Epreuve finale 2023/2024 - Master TC - Dép. Informatique - Faculté FSEA - Université Oran1

Algorithmique Avancée et Complexité / Durée 01h40

Exercice 1 (11 pts)

Q1) (3p) Dérouler le tri par tas sur T=[10, 20, 0, 30, -10, 40].

Q2) (1p) Soit un algorithme *Algo* en $\theta(2^{n+7})$. Démontrer que *Algo* est aussi en $\theta(2^n)$.

Q3) (3p) Résoudre l'équation récurrente F_n=$F_{n-1} + F_{n-2}, F_0$=2, F_1=3.

(1p) En considérant son comportement lorsque *n* est grand, montrer que $F_n > c \times \beta^n - 1$, où c et β sont des constantes à définir.

Q4) (3p) Soient 4 algorithmes (puissA, puissB, puissC, puissD) pour calculer 2^n :

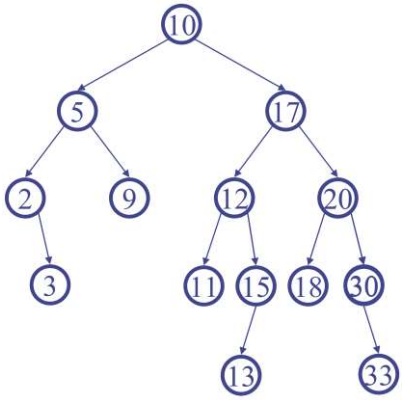
a. Formuler et résoudre l'équation récurrente de chacun des 4 algorithmes.

b. Lequel est le moins couteux ?

Algorithm puissA(n):
m = 1
for i=1 to n do
m = 2*m
endfor
return m
	Algorithm puissB(n):
	if n=0 then
	return 1
	else
	return puissB(n-1) + puissB(n-1)
	endif

Algorithm puissC(n):
if n=0 then
return 1
else
return 2*puissC(n-1)
endif
	Algorithm puissD(n):
	return puissD2(2, n)
	Algorithm puissD2(x, n):
	if n=0 then return 1
	else if (n est pair) then return puissD2(x*x, n/2)
	else return x*puissD2(x*x, (n-1)/2)
	endif

Exercice 2 (6 pts) Soit la structure AVL d'un arbre binaire de recherche A.



Q1) (2p) Illustrer toutes les étapes pour supprimer la clé 5 et conserver la structure AVL ?

Q2) Soit *A* un arbre AVL ayant n sommets et de hauteur h.

a. (1.5p) Démontrer que $h + 1 \geq log_2(1 + n)$.

b. (1.5p) Soit u_h le nombre minimal de sommets dans l'arbre AVL *A*.
Montrer que u_h=$u_{h-1} + u_{h-2} + a, u_0$=b, u_1=c, où a, b et c sont des constantes à déterminer.
En posant u_h=$F_h - 1$, en exploitant la solution de l'exercice 1, montrer que $h + 1 \leq R \times log_2(2 + n)$, où *R* est une constante à déterminer.

c. (1p) Que peut-on déduire des deux questions précédentes (a. et b.) ?

Exercice 3 (3 pts)

Soit une liste de *n* objets *E* = [1… *n*] de poids respectifs *P*=[p_1, …, p_n]. On veut déterminer un nombre *m* de boites, où chaque boite peut supporter un poids *C* (où $p_1 \leq C$, …, $p_n \leq C$). Nous voulons placer les *n* objets dans un minimum *m* de boites. Proposer un algorithme glouton pour résoudre ce problème, en déterminant le nombre minimal de boites *m*, et le placement de chaque objet.

Exemple : *P*=[9, 8, 7, 6, 5, 4, 3, 2]. *C*=11. Solution possible, 4 boites 

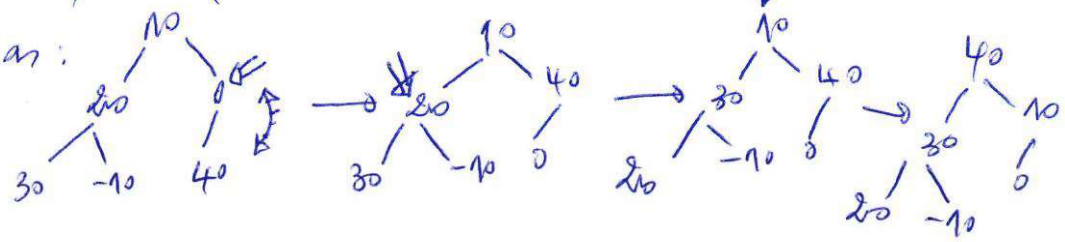
.

Exercice 1 (11 pts)

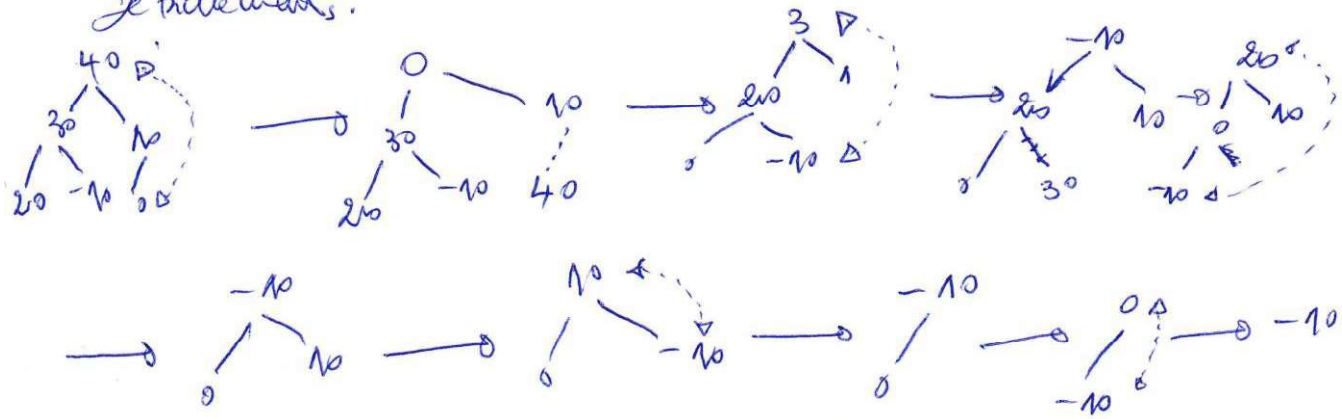
+3

Q1) Tri par Tas :

a) Construction :



b) Retirements :



+1

Q2) $\Theta(2^{h+7}) = \Theta(2^7 * 2^h)$

$= \Theta(2^h)$ car multiplier par une constante ne change pas la complexité.

+3 Q3)

$F_h = F_{h-1} + F_{h-2}$ en ajoutant $+1$ dans les deux membres ci-haut

$$F_0 = 2, F_1 = 3$$

Cette dernière équation est une équation récurrente linéaire d'ordre 2 à coefficients constants. On peut donc appliquer le théorème 1. On obtient la solution :

$$F_h = c_1\beta_1^h + c_2\beta_2^h$$

où $\beta_1 = \frac{1+\sqrt{5}}{2} \simeq 1.62$, $c_1 = \frac{5+2\sqrt{5}}{5} \simeq 1.89$,
$\beta_2 = \frac{1-\sqrt{5}}{2} \simeq -0.62$ et $c_2 = \frac{5-2\sqrt{5}}{5} \simeq 0.11$.

+1 On a $-1 \leq \beta_2^h$, d'où $c_2\beta_2^h \geq -c_2 \geq -0.11 \geq -1$. On peut poser

$$c_1\beta_1^h + c_2\beta_2^h \quad > c_1\beta_1^h - 1$$

Q4)

Soit $C(i)$ la complexité de l'algorithme :

+0.6 a) Equation récurrente de PuissA :

$$C(i) = \begin{cases} 1 & \text{si } i = 0 \\ C(i-1)+1 & \text{si } i > 0 \end{cases}$$

$$\Rightarrow C(i) = \sum_{i=0}^{n} 1 = n+1$$

$$\Rightarrow C(i) = \Theta(n)$$

b) Equation récurrente de PuissB :

+0.6

$$C(n) = \begin{cases} 1 & \text{si } n = 0 \\ C(n-1) + C(n-1) + 1 & \text{si } n > 0 \end{cases}$$

$$\Rightarrow C(n) = \begin{cases} 1 & \text{si } n = 0 \\ 2 \cdot C(n-1) + 1 & \text{si } n > 0 \end{cases}$$

Soit $F(n) = C(n) + a$

$F(n) = 2 \cdot F(n-1)$

$\Rightarrow C(n) = 2 \cdot C(n-1) + a$

par analogie : $a = 1$

$F(n) = C(n) + 1$

$F(n) = 2 \cdot F(n-1)$

$F(0) = 2$

$\Rightarrow F(n) = 2^{n+1}$

$$\Rightarrow C(n) = \prod_{i=0}^{n} 2 - 1 = 2^{n+1} - 1$$

$$\Rightarrow C(n) = \Theta(2^n)$$

c) Équation récurrente PuissC :

+0.6

$$C(n) = \begin{cases} 1 & \text{si } n = 0 \\ C(n-1) + 1 & \text{si } n > 0 \end{cases}$$

$$\Rightarrow C(n) = \sum_{n=0}^{n} 1 = n+1$$

$$\Rightarrow C(n) = \Theta(n)$$

d) Equation récurrente de PuissD :

+0.6

$$C(n) = \begin{cases} 1 & \text{si } n = 0 \\ C(n/2) + 1 & \text{si } n > 0 \end{cases}$$

qui est l'équation récurrente de la dichotomie, d'où :

$$C(n) = \Theta(\log(n))$$

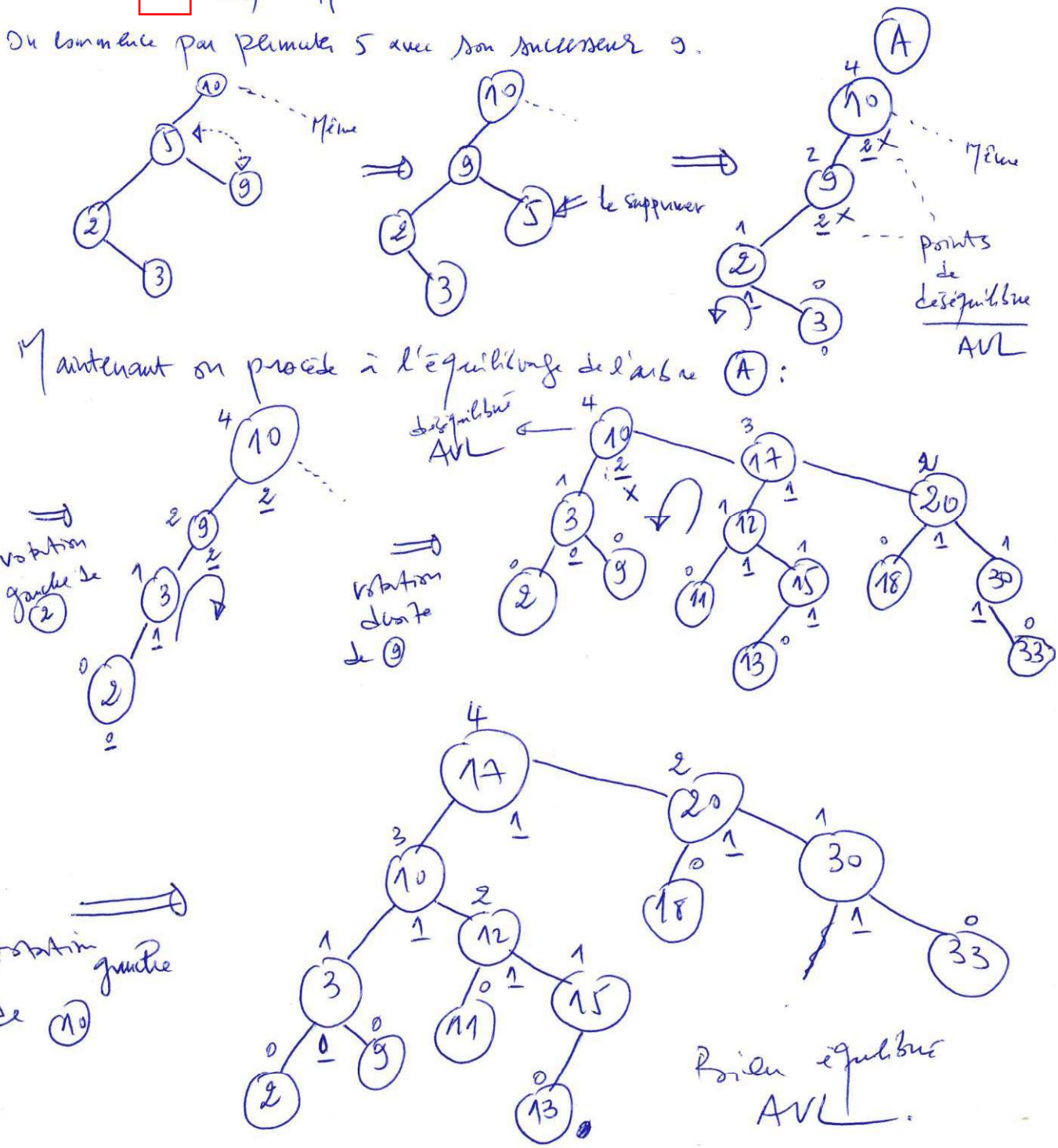
+0.6 Le moins coûteux est PuissD.

L'algorithme le plus performant est l'algorithme d).

Exercice 2 (6 pts)

+2 Q1) Suppression de la clé 5.

On commence par permuter 5 avec son successeur 9.



Q2)

+1.5 a)

Preuve On a toujours $n \leq 2^{h+1} - 1$, car le nombre de noeuds est toujours inférieur au max qu'on pourrait placer dans cet arbre. Et donc $\log_2(1+n) \leq 1 + h$.

b)

+1.5 Soit u_h le nombre minimum de sommets d'un arbre AVL de hauteur h. Alors

$u_h = u_{h-1} + u_{h-2} + 1$ car la diff des hauteur est 1 au min (sinon max...)

$$u_0 = 1, u_1 = 2$$

Posons $F_h = u_h + 1$. On a donc

$F_h = F_{h-1} + F_{h-2}$ en ajoutant $+1$ dans les deux membres ci-haut

$$F_0 = 2, F_1 = 3$$

Cette dernière équation est une équation récurrente linéaire d'ordre 2 à coefficients constants. On peut donc appliquer le théorème 1. On obtient la solution :

$$F_h = c_1\beta_1^h + c_2\beta_2^h$$

où $\beta_1 = \frac{1+\sqrt{5}}{2} \simeq 1.62$, $c_1 = \frac{5+2\sqrt{5}}{5} \simeq 1.89$,
$\beta_2 = \frac{1-\sqrt{5}}{2} \simeq -0.62$ et $c_2 = \frac{5-2\sqrt{5}}{5} \simeq 0.11$.
Et donc

$$u_h = c_1\beta_1^h + c_2\beta_2^h - 1$$

On a $-1 \leq \beta_2^h$, d'où $c_2\beta_2^h \geq -c_2 \geq -0.11 \geq -1$. On peut poser

$$c_1\beta_1^h + c_2\beta_2^h \qquad > c_1\beta_1^h - 1$$

$$h < \frac{log_2(n+2)}{\log_2 \beta_1} - \frac{\log_2 c_1}{\log_2 \beta_1} \leq \frac{log_2(n+2)}{\log_2 \beta_1} - 1.31\cdots$$

$$h \leq \frac{log_2(n+2)}{\log_2 \beta_1} - 1$$

En considérant $\leq$ et dominant $-\log_2 c_1$:

$$h + 1 \leq \frac{1}{\log_2 \beta_1} log_2(n+2)$$

+1 c) Que peut-on déduire ?
Soit A un arbre AVL ayant n sommets et de hauteur h. Alors

$$\log_2(1+n) \leq 1 + h \leq \alpha \log_2(2+n)$$

Exercice 3 +3

Algorithme MiseEnBoite (P = [P1, —, Pn], C, B[1..max], m)

Input : P = [P1, —, Pn] les poids des n objets
C = poids d'une boite

Output : B[1..max] { taille : taille de ~~chacune des~~ la boite
objets : Composition de la boite
m : nombre de boite ($m \leq max$)

```
m ← 0
for i ← 1 to n do
    j ← 1
    while (j ≤ max) and (B[j].taille + Pi > C) do
        j ← j+1
    endwhile
    if (j > max) then
        return "depassement du max de boites"
    endif
    B[j].objets ← Concatener(B[j].objets, [i])
    B[j].taille += Pi
    if (j > m) then
        m ← j
    endif
end for
```

Cet algorithme est en $O(n \times max)$ — polynomial