

N° D'ANONYMAT:



ANNÉE UNIVERSITAIRE 2017-2018  
SESSION 2 DE PRINTEMPS

**Parcours/Étape:** L3 Informatique      **Code UE:** 4TIN602U  
**Épreuve:** Techniques Algorithmiques et Programmation  
**Date:** 07/06/2018      **Heure:** 11h30      **Durée:** 1h30  
Documents: une seule feuille A4 recto-verso autorisée.  
Épreuve de M. Cyril GAVOILLE

université  
de **BORDEAUX**

Collège Sciences  
et Technologie

RÉPONDRE DIRECTEMENT SUR LE SUJET  
QUI EST À RENDRE DANS LA FEUILLE DOUBLE D'EXAMEN

## Dichotomie

On considère un tableau  $A$  de  $n$  valeurs réelles comprises dans l'intervalle  $[0, 1]$ . L'objectif est de déterminer l'indice  $k$  d'un minimum local dans  $A$ , c'est-à-dire tel que  $A[k] \leq A[k+1]$  et  $A[k] \geq A[k-1]$ . Pour simplifier, on suppose que  $A[0] = A[n-1] = 1$ .

**Question 1.** Donner le code  $C$  d'un algorithme récursif permettant de renvoyer l'indice d'un minimum local, supposé exister, pour un tableau  $A$  compris entre les indices  $i$  et  $j$  (bornes comprises). On supposera que les indices vérifient  $0 < i, j < n-1$ .

**Question 2.** Donner l'équation de récurrence que doit vérifier la complexité en temps  $T(n)$  de votre algorithme, où  $n$  représente la taille de l'intervalle de la recherche, soit  $n = j - i + 1$ .

**Question 3.** Donner la valeur asymptotique de  $T(n)$ . Justifiez.

# Voyageur de commerce

**Question 4.** Donner la formulation du problème du voyageur de commerce vu en cours.

Dans la suite, on notera  $\text{OPT}(V, d)$  la valeur de la solution optimale d'une instance  $(V, d)$  du voyageur de commerce. On supposera que  $V = \{v_0, \dots, v_{n-1}\}$  et on notera  $V^* = V \setminus \{v_{n-1}\}$  qui est l'ensemble  $V$  sans le dernier point. On définit alors la variable  $D(t, S)$ , pour tout sous-ensemble de points  $S \subseteq V^*$  et tout point  $t \in S$ , comme la longueur d'un chemin allant de  $v_{n-1}$  à  $t$  et qui visite tous (et seulement) les points de  $S$ .

**Question 5.** Exprimer  $\text{OPT}(V, d)$  en fonction d'un minimum impliquant la variable  $D$ .

On a vu dans le cours la formule de récurrence suivante pour la variable  $D(t, S)$ , pour tout  $S \subseteq V^*$  et tout  $t \in S$  (on rappelle que  $|S|$  est la cardinalité de  $S$ ) :

$$D(t, S) = \begin{cases} d(v_{n-1}, t) & \text{si } |S| = 1 \\ \min_{x \in S \setminus \{t\}} \{D(x, S \setminus \{t\}) + d(x, t)\} & \text{si } |S| > 1 \end{cases}$$

**Question 6.** Basée sur ces équations, donner le code C récursif d'une fonction `TSP_rec(t,S)` renvoyant  $D(v_t, S)$ . Votre fonction pourrait commencer par :

```
double TSP_rec(int t, set S){
    if(set_card(S)==1) return d(V[n-1],V[t]);
    ...
}
```

Vous supposerez la fonction de distance `d()`, le tableau de points `V[]`, l'entier `n` et le type `set` (ensemble) comme déjà définis, et aussi comme connues les opérations suivantes associées au type `set` : `int set_card(set S)` (renvoie  $|S|$ ), `set set_minus(set S, int t)` (renvoie  $S \setminus \{v_t\}$ ), `int set_in(int x, set T)` (teste si  $v_x \in T$ ).

**Question 7.** Donner et justifiez la complexité de votre fonction `TSP_rec(t,S)` en fonction de  $|S|$  et

$n = |V|$ . Pour simplifier, vous pourrez supposer que les fonctions associées au type `set` prennent toutes un temps constant.

**Question 8.** Dédurre des questions précédentes la complexité d'un algorithme renvoyant  $\text{OPT}(V, d)$  en fonction de  $n = |V|$ . Justifiez.

## Somme sur deux tableaux

On cherche un algorithme qui prend comme paramètre un nombre  $s$  et une paire de tableaux  $(A, B)$  chacun de  $n$  nombres (réels) et qui détermine si oui ou non il existe deux indices  $i, j$  tels que  $s = A[i] + B[j]$ . Notez bien que les tableaux ne sont pas forcément triés.

**Question 9.** Donner le code `C` d'un algorithme naïf pour résoudre ce problème, et donner sa complexité en temps en fonction de  $n$ . Justifiez.

On décide d'appliquer la méthode « diviser-pour-régner ». Pour cela on commence par découper  $A$  en deux tableaux. Le premier,  $A_1$ , contient tous les éléments de  $A$  qui sont  $\leq s/2$ . Le second,  $A_2$ , contient tous les éléments de  $A$  qui sont  $> s/2$  auxquels on a retranché  $s/2$ . Par exemple, si  $A = \{-4, 17.5, 1, 6\}$  et  $s = 5$  alors  $A_1 = \{-4, 1\}$  et  $A_2 = \{17.5 - s/2, 6 - s/2\} = \{15, 3.5\}$ . On découpe  $B$  en  $B_1$  et  $B_2$  de la même manière. On peut alors résoudre récursivement le problème sur les paires de tableaux  $(A_1, B_2)$  et  $(B_1, A_2)$  avec une valeur adéquate du paramètre  $s$ .

Pour simplifier, on fera l'hypothèse d'un découpage équilibré, c'est-à-dire qu'à chaque découpe la taille des tableaux est divisée par deux.

**Question 10.** Donner un algorithme récursif (le principe de l'algorithme suffit) basé sur la méthode décrite ci-dessus sous l'hypothèse d'un découpage équilibré.

**Question 11.** *Donner l'équation de récurrence que doit vérifier la complexité en temps  $T(n)$  de votre algorithme. Justifiez.*

**Question 12.** *Donner la valeur asymptotique de  $T(n)$ .*

**Question 13.** *Proposer un autre algorithme tout aussi efficace (le principe suffit), ainsi que sa complexité en temps, permettant de se passer de l'hypothèse d'un découpage équilibré.*