pendant l'oral : Montrer que la stratégie précédente est optimale pour les étoiles et pour les chaînes mais qu'elle ne l'est pas dans le cas général.

Question 5 *En fixant $d = 1000$, calculer le poids de la tournée optimale pour les arbres suivants :* ***a)*** $\widehat{\mathcal{T}}_{10}$ ***b)*** $\widehat{\mathcal{T}}_{100}$ ***c)*** $\widehat{\mathcal{T}}_{200}$

Question 6 *En fixant $d = 1000$, calculer le poids de la tournée optimale pour les arbres suivants :* ***a)*** $\widetilde{\mathcal{T}}_{10}$ ***b)*** $\widetilde{\mathcal{T}}_{100}$ ***c)*** $\widetilde{\mathcal{T}}_{200}$

Pour construire une tournée optimale, on va s'intéresser aux fonctions $\lambda_i : \mathbb{Z} \mapsto \mathbb{N}$ qui à d associent le poids maximal des tournées de longueur inférieure ou égale à d dans le sous-arbre enraciné en i.
Ces fonctions sont clairement croissantes et bornées par le nombre de nœuds du sous-arbre correspondant et $\lambda_i(d+1) \leqslant \lambda_i(d) + 1$. Elles se représentent donc de façon unique par leur support, c'est-à-dire par un ensemble de valeurs $-\infty = d_{-1} < d_0 = 0 < d_1 < \dots < d_n < d_{n+1} = +\infty$ tel que pour tout $d \in [d_i, d_{i+1}[$: $\lambda(d) = i + 1$.
On vient de voir qu'il était aisé de construire une telle fonction lorsque le sous-arbre considéré est une étoile ou une chaîne. Si i a pour fils j_1 et j_2, on a :

$$\lambda_i(d) = 1 + \max_{\substack{d_1 + d_2 = d \\ d_1 \geqslant 0, d_2 \geqslant 0}} \lambda_{j_1}(d_1 - w(i, j_1)) + \lambda_{j_2}(d_2 - w(i, j_2))$$

Question 7 *En déduire un algorithme pour calculer récursivement les fonctions λ_i et indiquer le poids de la tournée optimale (avec $d = 6000$) pour les arbres suivants :*
a) $\mathcal{T}_{10}$ ***b)*** $\mathcal{T}_{100}$ ***c)*** $\mathcal{T}_{200}$

Question à développer pendant l'oral : Donner la complexité de l'algorithme calculant λ_1.

4 Plusieurs tournées

Le robot explorateur ressortant des grottes après chaque exploration, il est possible de l'envoyer plusieurs fois. Le temps de charge des batteries étant relativement long, on va donc s'intéresser dans cette section à la minimisation du nombre de tournées nécessaires à la visite de l'intégralité des grottes.

Définition 7 (Minimisation des tournées) *Étant donnés un arbre $\mathcal{T}$ et une longueur d maximale, trouver un ensemble de p tournées $(\mathcal{S}_1, \mathcal{S}_2, \dots, \mathcal{S}_p)$ de longueurs inférieures ou égales à d, couvrant $\mathcal{T}$ et tel que p soit minimal.*

On cherchera donc à minimiser p tout en s'assurant que $\bigcup_{j=1}^{p} V_{\mathcal{S}_j} = V$ *et* $l(\mathcal{S}_j) \leqslant d$ *pour tout* j.

4.1 Cas des étoiles

Dans le cas des étoiles, on se ramène au problème suivant :

Définition 8 (Problème d'empaquetage) *Étant donnée une liste d'entiers positifs* $A = (a_1, \ldots, a_n)$ *et* d *un entier positif supérieur à tous les* a_i*, trouver la plus petite partition de* A *en* p *boites* $B_1, \ldots, B_p$ *de capacité* d*.*
On cherchera donc à minimiser p *tout en vérifiant que pour tout* $j \in \{1, \ldots, p\}$ $\sum_{a_i \in B_j} a_i \leqslant d$*.*

Un algorithme naïf pour résoudre un problème d'empaquetage consiste à remplir les boites gloutonnement :
- On essaye d'insérer un à un les éléments dans une première boite en passant à l'élément suivant en cas d'échec. On arrive ainsi à remplir une première boite.
- On introduit alors une seconde boite et on réessaye d'insérer un à un les éléments restants toujours dans le même ordre jusqu'à ce que cette seconde boite soit pleine.
- On recommence avec une nouvelle boite et avec les éléments restants jusqu'à ce que tous les éléments de A aient été insérés.

On arrive ainsi à trouver un empaquetage de A.

Question 8 *En fixant* $d = 2500$*, calculer le nombre de tournées nécessaires, à l'aide de cette technique (en prenant les fils de la racine par indice croissant), pour les arbres suivants :* **a)** $\widehat{\mathcal{T}}_{10}$ **b)** $\widehat{\mathcal{T}}_{100}$ **c)** $\widehat{\mathcal{T}}_{200}$

En prenant les a_i dans un autre ordre, on obtient cependant un empaquetage différent, potentiellement meilleur. En notant qu'une solution optimale à un problème d'empaquetage définit naturellement une permutation de A à partir de laquelle la technique précédente redonne l'empaquetage initial (optimal), notre problème se ramène donc à trouver une permutation de A.

Question à développer pendant l'oral : Donner une instance sur lequel le nombre de tournées calculé par l'algorithme naïf de la question précédente n'est pas optimal. En cas de manque d'inspiration, vous pourrez par exemple générer aléatoirement des listes A et tester toutes les permutations possibles.

Intuitivement, les a_i les plus difficiles à placer sont les plus gros et il vaut donc mieux commencer par eux. On propose donc de d'abord trier la liste A par ordre décroissant avant de procéder à l'empaquetage glouton.

Question 9 *En fixant* $d = 2500$*, calculer le nombre de tournées nécessaires à l'aide de cette technique pour les arbres suivants :* **a)** $\widehat{\mathcal{T}}_{10}$ **b)** $\widehat{\mathcal{T}}_{100}$ **c)** $\widehat{\mathcal{T}}_{200}$

Question à développer pendant l'oral : Donner une instance sur lequel le nombre de tournées calculé par cet algorithme n'est pas optimal.

4.2 Cas général

Faisons quelques remarques sur notre problème de minimisation de tournée :
- Si $d < \pi(1)$, le problème n'a pas de solution.
- Dans le cas d'une chaîne, une seule tournée est nécessaire.
- Dans le cas d'une étoile, le problème se ramène à trouver une partition des feuilles et on dispose d'un algorithme relativement efficace effectuant d'abord un tri, puis ensuite un empaquetage glouton.

Question à développer pendant l'oral : Soit $(\mathcal{S}_1, \mathcal{S}_2, \dots, \mathcal{S}_p)$ un ensemble de tournées couvrant $\mathcal{T}$. Montrer que l'on peut construire un ensemble de tournées $(\mathcal{S}'_1, \mathcal{S}'_2, \dots, \mathcal{S}'_p)$ tel que toute feuille de $\mathcal{S}'_j$ est une feuille de $\mathcal{T}$ et les $\mathcal{S}'_j$ induisent une partition des feuilles de $\mathcal{T}$.

Fort des remarques précédentes, le problème de minimisation de tournées se ramène à un problème de partition des feuilles de $\mathcal{T}$. La permutation dans le cas des étoiles correspond dans le cas général à un parcours de l'arbre. Ce parcours définit une permutation des feuilles et la partition s'obtient alors gloutonnement de la façon suivante :
- On essaye de visiter les feuilles dans l'ordre définit par le parcours tant que possible (i.e. tant que la tournée est de longueur inférieure ou égale à d). On passe à la feuille suivante en cas d'échec et on arrive ainsi à constituer une première tournée.
- On construit alors une seconde tournée sur le même principe en visitant les feuilles restantes dans le même ordre jusqu'à ce que cette seconde tournée soit pleine.
- On recommence avec une nouvelle tournée et avec les feuilles restantes jusqu'à ce que toutes les feuilles (et donc tout l'arbre) aient été visitées.

Question 10 *On effectue un parcours prenant les nœuds par indice croissant. En fixant $d = 2500$, calculer le nombre de tournées nécessaires à l'aide de cette technique pour les arbres suivants :* ***a)*** $\mathcal{T}_{10}$ ***b)*** $\mathcal{T}_{100}$ ***c)*** $\mathcal{T}_{200}$

Question 11 *On suppose qu'on effectue cette fois-ci un parcours par profondeur totale σ décroissante (et en cas d'égalité, on commencera par les nœuds de plus petit indice). En fixant $d = 2500$, calculer le nombre de tournées nécessaires à l'aide de cette technique pour les arbres suivants :* ***a)*** $\mathcal{T}_{10}$ ***b)*** $\mathcal{T}_{100}$ ***c)*** $\mathcal{T}_{200}$

Fiche réponse type: Vestiges troglodytiques

$\widetilde{u_0}$: 27

Question 1

a) 41917

b) 9802

c) 28386

Question 2

a) 3 5

b) 6 51

c) 8 100

Question 3

a) 1508 3536

b) 3854 40500

c) 4444 77646

Question 4

a) 4

b) 5

c) 5

Question 5

a) 6

b) 16

c) 26

Question 6

a) 3

b) 3

c) 3

Question 7

a) 10

b) 27

c) 30

Question 8

a) 2

b) 17

c) 32

Question 9

a) 2

b) 17

c) 32

Question 10

a) 2

b) 25

c) 58

Question 11

a) 2

b) 24

c) 55

Fiche réponse: Vestiges troglodytiques

Nom, prénom, u_0: ....

Question 1

a)

b)

c)

Question 2

a)

b)

c)

Question 3

a)

b)

c)

Question 4

a)

b)

c)

Question 5

a)

b)

c)

Question 6

a)

b)

c)

Question 7

a)

b)

c)

Question 8

a)

b)

c)

Question 9

a)

b)

c)

Question 10

a)

b)

c)

Question 11

a)

b)

c)