N° d'anonymat:

Année Universitaire 2018-2019
Session 2 de printemps

Parcours/Étape: L3 Informatique **Code UE:** 4TIN602U
Épreuve: Techniques Algorithmiques et Programmation
Date: 02/06/2019 **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

Solution. Total des points : $14 \times 3 = 42$ pts.

Questions de cours

Question 1. *Donnez un exemple de problème, que vous devrez définir le plus précisément possible, qui peut être résolu efficacement par la méthode* diviser pour régner.

Solution. [3 pts] La paire de points la plus proche, c'est-à-dire le problème consitant à trouver, dans un ensemble V de n points du plan, les deux points dont la distance est la plus petite.

Question 2. *Donnez les étapes principales d'un algorithme résolvant ce problème par la méthode* diviser pour régner.

Solution. [3 pts] On coupe l'ensemble V en deux sous-ensemble A, B l'ensemble des points de même taille (à un près), selon leurs abscisses. On résout récursivement le problème dans A et B, et on note δ la plus petite des deux valeurs obtenues. On peut alors trouver les deux points les plus proches entre A et B à l'aide d'un parcours de bande de largeur 2δ située à cheval sur A et B. Si les ensembles sont triés selon l'ordonnées, cette dernière étape peut être réalisée efficacement en temps linéaire.

Le plus grand bloc pair

On considère le problème consistant à compter le nombre de mots binaires de longueur n dont le plus grand bloc de 1 consécutifs est de longueur paire. Et on notera $f(n)$ ce nombre.

Par exemple, pour $n = 5$, seuls les 14 mots suivants de longueur n ont leur plus grand bloc de 1 de longueur paire (un mot sans bloc de 1 étant considéré un bloc de longueur paire) :

$$00000,\ 00011,\ 00110,\ 01011,\ 01100,\ 01101,\ 01111,$$
$$10011,\ 10110,\ 11000,\ 11001,\ 11010,\ 11011,\ 11110$$

Ainsi $f(5) = 14$.

Question 3. *Donnez le principe d'un algorithme qui calculerait $f(n)$ selon une recherche exhaustive.*

SOLUTION. [3 pts] Lister tous les mots binaires de longueur n, et pour chacun vérifier si la propriété est satisfaite. Renvoyer à la fin le nombre de fois que la propriété est satisfaite.

On va représenter un mot binaire S de longueur n par un tableau `int S[n]` contenant seulement les entiers `0` et `1`.

Question 4. *Donnez le code d'une fonction* `int bloc(int S[],int n)` *renvoyant la longueur du plus grand bloc de* 1 *consécutifs dans* `S`*.*

SOLUTION. [3 pts]

```
int bloc(int S[],int n){
  int m,c,i; // c=longueur du bloc courant, m=longueur max
  for(i=m=c=0;i<n;i++){ // on parcourt S
    if(S[i]){ c++; if(c>m) m=c; }
    else c=0;
  }
  return m;
}
```

Question 5. *Quelle est la complexité de votre fonction* `bloc()` *? Justifiez.*

SOLUTION. [3 pts] $O(n)$ car il s'agit d'un simple parcours des bits de `S`.

On suppose donnée une fonction `bool inc(int S[],int n)` permettant d'incrémenter `S` vu ici comme un compteur binaire. Dit autrement, la fonction `inc(S,n)` modifie `S` comme si on ajoutait 1 à l'entier dont la représentation binaire est donnée par `S`. La fonction renvoie `false` si et seulement si, avant l'appel à `inc(S,n)`, `S` n'était composé que de 1.

Par exemple, si `S[]={0,1,1}`, l'appel à `inc(S,3)` modifiera `S` en `{1,0,0}`, tout en renvoyant `true`. Puis, en réappliquant trois fois `inc(S,3)`, on obtiendra successivement `{1,0,1}`, `{1,1,0}` et `{1,1,1}`. Si on applique une dernière fois `inc(S,3)`, la fonction renverra `false`.

Le code de cette fonction pourrait ressembler à ceci :

```
bool inc(int S[],int n){
  for(int i=n-1;i>=0;i--){
    S[i]=(S[i]+1)%2;
    if(S[i]) return true;
  }
  return false;
}
```

Question 6. *Donnez le code d'une fonction* `int f(int n)` *renvoyant le nombre* $f(n)$.

SOLUTION. [3 pts]

```
int f(int n){
  int S[n];
  for(int i=0;i<n;i++) S[i]=0;
  int c=0; // pour compter
  do if(bloc(S,n)%2==0) c++; while(inc(S,n));
  return c;
}
```

Question 7. *Quelle est la complexité de votre fonction* `f()` *? Justifiez.*

SOLUTION. [3 pts] C'est $O(n \cdot 2^n)$ puisque la boucle `do ... while` est répétée 2^n fois, le nombre de mots binaire de longueur n, et que chaque boucle coûte $O(n)$, pour `bloc(S,n)` et pour `inc(S,n)`.

Question 8. *Diriez vous que la complexité de votre fonction* `f(n)` *est :*

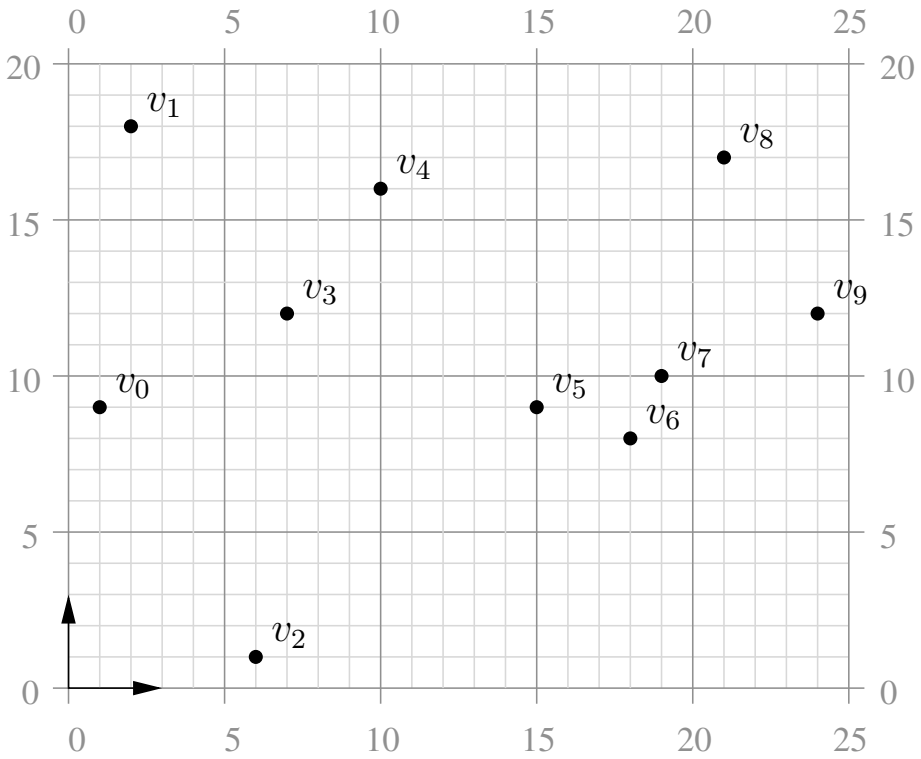
- $\boxed{1}$ *logarithmique,*
- $\boxed{2}$ *linéaire,*
- $\boxed{3}$ *quadratique,*
- $\boxed{4}$ *exponentielle, ou*
- $\boxed{5}$ *doublement exponentielle.*

Justifiez.

SOLUTION. [3 pts] La complexité doit être exprimée en fonction de la taille de l'entrée. Notons N cette taille (exprimée en bits). Si on écrit $n-1$ en binaire par exemple, on aura alors une description de n sur $N = \lceil \log_2 n \rceil$ bits d'après une proposition du cours. Et donc $\log_2 n \leqslant N < (\log_2 n) + 1$ ou encore $n \in [2^{N-1}, 2^N[$. Dit autrement $n = 2^{\Theta(N)}$. La réponse est donc $\boxed{5}$ car la complexité est $n \cdot 2^n = 2^{\Theta(n)} = 2^{\Theta(2^N)} = 2^{2^{\Theta(N)}}$ ce qui est doublement exponentielle en la taille N de l'entrée.

MST et voyageur de commerce

On considère un ensemble de 10 points $V = \{v_0, v_1, \ldots, v_9\}$ placés sur une grille rectangulaire comme ci-après. La distance entre deux points de V est la distance dîte de *Manhattan*, c'est-à-dire la longueur d'un plus court chemin qui suit la grille. Plus précisément, la distance entre les points $a = (x_a, y_a)$ et $b = (x_b, y_b)$ est $d(a, b) = |x_a - x_b| + |y_a - y_b|$. Par exemple, $d(v_0, v_1) = |1-2| + |9-18| = 10$. Notez bien que toutes les distances sont entières.



Question 9. *Calculez un arbre T de poids minimum couvrant V, le poids de chaque arête $v_i - v_j$ correspondant à sa longueur, c'est-à-dire à $d(v_i, v_j)$. Vous devez dessiner votre solution directement sur la grille.*

SOLUTION. [3 pts] L'arbre, qui est unique, est l'union des deux chemins suivants : $v_8 - v_9 - v_7 - v_6 - v_5 - v_3 - v_4 - v_1$ et $v_2 - v_3 - v_0$.

Question 10. *Donnez le poids total de l'arbre T. (Vous devez expliciter la somme des poids.)*

SOLUTION. [3 pt] Poids de T : $(8 + 7 + 3 + 4 + 12 + 7 + 10) + (12 + 9) = 72$.

Question 11. *Donnez la tournée obtenue à partir d'un parcours en profondeur de T depuis le point v_0. S'il y a un choix, votre parcours devra choisir le point voisin ayant le plus petit indice. Pour cela vous devez donner l'ordre de visite des points (incluant le point de retour), ainsi que la longueur de la tournée obtenue. (Vous devez expliciter la somme des distances.)*

SOLUTION. [3 pts] Tournée : $v_0 \to v_3 \to v_2 \to v_4 \to v_1 \to v_5 \to v_6 \to v_7 \to v_9 \to v_8 \to v_0$
Longueur : $9 + 12 + 19 + 10 + 22 + 4 + 3 + 7 + 8 + 28 = 122$.

Question 12. *Proposez une tournée de longueur strictement inférieure obtenue en décroisant deux segments. Donnez simplement les segments à décroisez ainsi que le gain obtenu.*

SOLUTION. [3 pts] En décroisant $v_0 \to v_8$ et $v_3 \to v_2$, on obtient le gain : $(v_0 \to v_8 + v_3 \to v_2) - (v_0 \to v_2 + v_8 \to v_3) = (12 + 28) - (13 + 19) = 18$.

Question 13. *Proposez une tournée obtenue par l'algorithme glouton en partant du point v_4. (Donnez l'ordre de visite, puis la longueur de la tournée en explicitant la somme des distances.)*

SOLUTION. [3 pts] Tournée : $v_4 \to v_3 \to v_0 \to v_1 \to v_8 \to v_9 \to v_7 \to v_6 \to v_5 \to v_2 \to v_4$
Longueur : $7 + 9 + 10 + 20 + 8 + 7 + 3 + 4 + 17 + 19 = 104$.

Une tournée est *bitonique* si la suite des abscisses des points rencontrés est d'abord décroissante puis croissante. Dit autrement, c'est une tournée pour laquelle on peut partir du point le plus à droite et aller jusqu'au point le plus à gauche en suivant des points dont les abscisses ne font que décroître, puis revenir au point de départ en suivant des points d'abscisses croissantes.

Question 14. *Proposez une tournée bitonique pour V sans croisement ainsi que sa longueur. (Donnez l'ordre de visite, puis la longueur de la tournée en explicitant la somme des distances.)*

SOLUTION. [3 pts] Tournée : $v_9 \to v_8 \to v_4 \to v_3 \to v_1 \to v_0 \to v_2 \to v_4 \to v_6 \to v_7 \to v_9$
Longueur : $8 + 12 + 7 + 11 + 10 + 13 + 17 + 4 + 3 + 7 = 92$.

FIN.