



Algorithmique

Cours avec 957 exercices
et 158 problèmes



INF2422 - Complexité Algorithmique
Chapitre IV : Récurrences

YOUSSOU DIENG (YDIENG@UNIV-ZIG.SN)

1

Introduction

Le temps d'exécution d'un algorithme récursif, peut souvent être décrit par une récurrence.

Une **récurrence** étant une équation/inégalité qui décrit une fonction à partir de sa valeur sur des entrées plus petites.

Exemple : Le temps d'exécution $T(n)$ du cas le plus défavorable de la procédure TRI-FUSION pouvait être décrit par la récurrence :

$$T(n) = \begin{cases} \Theta(1), & \text{si } n = 1 \\ 2T\left(\frac{n}{2}\right) + \Theta(n), & \text{si } n > 1 \end{cases}$$

dont nous avons affirmé que la solution était $T(n) = \Theta(n \lg n)$.

YOUSSOU DIENG (YDIENG@UNIV-ZIG.SN)

2

Dans cette partie, ...

Le souhait est de présenter ici une méthode pour résoudre les récurrences (obtenir des bornes asymptotiques « Θ » ou « O » pour la solution).

1. La **méthode de substitution**: on devine une borne puis on utilise une récurrence mathématique pour démontrer la validité de la conjecture.
2. La **méthode de l'arbre récursif**: On convertit la récurrence en un arbre dont les nœuds représentent les coûts induits à différents niveaux de la récursivité ; On emploie ensuite des techniques de bornage de sommation pour résoudre la récurrence.
3. La **méthode générale**: elle fournit des bornes pour les récurrences de la forme :

$$T(n) = aT\left(\frac{n}{b}\right) + f(n) \text{ où } a \geq 1, b > 1 \text{ et,}$$

$f(n)$ est une fonction donnée

YOUSSOU DIENG (YDIENG@UNIV-ZIG.SN)

3

Dans cette partie, ...

Le souhait est de présenter ici une méthode pour résoudre les récurrences (obtenir des bornes asymptotiques « Θ » ou « O » pour la solution).

1. La **méthode de substitution**: on devine une borne puis on utilise une récurrence mathématique pour démontrer la validité de la conjecture.
2. La **méthode de l'arbre récursif**: On convertit la récurrence en un arbre dont les nœuds représentent les coûts induits à différents niveaux de la récursivité ; On emploie ensuite des techniques de bornage de sommation pour résoudre la récurrence.
3. La **méthode générale**: elle fournit des bornes pour les récurrences de la forme :

$$T(n) = aT\left(\frac{n}{b}\right) + f(n) \text{ où } a \geq 1, b > 1 \text{ et;}$$

$f(n)$ est une fonction donnée

YOUSSOU DIENG (YDIENG@UNIV-ZIG.SN)

4

MÉTHODE DE L'ARBRE RÉCURSIF

1. Tracer un arbre récursif, est un moyen direct d'arriver à une bonne conjecture.
2. Dans un *arbre récursif*, chaque nœud représente le coût d'un sous-problème individuel, situé quelque part dans l'ensemble des invocations récursives de fonction.
3. On totalise les coûts pour chaque niveau de l'arbre afin d'obtenir un ensemble de coûts par niveau, puis l'on cumule tous les coûts par niveau pour calculer le coût total de la récursivité tous niveaux confondus.

YOUSSOU DIENG (YDIENG@UNIV-ZIG.SN)

5

MÉTHODE DE L'ARBRE RÉCURSIF

Les arbres récursifs sont particulièrement utiles quand la récurrence décrit le temps d'exécution d'un algorithme diviser-pour-régner.

Un arbre récursif s'utilise de préférence pour générer une bonne conjecture avant de la confirmer via la **méthode de substitution**.

➤ Néanmoins, si l'on trace l'arbre récursif et que l'on cumule les coûts avec une très grande méticulosité, l'arbre fournit une preuve directe d'une solution pour la récurrence.

➤ Voyons, comment un arbre récursif fournirait une bonne conjecture pour la récurrence $T(n) = 3T(n/4) + \Theta(n^2)$.

YOUSSOU DIENG (YDIENG@UNIV-ZIG.SN)

6

MÉTHODE DE L'ARBRE RÉCURSIF

Comment trouver une borne supérieure pour la solution?

$$T(n) = 3T(n/4) + \Theta(n^2)$$

Nous créons un arbre récursif pour la récurrence $T(n) = 3T(n/4) + cn^2$, dans laquelle nous avons explicité le coefficient constant $c > 0$ implicite.

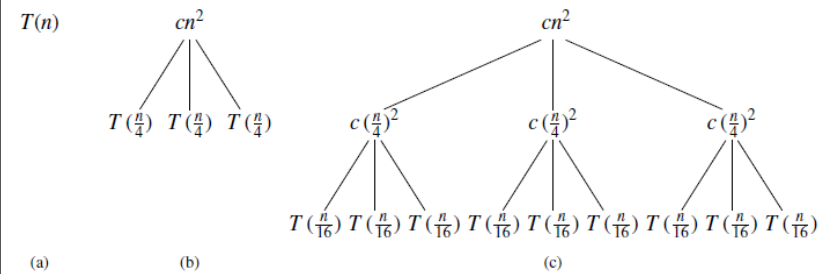
Pour des raisons de commodité, l'on suppose que n est une puissance exacte de 4.

YOUSSOU DIENG (YDIENG@UNIV-ZIG.SN)

7

MÉTHODE DE L'ARBRE RÉCURSIF

Tracé progressif de l'arbre récursif associé à $T(n) = 3T(n/4) + cn^2$



YOUSSOU DIENG (YDIENG@UNIV-ZIG.SN)

8

$T(n)$

(a) (b) (c)

a) La partie (a) : On montre $T(n)$,

b) La partie (b) : $T(n)$ est développé pour devenir un arbre équivalent représentant la récurrence. Le terme cn^2 situé sur la racine représente le coût au niveau supérieur de la récursivité, et les trois sous-arbres de la racine représentent les coûts induits par les sous-problèmes de taille $n/4$.

c) La partie (c) : On montre le processus une étape plus loin, après expansion de chacun des nœuds de coût $T(n/4)$ de la partie (b). Le coût de chacun des trois enfants de la racine est $c(n/4)^2$.

On continue de développer chaque nœud de l'arbre en le décomposant en ses parties constitutives, telles que déterminées par la récurrence.

YOUSSOU DIENG (YDIENG@UNIV-ZIG.SN)

9

$T(n)$

(a) (b) (c)

Comme la taille du sous-problème décroît à mesure que l'on s'éloigne de la racine, on finira par atteindre une condition aux limites.

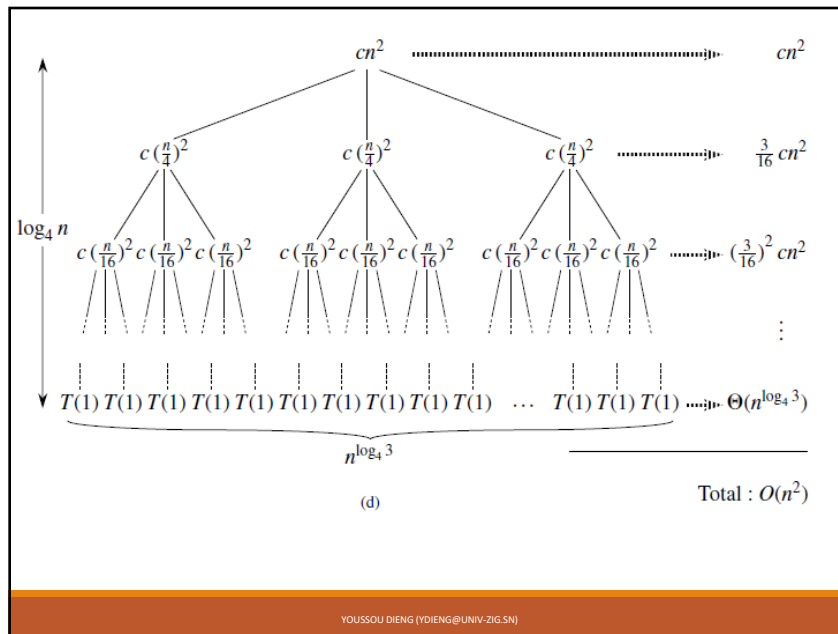
À quelle distance de la racine cela se produira-t-il ?

Comme la taille de sous-problème pour un nœud situé à la profondeur i est de $n/4^i$. La taille de sous-problème prendra la valeur $n = 1$ quand $n/4^i = 1$ ou, de manière équivalente, quand $i = \log_4 n$.

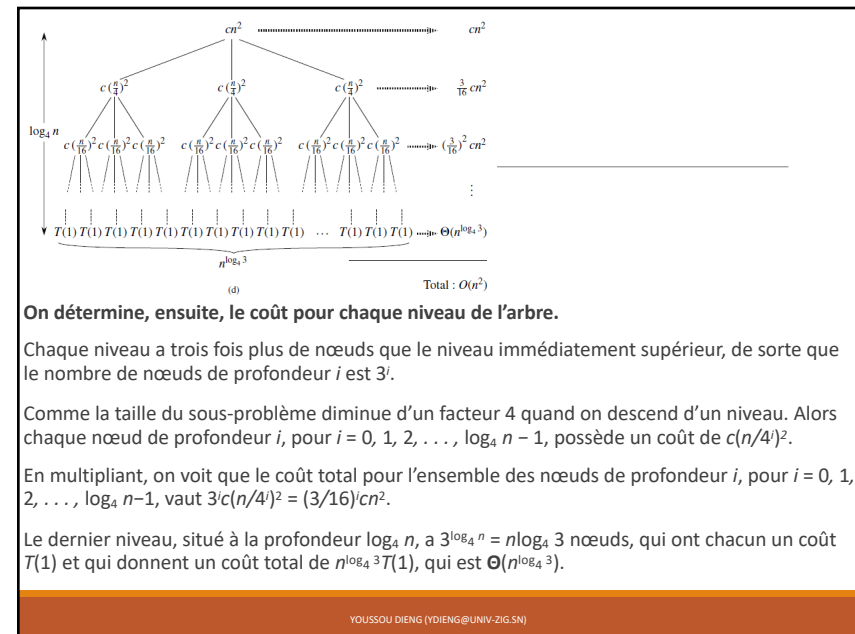
Par conséquent, l'arbre a $\log_4 n + 1$ niveaux $(0, 1, 2, \dots, \log_4 n)$.

YOUSSOU DIENG (YDIENG@UNIV-ZIG.SN)

10



11



On détermine, ensuite, le coût pour chaque niveau de l'arbre.

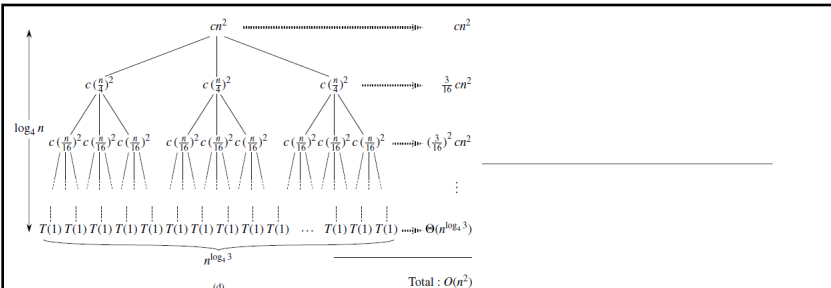
Chaque niveau a trois fois plus de nœuds que le niveau immédiatement supérieur, de sorte que le nombre de nœuds de profondeur i est 3^i .

Comme la taille du sous-problème diminue d'un facteur 4 quand on descend d'un niveau. Alors chaque nœud de profondeur i , pour $i = 0, 1, 2, \dots, \log_4 n - 1$, possède un coût de $c(n/4^i)^2$.

En multipliant, on voit que le coût total pour l'ensemble des nœuds de profondeur i , pour $i = 0, 1, 2, \dots, \log_4 n - 1$, vaut $3^i c(n/4^i)^2 = (3/16)^i cn^2$.

Le dernier niveau, situé à la profondeur $\log_4 n$, a $3^{\log_4 n} = n^{\log_4 3}$ nœuds, qui ont chacun un coût $T(1)$ et qui donnent un coût total de $n^{\log_4 3} T(1)$, qui est $\Theta(n^{\log_4 3})$.

12



On détermine, enfin, le coût global de l'arbre.
On cumule les coûts de tous les niveaux :

$$T(n) = cn^2 + \frac{3}{16}cn^2 + \left(\frac{3}{16}\right)^2 n^2 + \dots + \left(\frac{3}{16}\right)^{\log_4 n - 1} cn^2 + \Theta(n^{\log_4 3})$$

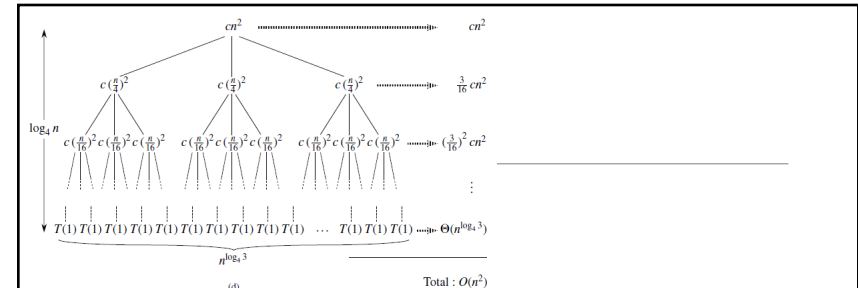
$$= \sum_{i=0}^{\log_4 n - 1} \left(\frac{3}{16}\right)^i cn^2 + \Theta(n^{\log_4 3})$$

$$= \frac{\left(\frac{3}{16}\right)^{\log_4 n} - 1}{\frac{3}{16} - 1} cn^2 + \Theta(n^{\log_4 3})$$

Une petite approximation pour borner supérieurement le résultat à l'aide d'une série géométrique décroissante.

YOUSSOU DIENG (YDIENG@UNIV-ZIG.SN)

13



On borner supérieurement le résultat :

$$T(n) = \sum_{i=0}^{\log_4 n - 1} \left(\frac{3}{16}\right)^i cn^2 + \Theta(n^{\log_4 3}) < \sum_{i=0}^{\infty} \left(\frac{3}{16}\right)^i cn^2 + \Theta(n^{\log_4 3})$$

$$= \frac{1}{1 - \frac{3}{16}} cn^2 + \Theta(n^{\log_4 3}) = \frac{16}{13} cn^2 + \Theta(n^{\log_4 3}) = O(n^2)$$

Nous avons donc déterminé une conjecture $T(n) = O(n^2)$ pour la récurrence originelle $T(n) = 3T(n/4) + \Theta(n^2)$.

Dans cet exemple, les coefficients de cn^2 forment une série géométrique décroissante ; la somme de ces coefficients est bornée supérieurement par la constante 16/13.

Comme la contribution de la racine au coût global est cn^2 , la racine contribue pour une fraction constante du coût global. En d'autres termes, le coût total de l'arbre est dominé par le coût de la racine.

YOUSSOU DIENG (YDIENG@UNIV-ZIG.SN)

14

En bref, ...

$O(n^2)$ est bien une borne supérieure pour la récurrence, alors il doit s'agir d'une borne très approchée. En effet, le premier appel récursif contribue pour un coût de $\Theta(n^2)$, et donc $\Omega(n^2)$ doit être une borne inférieure pour la récurrence.

On peut maintenant employer la **méthode de substitution** pour vérifier la validité de la conjecture, à savoir que $T(n) = O(n^2)$ est une borne supérieure de la récurrence $T(n) = 3T(n/4) + \Theta(n^2)$.

Dans cette partie, ...

Le souhait est de présenter ici une méthode pour résoudre les récurrences (obtenir des bornes asymptotiques « Θ » ou « O » pour la solution).

1. La **méthode de substitution**: on devine une borne puis on utilise une récurrence mathématique pour démontrer la validité de la conjecture.
2. La **méthode de l'arbre récursif**: On convertit la récurrence en un arbre dont les nœuds représentent les coûts induits à différents niveaux de la récursivité ; On emploie ensuite des techniques de bornage de sommation pour résoudre la récurrence.
3. La **méthode générale**: elle fournit des bornes pour les récurrences de la forme :

$$T(n) = aT\left(\frac{n}{b}\right) + f(n) \text{ où } a \geq 1, b > 1 \text{ et;} \\ f(n) \text{ est une fonction donnée}$$

La méthode par substitution

Elle se déroule en deux phases :

1. Conjecturer la forme de la solution.
2. Employer une récurrence mathématique pour trouver les constantes et prouver que la solution est correcte.

Cette méthode est puissante, mais ne peut s'utiliser que lorsque la forme de la réponse est facile à deviner.

Elle peut servir à borner une récurrence par excès ou par défaut.

YOUSSOU DIENG (YDIENG@UNIV-ZIG.SN)

17

Exemple

Calculons une borne supérieure pour la récurrence

$$T(n) = 2T\left(\frac{n}{2}\right) + n$$

On conjecture que la solution est $T(n) = O(n \lg n)$.

La méthode consiste à démontrer que $T(n) \leq c(n \lg n)$ pour un choix approprié de la constante $c > 0$.

YOUSSOU DIENG (YDIENG@UNIV-ZIG.SN)

18

Exemple

On commence par supposer que cette borne est valable pour $n/2$, autrement dit que

$$T(\lfloor n/2 \rfloor) \leq c \lfloor n/2 \rfloor \lg(\lfloor n/2 \rfloor).$$

La substitution dans,

$$T(n) = 2T\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + n$$

donne

$$\begin{aligned} T(n) &\leq 2(c \lfloor n/2 \rfloor \lg(\lfloor n/2 \rfloor)) + n \\ &\leq cn \lg(n/2) + n \\ &= cn \lg n - cn \lg 2 + n \\ &= cn \lg n - cn + n \\ &\leq cn \lg n, \end{aligned}$$

la dernière étape étant vraie si $c \geq 1$.

YOUSSOU DIENG (YDIENG@UNIV-ZIG.SN)

19

Vérification

On souhaite vérifier la validité de la conjecture de la première partie qui dit que $T(n) = O(n^2)$ est une borne supérieure de la récurrence $T(n) = 3T(n/4) + \Theta(n^2)$.

□ On veut montrer que $T(n) \leq dn^2$ pour une certaine constante $d > 0$.

□ En utilisant la même constante $c > 0$ que précédemment, on a

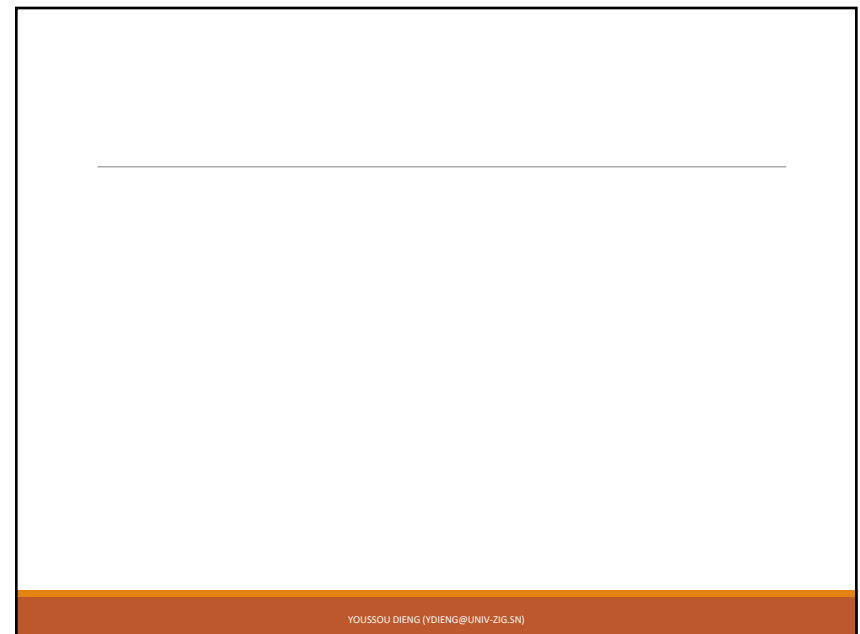
$$\begin{aligned} T(n) &\leq 3T(\lfloor n/4 \rfloor) + cn^2 \leq 3d \lfloor n/4 \rfloor^2 + cn^2 \\ &\leq 3d(n/4)^2 + cn^2 = \frac{3}{16} dn^2 + cn^2 \leq dn^2, \text{ si } d \geq (16/13)c. \end{aligned}$$

YOUSSOU DIENG (YDIENG@UNIV-ZIG.SN)

20



21



22

MÉTHODE GÉNÉRALE

- La méthode générale donne une « recette » pour résoudre les récurrences de la forme:

$$T(n) = aT(n/b) + f(n), \quad (4.5)$$

- où $a \geq 1$ et $b > 1$ sont des constantes, et $f(n)$ une fonction asymptotiquement positive.
- La méthode générale oblige à traiter trois cas de figure, mais permet de déterminer la solution de nombreuses récurrences assez facilement.

YOUSSOU DIENG (YDIENG@UNIV-ZIG.SN)

23

MÉTHODE GÉNÉRALE

- La récurrence (4.5) décrit le temps d'exécution d'un algorithme qui divise un problème de taille n en a sous-problèmes, chacun de taille n/b , a et b étant des constantes positives.
- Les a sous-problèmes sont résolus récursivement, chacun dans un temps $T(n/b)$.
- Le coût induit par la décomposition du problème et la combinaison des résultats des sous-problèmes est décrit par la fonction $f(n)$.
 - C'est-à-dire par $f(n) = D(n) + C(n)$,
- Ainsi, la récurrence résultant de la procédure TRI-FUSION donne $a = 2$, $b = 2$ et $f(n) = \Theta(n)$.

YOUSSOU DIENG (YDIENG@UNIV-ZIG.SN)

24

MÉTHODE GÉNÉRALE

- ❑ Rigoureusement parlant, la récurrence n'est pas vraiment bien définie, puisque n/b pourrait ne pas être un entier.
- ❑ Cependant, remplacer chacun des a termes $T(n/b)$ par $T(n/b)$ ou par $T(n/b)$ n'affecte pas le comportement asymptotique de la récurrence.
- ❑ On trouve donc généralement commode d'omettre les parties entières quand on écrit des récurrences diviser-pour-régner de cette forme.

YOUSSOU DIENG (YDIENG@UNIV-ZIG.SN)

25

Théorème général

Théorème 4.1 (Théorème général.) Soient $a \geq 1$ et $b > 1$ deux constantes, soit $f(n)$ une fonction et soit $T(n)$ définie pour les entiers non négatifs par la récurrence

$$T(n) = aT(n/b) + f(n),$$

où l'on interprète n/b comme signifiant $\lfloor n/b \rfloor$ ou $\lceil n/b \rceil$. $T(n)$ peut alors être bornée asymptotiquement de la façon suivante.

- 1) Si $f(n) = O(n^{\log_b a - \varepsilon})$ pour une certaine constante $\varepsilon > 0$, alors $T(n) = \Theta(n^{\log_b a})$.
- 2) Si $f(n) = \Theta(n^{\log_b a})$, alors $T(n) = \Theta(n^{\log_b a} \lg n)$.
- 3) Si $f(n) = \Omega(n^{\log_b a + \varepsilon})$ pour une certaine constante $\varepsilon > 0$, et si $af(n/b) \leq cf(n)$ pour une certaine constante $c < 1$ et pour tout n suffisamment grand, alors $T(n) = \Theta(f(n))$.

YOUSSOU DIENG (YDIENG@UNIV-ZIG.SN)

26

Théorème général

- Avant d'appliquer le théorème à des exemples, prenons un moment pour essayer de comprendre sa signification.
- Dans chacun des trois cas, on compare la fonction $f(n)$ à la fonction $n^{\log_b a}$.
- Intuitivement, la solution de la récurrence est déterminée par la plus grande des deux fonctions.
- Si c'est la fonction $n^{\log_b a}$ qui est la plus grande (cas 1), alors la solution est $T(n) = \Theta(n^{\log_b a})$.
- Si c'est la fonction $f(n)$ qui est la plus grande (cas 3), alors la solution est $T(n) = \Theta(f(n))$.
- Si les deux fonctions ont la même taille (cas 2), on multiplie par un facteur logarithmique et la solution est $T(n) = \Theta(n^{\log_b a} \lg n) = \Theta(f(n) \lg n)$.

YOUSSOU DIENG (YDIENG@UNIV-ZIG.SN)

27

Théorème général

- Derrière cette intuition se cachent certaines finesses techniques.
- Dans le premier cas, non seulement $f(n)$ doit être plus petite que $n^{\log_b a}$, mais elle doit être plus petite *polynomialement*.
- Autrement dit, $f(n)$ doit être asymptotiquement inférieure à $n^{\log_b a}$ d'un facteur n^ε , pour une certaine constante $\varepsilon > 0$.
- Dans le troisième cas, non seulement $f(n)$ doit être plus grande que $n^{\log_b a}$, mais elle doit être plus grande polynomialement et,
 - satisfaire à la condition de « régularité » selon laquelle $af(n/b) \leq cf(n)$.
 - Cette condition est satisfaite par la plupart des fonctions polynomialement bornées que nous rencontrerons.

YOUSSOU DIENG (YDIENG@UNIV-ZIG.SN)

28

Théorème général

- ❑ Il faut bien comprendre que les trois cas ne recouvrent pas toutes les possibilités pour $f(n)$.
 - ❑ Il y a un fossé entre les cas 1 et 2, quand $f(n)$ est plus petite que $n^{\log_b a}$ mais pas polynomialement plus petite.
 - ❑ De même, il existe un fossé entre les cas 2 et 3 quand $f(n)$ est plus grande que $n^{\log_b a}$ mais pas polynomialement plus grande.
- ❑ Si la fonction $f(n)$ tombe dans l'un de ces fossés, ou si la condition de régularité du cas 3 n'est pas vérifiée, alors la méthode générale ne peut pas servir à résoudre la récurrence.

YOUSSOU DIENG (YDIENG@UNIV-ZIG.SN)

29

Utilisation de la méthode générale

- ❑ Pour utiliser la méthode générale, on se contente de déterminer le cas du théorème général qui s'applique (s'il y en a un) puis d'écrire la réponse.
- ❑ Comme premier exemple, considérons: $T(n) = 9T(n/3) + n$.
 - ❑ Pour cette récurrence, on a $a = 9$, $b = 3$ et $f(n) = n$; on a donc $n^{\log_b a} = n^{\log_3 9} = \Theta(n^2)$.
 - ❑ Puisque $f(n) = O(n^{\log_3 9 - \epsilon})$, où $\epsilon = 1$, on peut appliquer le cas 1 du théorème général et dire que la solution est $T(n) = \Theta(n^2)$.

YOUSSOU DIENG (YDIENG@UNIV-ZIG.SN)

30

Utilisation de la méthode générale

- Considérons à présent : $T(n) = T(2n/3) + 1$,
 - où $a = 1$, $b = 3/2$, $f(n) = 1$ et $n^{\log_b a} = n^{\log_{3/2} 1} = n^0 = 1$.
 - On est dans le cas 2 puisque $f(n) = \Theta(n^{\log_b a}) = \Theta(1)$, et donc la solution de la récurrence est $T(n) = \Theta(\lg n)$.
- \log_b

YOUSSOU DIENG (YDIENG@UNIV-ZIG.SN)

31

Utilisation de la méthode générale

- Pour la récurrence: $T(n) = 3T(n/4) + n \lg n$,
on a $a = 3$, $b = 4$, $f(n) = n \lg n$ et $n^{\log_b a} = n^{\log_4 3} = O(n^{0,793})$.
Puisque $f(n) = \Omega(n^{\log_4 3 + \varepsilon})$,
où $\varepsilon \approx 0,2$, on est dans le cas 3 si l'on peut montrer que la condition de régularité est vraie pour $f(n)$.
Pour n suffisamment grand, $af(n/b) = 3(n/4) \lg(n/4) \leq (3/4)n \lg n = cf(n)$, avec $c = 3/4$.
Par conséquent, d'après le cas 3, la solution de la récurrence est $T(n) = \Theta(n \lg n)$.
- $\log_{3/2} n = \Omega(n^{\log_4 3 + \varepsilon})$

YOUSSOU DIENG (YDIENG@UNIV-ZIG.SN)

32

Utilisation de la méthode générale

- ❑ La méthode générale ne s'applique pas à la récurrence :

$$T(n) = 2T(n/2) + n \lg n ,$$

- ❑ bien qu'elle ait la forme correcte : $a = 2, b = 2, f(n) = n \lg n$ et $n^{\log_b a} = n$.

- ❑ On pourrait penser que le cas 3 s'applique, puisque $f(n) = n \lg n$ est plus grande asymptotiquement que $n^{\log_b a} = n$.

- ❑ Le problème est qu'elle n'est pas *polynomialement* plus grande.

Le rapport $f(n)/n^{\log_b a} = (n \lg n)/n = \lg n$

- ❑ est asymptotiquement plus petit que n^ϵ pour toute constante positive ϵ .
- ❑ En conséquence, la récurrence tombe dans le fossé situé entre le cas 2 et le cas 3.