Nom :

Prénom :

Login EPITA :

UID :

Examen d'algorithmique

EPITA ING1 2010 S1, A. DURET-LUTZ

Durée : 1 heure 30

Janvier 2008

Consignes

- Tous les documents (sur papier) sont autorisés.
  Les calculatrices, téléphones, PSP et autres engins électroniques ne le sont pas.
- Répondez sur le sujet dans les cadres prévus à cet effet.
- Il y a 6 pages. Rappelez votre UID en haut de chaque page au cas où elles se mélangeraient.
- Ne donnez pas trop de détails. Lorsqu'on vous demande des algorithmes, on se moque des points-virgules de fin de commande etc. Écrivez simplement et lisiblement. Des spécifications claires et implémentables sont préférées à du code C ou C++ verbeux.
- Le barème est indicatif et correspond à une note sur 20.

File (4 points)

On souhaite représenter un ensemble dynamique S d'éléments entiers. Cet ensemble, initialement vide, peut être augmenté ou diminué par les opérations suivantes :
- `insert` ajoute un élément à S, et
- `extract_min` retire de S son plus petit élément.

1. **(1 point)** En une phrase, de quelle façon doit-on modifier l'implémentation par tas de la file de priorité vue en cours pour pouvoir effectuer ces deux opérations efficacement ? (Rappel : la file de priorité du cours supporte `insert` et `extract_max`.)

Réponse :

2. **(1 point)** Avec cette nouvelle implémentation quelles sont les complexités du temps d'exécution de `insert` et `extract_min` en fonction du nombre n d'éléments de S ?

Réponse :

3. **(1 point)** Toujours avec cette implémentation par tas, donnez l'état du tableau représentant S après y avoir effectué chacune des opérations suivantes :
   - `insert(6)`
   - `insert(2)`
   - `insert(5)`
   - `insert(3)`
   - `extract_min()`
   - `insert(7)`
   - `insert(8)`
   - `extract_min()`

4. **(1 point)** Sur cette structure de données, quelle serait la complexité d'une opération `extract_next_to_min` pour retirer de S le deuxième plus petit élément (sans retirer le premier) ? Justifiez votre réponse.

Réponse :

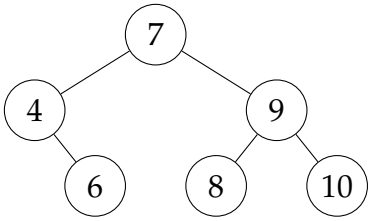
Arboriculture (6 points)

Dans toutes ces questions, on considère des arbres binaires d'entiers.

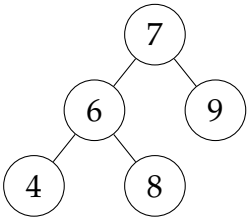
1. **(2 points)** Proposez un algorithme récursif retournant `true` si et seulement si l'arbre binaire passé en argument est un Arbre Binaire de Recherche.

   Attention, la première idée n'est pas toujours la bonne. Assurez-vous au moins que votre algorithme fonctionne sur les deux exemples qui suivent.

Cet arbre est un ABR :



Celui-là n'en est pas un :

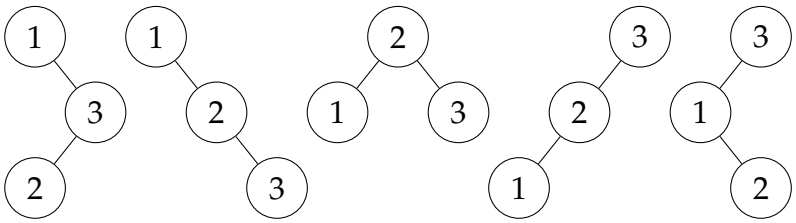


Réponse :

2. **(1 point)** Donnez la complexité de cet algorithme en fonction du nombre n de nœuds de l'arbre inspecté. Justifiez votre réponse.

Réponse :

3. **(2 points)** Notons $A(n)$ le nombre d'arbres binaires de recherche différents qui permettent de représenter n entiers donnés. Par exemple il existe 5 ABR représentant $\{1, 2, 3\}$:



On a $A(0) = 1$ (l'arbre vide), $A(1) = 1$, $A(2) = 2$, $A(3) = 5$, $A(4) = 14$.
Donnez une définition récursive de $A(n)$.

Réponse :

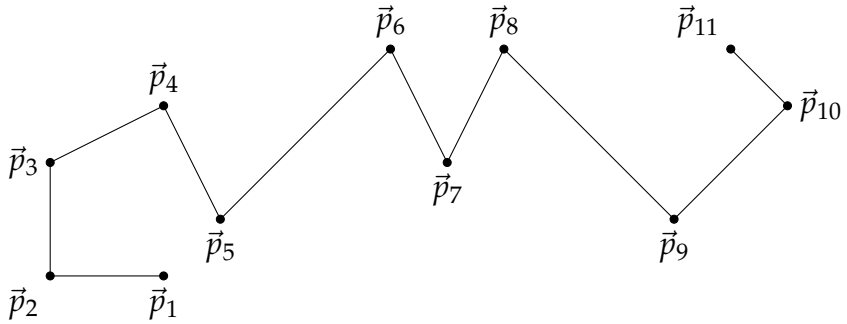
4. **(1 point)** Notons $B(n)$ le nombre d'arbres binaires différents que l'on peut construire en utilisant n nœuds. (Cette fois-ci les arbres ne sont pas des arbres de recherche.) On a $B(3) = 5$ car il n'existe que cinq « formes » d'arbres qui utilisent 3 nœuds.
Donnez une définition de $B(n)$. (Vous pouvez y utiliser $A(n)$.)

Réponse :

Segmentation par les moindres carrés (6 points)

Dans cet exercice nous souhaitons épurer un chemin enregistré par un récepteur GPS. Le récepteur GPS enregistre sa position à intervalles réguliers, ce qui nous permet de tracer le chemin parcouru en reliant ces positions. Pour simplifier nous allons travailler en deux dimensions, sur un plan.

Voici un exemple de chemin possédant 11 points notés $\vec{p}_1 \dots \vec{p}_{11}$. Dans le cas général on notera n le nombre de points enregistrés.

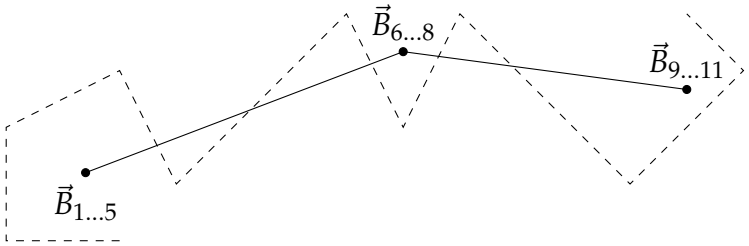


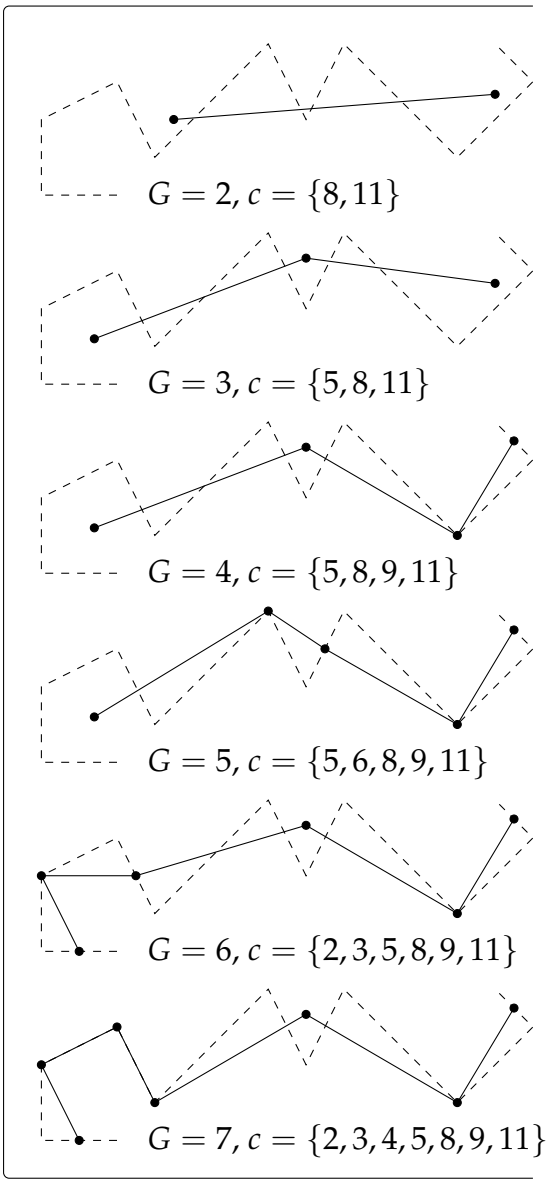
Notre objectif est de simplifier cette ligne brisée pour obtenir un nombre de points $G \leq n$ donné. On va pour cela regrouper les n points en G groupes de points consécutifs ; chaque groupe de points sera représenté par son barycentre. Les groupes seront formés de façon à minimiser la somme des carrés des distances entre chacun des points $\vec{p}_k$ et le barycentre $\vec{B}_{i\dots j}$ de leur groupe.

Par exemple pour $G = 3$, un regroupement optimal des points de notre exemple est

$$\underbrace{\vec{p}_1, \vec{p}_2, \vec{p}_3, \vec{p}_4, \vec{p}_5}_{\vec{B}_{1\dots5}}, \underbrace{\vec{p}_6, \vec{p}_7, \vec{p}_8}_{\vec{B}_{6\dots8}}, \underbrace{\vec{p}_9, \vec{p}_{10}, \vec{p}_{11}}_{\vec{B}_{9\dots11}}$$

Graphiquement, on a :





Comme on ne regroupe que des points consécutifs, on pourra représenter un regroupement par l'ensemble c des indices des derniers points de chaque groupe. Dans notre exemple avec $G = 3$, on a $c = \{5, 8, 11\}$. L'encart de droite montre des regroupements optimaux pour diverses valeurs de G sur ce même exemple.

Formalisons tout ceci plus proprement. Tout d'abord, définissons $\vec{B}_{i..j}$ le barycentre des vecteurs $\vec{p}_i$ à $\vec{p}_j$ et $\mathrm{SDC}(i,j)$ la somme des distances au carré entre ces vecteurs et leur barycentre :

$$\vec{B}_{i...j} = \frac{1}{j-i+1} \sum_{k=i}^{j} \vec{p}_k$$

$$\mathrm{SDC}(i,j) = \sum_{k=i}^{j} \left\| \vec{p}_k - \vec{B}_{i...j} \right\|^2$$

Pour un G et un ensemble de $\vec{p}_i$ donnés, notre objectif est de trouver l'ensemble $c = \{c_1, c_2, \ldots, c_G\}$ qui minimise $D = \mathrm{SDC}(0, c_1) + \mathrm{SDC}(c_1 + 1, c_2) + \cdots + \mathrm{SDC}(c_{G-1} + 1, c_G)$.

On cherche à minimiser cette distance par programmation dynamique.

Notons $P(k,j)$ la valeur de D minimale que l'on peut obtenir pour le problème réduit à la recherche de k groupes parmi les j premiers points. On a, pour tout $j \leq n$:

$$\begin{aligned} P(1,j) &= \mathrm{SDC}(1,j) \\ P(k,j) &= \min_{i \in \{k-1,\ldots,j-1\}} \{P(k-1,i) + \mathrm{SDC}(i+1,j)\} \quad \text{si } 2 \leq k \leq G \end{aligned}$$

1. **(2 points)** Justifiez brièvement cette dernière définition récursive.

Réponse :

2. **(2 points)** Sachant qu'une astuce de calcul (non décrite ici) permet de calculer $\mathrm{SDC}(i,j)$ en $\Theta(1)$, quelle serait, en fonction de G et n, la complexité d'un algorithme calculant $P(G,n)$ par programmation dynamique ?

Réponse :

3. **(2 points)** Une fois que l'on possède un algorithme de programmation dynamique pour calculer $P(k,j)$, comment le modifier pour obtenir $c_1, c_2, \ldots, c_G$?

Réponse :

Recherche ternaire (4 points)

L'algorithme de recherche ternaire dans un tableau trié fonctionne comme celui de la recherche binaire (ou dichotomique) mais divise le problème en trois plutôt qu'en deux.

1. **(2 points)** Proposez un algorithme récursif de recherche ternaire.

Réponse :

2. **(1 point)** Calculez et justifiez la complexité de cet algorithme en fonction de la taille n du tableau.

Réponse :

3. **(1 point)** Est-il plus intéressant de faire une recherche binaire ou ternaire ?

Réponse :