RODRIGUES Ruben 107

IUT Paris Descartes – Dép. Informatique – Analyse et Méthodes numériques S2 – DST du 7 juin 2017

Sans calculatrice, sans portable, sans document. – Durée : 1h30min. Il est demandé de justifier soigneusement vos réponses.

3,25

Exercice 1 (4 points) ***Dans cet exercice, vous répondrez directement sur le sujet. Pour chaque question, une ou plusieurs réponse(s) sont possibles. Une réponse juste rapporte 0.5 points, une réponse fausse enlève 0.25 points.***

1. Le nombre de multiplications de deux nombres réels effectuées dans la multiplication de deux matrices de taille n est :

   ☐ $3n$ ☒ n^3 ☐ n^2 ☐ $2n$

2. On note $C(n)$ le nombre de multiplications de nombres réels nécessaires dans le calcul direct (sans optimisation) de la puissance 25-ième d'une matrice carrée de taille n. Parmi les affirmations suivantes, données au voisinage de $+\infty$, cochez celles qui sont vraies :

   ☐ $C(n) = \Theta(n)$ ☒ $C(n) = \Theta(n^3)$ ☐ $C(n) = \Theta(n^a)$, $2 < a < 3$ ☐ $C(n) \sim n^{25}$
   ☒ $C(n) \sim 25n^3$ ☐ $C(n) \sim n^3$

3. A l'heure actuelle, les meilleurs algorithmes optimisant le produit de deux matrices de taille n ont une complexité en $\Theta(\log(n))$:

   ☒ Vrai ☐ Faux

4. Dans Scilab, la commande `plot2d([0 5],[0 6])` permet, dans une fenêtre graphique, de :

   ☐ placer deux points de coordonnées $(0;5)$ et $(0;6)$
   ☐ représenter les vecteurs de coordonnées $\begin{pmatrix}0\\5\end{pmatrix}$ et $\begin{pmatrix}0\\6\end{pmatrix}$ à partir de l'origine du repère
   ☒ représenter le vecteur de coordonnées $\begin{pmatrix}5\\6\end{pmatrix}$ à partir de l'origine du repère
   ☐ représenter un segment de droite entre les points de coordonnées $(0;5)$ et $(0;6)$

5. Laquelle de ces commandes Scilab permet de renvoyer dans la variable B la puissance 30-ième d'une matrice A en un temps minimal ?

   ☐ `B=A∧30;`
   ☐ `[P,D]=spec(A); B=P*D∧30*inv(P);`
   ☒ `[P,D]=spec(A);vp=diag(D); vpk=vp.∧30; Dk=diag(vpk); B=P*Dk*inv(P);`

6. On souhaite résoudre, par une méthode numérique, une équation du type $f(x) = 0$ dans un intervalle où f est continue et ne s'annule qu'une seule fois. On ne connaît pas la convexité de f. Quelle est la méthode la plus appropriée ?

   ☒ Méthode par dichotomie ☐ Méthode de la Sécante

7. Soit A une matrice carrée de taille 2 et X un vecteur non nul de $\mathbb{R}^2$. Soit λ_0 une valeur propre dominante de A (c'est à dire une valeur propre dont la valeur absolue est la plus grande) et b_0 un vecteur propre associé à λ_0.
   Quand k devient grand :

   ☐ A^kX se rapproche de λ_0
   ☒ la direction de A^kX se rapproche de la direction de b_0, quel que soit X non nul
   ☐ la direction de A^kX ne converge que si X est orthogonal à b_0

6/7

Exercice 2 (9 points)

On rappelle le principe de l'algorithme de dichotomie pour déterminer une valeur approchée de l'unique solution de $f(x) = 0$ dans l'intervalle $[a, b]$ avec $a < b$.

Deux suites (u_k) et (v_k) sont définies comme suit :

- $u_0 = a, \quad v_0 = b$
- Si $f(u_k) \times f(\frac{u_k+v_k}{2}) < 0$, alors $u_{k+1} = u_k$ et $v_{k+1} = \frac{u_k+v_k}{2}$
- Si $f(u_k) \times f(\frac{u_k+v_k}{2}) > 0$, alors $u_{k+1} = \frac{u_k+v_k}{2}$ et $v_{k+1} = v_k$

On itère le processus jusqu'à ce que la différence $|v_k - u_k|$ soit inférieure à la précision ε demandée. N'importe quelle valeur de l'intervalle $[u_k, v_k]$ fournit alors une valeur approchée à ε près de la solution α recherchée.

On cherche à montrer que pour un intervalle de largeur $n > 2$ et pour une précision demandée $\varepsilon = 1$, la complexité de cet algorithme s'exprime en $\Theta(\ln(n))$.

6

1. Compléter très lisiblement le calcul "à trous" ci-dessous ainsi que les justifications quand elles sont demandées.

   Soit k un entier naturel. On va calculer $|u_{k+1} - v_{k+1}|$ en distinguant deux cas :

   - soit $|u_{k+1} - v_{k+1}| = \left|u_k - \frac{u_k+v_k}{2}\right| = \left|\frac{1}{2}u_k - \frac{1}{2}v_k\right|$

   - soit $|u_{k+1} - v_{k+1}| = \left|\frac{u_k+v_k}{2} - v_k\right| = \left|\frac{1}{2}u_k - \frac{1}{2}v_k\right|$

   Ainsi dans tous les cas $|u_{k+1} - v_{k+1}| = \frac{1}{2} \times |u_k - v_k|$

   $(|u_k - v_k|)_{k\in\mathbb{N}}$ est donc une suite géométrique de raison $q = \frac{1}{2}$

   Et on a alors $\quad |u_k - v_k| = \left(\frac{1}{2}\right)^k \times |u_0 - v_0| = \frac{1}{2^k} \times |u_0 - v_0|$

   L'intervalle $[a, b]$ étant de largeur n, on a donc en fonction de n :

   $$|u_k - v_k| = \cancel{\frac{1}{2^k} \times |u_0 - v_0|} \quad \boxed{\frac{n}{2^k}}$$

   Si la précision demandée est $\varepsilon = 1$, alors on doit itérer le processus jusqu'à ce que

   $$|u_k - v_k| < 1$$

   Donc :

   $$\frac{n}{2^k} < 1$$
   $$\ln n < \ln 2^k$$
   $$\frac{\ln n}{\ln 2} < k$$

   $$\boxed{k > \cancel{\frac{\ln(|u_0 - v_0|)}{\ln 2}}}$$

7/7

2. On propose ci-dessous en pseudo-langage, un algorithme de recherche dichotomique d'un nombre entier x dans la liste des entiers de 1 à n.

    INITIALISATION :
        $u = 1$
        $v = n$
    WHILE $|u - v| > 1$: 1
        $m = E\left(\frac{u+v}{2}\right)$ 2,3 $\qquad$ //(*où E est la fonction partie entière*)
            IF $m = x$ THEN Return m
            ELSE
                IF $x > m$ THEN $u = m$ 4
                ELSE $v = m$ 5
                END (IF)
            END (IF)
    END (WHILE)
    Return m

(a) Combien d'instructions sont exécutées à chaque passage dans la boucle WHILE ? (On considère comme "instructions" les instructions de calcul et/ou d'affectation). Justifier.

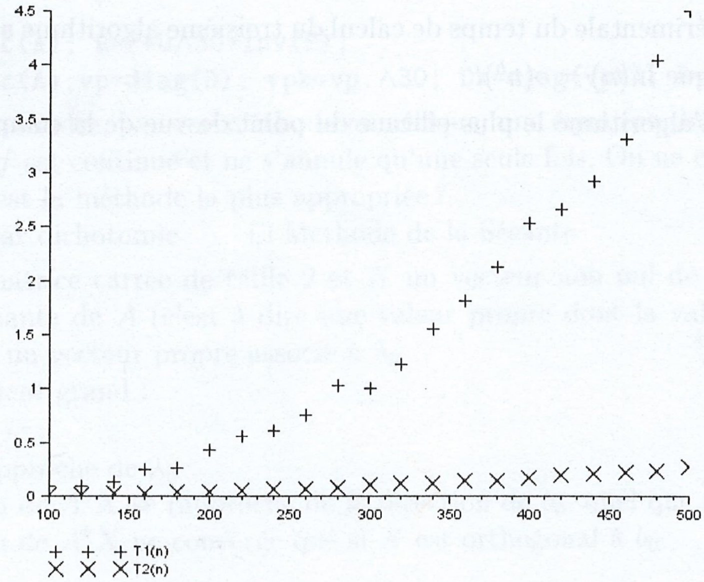
(b) En vous aidant de la question 1), quel est alors le nombre d'instructions $c(n)$ effectuées dans cet algorithme dans le pire des cas ?

(c) En déduire le type de complexité (en termes de nombre d'instructions) de l'algorithme de dichotomie. On donnera la réponse en $\Theta(g(n))$.

Exercice 3 (7 points)

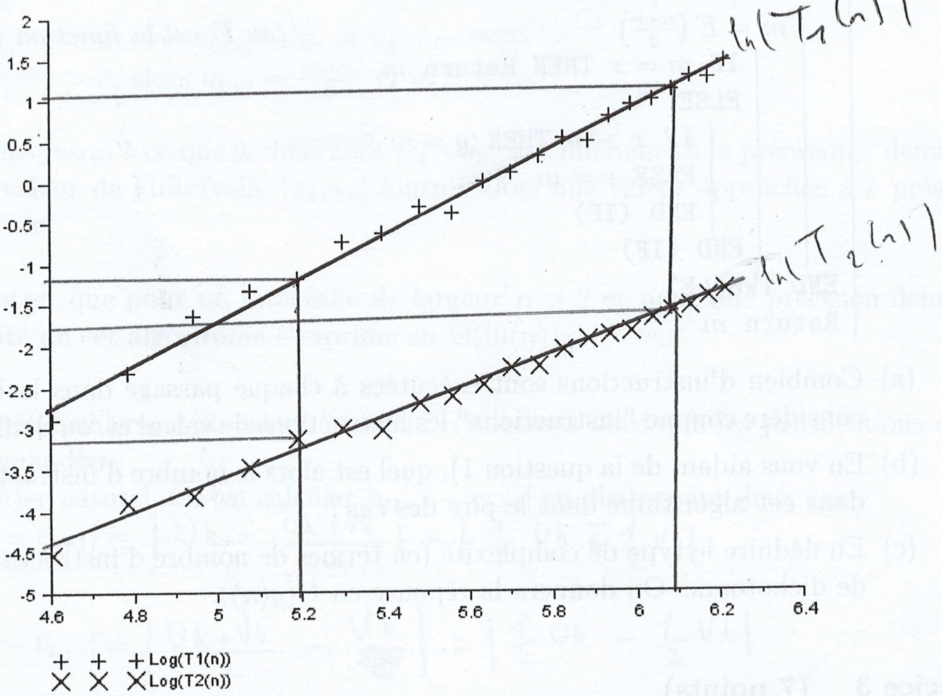
On considère trois algorithmes numérotés de 1 à 3 effectuant une même tâche sur une donnée de taille n. On s'intéresse au temps d'exécution $Ti(n)$ de chaque algorithme i en fonction de la taille n de l'argument d'entrée.

On effectue des simulations pour les algorithmes 1 et 2, avec des entrées aléatoires de taille n, pour n allant de 100 à 500. On obtient le graphique suivant :



1. Quel(s) type(s) de croissance asymptotique peut-on envisager pour les temps $T1(n)$ et $T2(n)$?

2. (a) On trace les courbes
$C_1 : \ln(n) \mapsto \ln(T1(n))$ et $C_2 : \ln(n) \mapsto \ln(T2(n))$ ci-dessous.



OK

On peut considérer qu'on obtient quasiment des droites.
En utilisant le graphique, estimez les valeurs approchées des pentes (ou coefficients directeurs) a et b de chacune des courbes C_1 et C_2. (on laissera en évidence les éventuels traits de construction sur le graphique et les éventuels calculs).
Indication : On devrait trouver $b > 1$.

(b) En déduire que $T1(n) = C \times n^a$ et $T2(n) = D \times n^b$ où C et D sont des constantes positives. (On ne demande pas de déterminer C et D.)

(c) Montrer qu'alors $T1(n) = O(n^a)$ et $T2(n) = O(n^b)$.

(d) De quel type sont les complexités temporelles $T1(n)$ et $T2(n)$?

3. Une étude expérimentale du temps de calcul du troisième algorithme a fourni $T3(n) = \ln(n)$.

(a) Montrer que $\ln(n) = o(n^b)$.

(b) Quel est l'algorithme le plus efficace du point de vue de la complexité temporelle ?

UNIVERSITÉ PARIS DESCARTES
IUT
DÉPARTEMENT INFORMATIQUE

NOM : RODRIGUES (en capitales)
Prénom : Ruben
Écrire très lisiblement

DISCIPLINE : Analyse
Date de l'épreuve : 07/06/2017
Année : 1A Groupe : 107

NOTE DE 0 À 20	APPRÉCIATIONS
18	3,25 + 7,5 + 7 Bravo !

Ne rien écrire dans cette marge

Exercice 2 :

2) a)

À chaque passage dans la boucle while on effectue les instructions suivantes :

		nbr d'instructions
$\lvert u-v \rvert > 1$	(condition de la boucle) calcul	1
$m = E\left(\frac{u+v}{2}\right)$	Calcul + affectation	2,3

On peut s'arrêter ici ou continuer :

$u = m$	affectation	4
ou		
$v = m$	affectation	4

À chaque passage dans la boucle on effectue au mieux trois instructions, au pire quatre instructions

1

2) b) D'après la question a) dans le pire des cas on effectue 4 instructions,

soit C(n) la fonction représentative du nombre d'instructions effectuées par cette algorithme dans le pire des cas avec n la ~~nombre d'itération~~ :
borne maximale de l'intervalle

$$C(n) = 4k + 3$$

+3 représentant les deux affectations de l'initialisation et le dernier calcul de la condition du while.

k le nombre d'itérations:

$$\cancel{\text{Donc}}\ k \cancel{>} \cancel{<} \frac{\ln(|\overset{n}{\cancel{1-n}}|)}{\ln 2} \qquad U_0 = 1,\ V_0 = n$$

$$k \cancel{<} \log_2(|\overset{n}{\cancel{1-n}}|) \qquad \frac{\ln x}{\ln 2} = \log_2(x)$$

0,5

$$C(n) = 4 \log_2(|\overset{n}{\cancel{1-n}}|) + 3$$

Exercice 3 :

1. $T_1(n)$ semble avoir une croissance exponentielle ou polynomiale tandis que $T_2(n)$ semble avoir une croissance linéaire ou polynomiale.

1

2.a) On calcule les coefficients directeurs des pentes : $a = \dfrac{y_B - y_A}{x_B - x_A}$

$$a \approx \frac{1 + 1,5}{6,1 - 5,2} \approx 2,9$$

$$b \approx \frac{-1,5 + 3}{6,1 - 5,2} \approx 1,7$$

1

Les équations de droites sont donc de la forme $y = ax + b$

Pour $\ln(T_1(n))$:

$$y_1 = a \ln(n) + c$$

Pour $\ln(T_2(n))$:

$$y_2 = b \ln(n) + d$$

b) $y_1 = \ln(T_1(n))$

$$\ln(T_1(n)) = a\ln(n) + c$$

$$e^{\ln(T_1(n))} = e^{a\ln(n) + c}$$

$$T_1(n) = e^{a \ln n} * e^c$$

$$T_1(n) = n^a \times e^c \qquad e^c = C \qquad e^c > 0, C \in$$

$$T_1(n) = n^a \times C$$

3

De même pour y_2 :

$y_2 = b\ln(n) + d = \ln(T_2(n))$

$e^{\ln(T_2(n))} = e^{b\ln(n)+d}$

$T_2(n) = e^{b\ln(n)} \times e^{d}$ $\quad e^{d} = D \quad e^{d} > 0, d \in \mathbb{R}$

1 $T_2(n) = n^{b} \times D$

c) $\forall n \in \mathbb{N}$,

$T_1(n) = O(n^{a}) \iff \left|\frac{T_1(n)}{n^{a}}\right| \leq K \quad K \in \mathbb{R}^{+}$

$\iff \left|\frac{n^{a} \times C}{n^{a}}\right| \leq K$

$\lim\limits_{n\to+\infty} \frac{n^{a} \times C}{n^{a}} = C \quad C \in \mathbb{R}$

On a donc bien $T_1(n) = O(n^{a})$

De même, $T_2(n) = O(n^{b}) \iff \forall n \in \mathbb{N} \left|\frac{T_2(n)}{n^{b}}\right| \leq K_1 \quad K_1 \in \mathbb{R}^{+}$

$\lim\limits_{n\to+\infty} \frac{T_2(n)}{n^{b}} = \lim \frac{n^{b} \times D}{n^{b}} = \lim\limits_{n\to+\infty} D$

D constante, $D \in \mathbb{R}^{+}$

1 On a donc $T_2(n) = O(n^{b})$

d) Les complexités temporelles de $T_1(n)$ et $T_2(n)$ sont donc polynomiales. B

1

3. $T_3(n) = \ln(n)$

a) $\ln(n) = O(n^{b})$ si $\lim\limits_{n\to+\infty} \frac{\ln(n)}{n^{b}} = 0$

or on sait que $\lim\limits_{A\to+\infty} \frac{\ln(A)^{c}}{A^{b}} = 0 \quad b, c \in \mathbb{R}^{+*}$

4/7

UNIVERSITÉ
PARIS DESCARTES
IUT
DÉPARTEMENT INFORMATIQUE

DISCIPLINE : Analyse
Date de l'épreuve : 07/06/2017
Année : 1A Groupe : 107

Écrire très lisiblement
NOM (en capitales) : RODRIGUES
Prénom : Ruben

NOTE DE 0 À 20	APPRÉCIATIONS

Ne rien écrire dans cette marge

Exercice 3 : (suite) :

3. a) donc $\lim\limits_{n \to +\infty} \frac{\ln(n)}{n^b} = 0$

1 donc $\ln(n) = o(n^b)$ est vrai

b) Du point de vue de la complexité temporelle l'algorithme le plus efficace est l'algorithme n°3 ~~$T_3(n)$~~, il a une complexité de type $T_3(n) = \Theta(\ln(n))$

1

5/1