LE MASTER THEOREM

Algorithme diviser-pour-régner

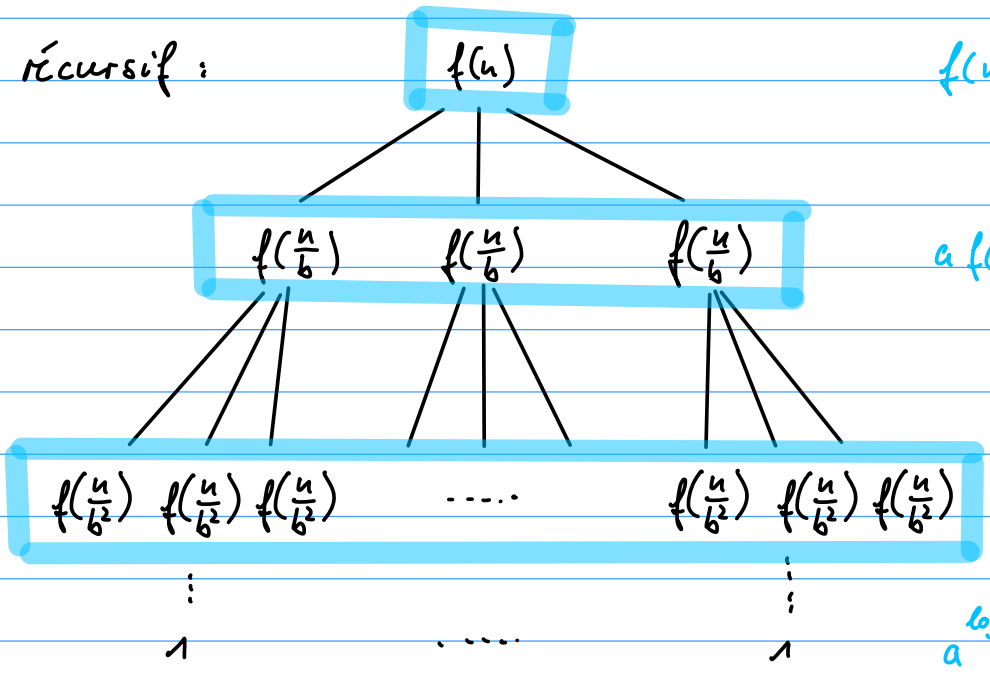
1) **Diviser** le problème en plusieurs sous-problèmes
2) **Régner** : Résudre chaque sous-problème de manière récursive
3) **Combiner** les solutions des sous-problèmes pour produire une solution du problème initial

Complexité d'un tel algorithme : $T(n) = a \cdot T(\frac{n}{b}) + f(n)$

- $T(n) : \mathbb{N} \to \mathbb{N}$: coût d'execution sur un problème de taille n
- a : nombre de sous-problèmes
- n/b : taille de chaque sous-problème
- $f(n)$: coût de 1) + 3) (diviser + combiner)

<u>Remarque</u> : n/b veut dire $\lfloor \frac{n}{b} \rfloor$ ou $\lceil \frac{n}{b} \rceil$. Dans les deux cas le résultat est le même (asymptotiquement).

L'arbre récursif :



coût des feuilles : $a^{\log_b n} = a^{\frac{\ln n}{\ln b}}$

$= a^{\frac{\ln n}{\ln a} \cdot \frac{\ln a}{\ln b}} = a^{\log_a n \cdot \log_b a} = n^{\log_b a}$

coût de la racine : $f(n)$

, si on suppose qu'on paye $\Theta(1)$ pour chaque feuille

au total : $\Theta(n^{\log_b a}) + \sum_{j=0}^{\log_b n - 1} a^j f\left(\frac{n}{b^j}\right)$

La solution de la récurrence est déterminée par la plus grande des deux fonctions :

Théorème :

$T(n)$ peut être bornée asymptotiquement de la façon suivante :

1) Si $f(n) = O(n^{\log_b a - \varepsilon})$ pour une certaine constante $\varepsilon > 0$
[le coût des feuilles domine le coût de la racine]
alors $T(n) = \Theta(n^{\log_b a})$

2) Si $f(n) = \Theta(n^{\log_b a}) \equiv f(n) = O(n^{\log_b n}) \wedge n^{\log_b n} = O(f(n))$
[le coût des feuilles et le coût de la racine sont du même comportement asymptotique]
alors $T(n) = \Theta(n^{\log_b a} \log n)$

3) Si $f(n) = \Omega(n^{\log_b a + \varepsilon})$ pour une certaine constante $\varepsilon > 0$
[le coût de la racine domine le coût des feuilles]
condition de régularité → [et si $a f\left(\frac{n}{b}\right) \leq c f(n)$ pour une certaine constante $c < 1$ et pour tout n suffisamment grand alors $T(n) = \Theta(f(n))$

Exemple : $T(n) = 9\,T(\frac{n}{3}) + n$

$a = 9,\ b = 3,\ f(n) = n$

$\Rightarrow n^{\log_b a} = n^{\log_3 9} = \Theta(n^2)$

on a $f(n) = O(n^{\log_3 9 - \varepsilon})$ pour $\varepsilon = 1 > 0$, on est donc dans le cas 1)

Alors $T(n) = \Theta(n^2)$

Exemple : $T(n) = 3\,T(\frac{n}{4}) + n \lg n$

$a = 3,\ b = 4,\ f(n) = n \lg n$

$\Rightarrow n^{\log_b a} = n^{\log_4 3} = O(n^{0,793})$

$\Rightarrow f(n) = \Omega(n^{\log_4 3 + \varepsilon})$ avec $\varepsilon \approx 0,2$

on est donc dans le cas 3)

Il faut vérifier la condition de regularité :

$af(n/b) = 3\,\frac{n}{4} \lg(\frac{n}{4}) \leq \frac{3}{4}\, n \lg n = c\, f(n)$

$\Rightarrow T(n) = \Theta(n \lg n)$

Exemple : $T(n) = 2\,T(\frac{n}{2}) + n \lg n$

$a = 2,\ b = 2,\ f(n) = n \lg n$

$n^{\log_b a} = n^{\log_2 2} = n \leq n \log n = f(n)$

$f(n) = n \log n \neq \Omega(n^{1+\varepsilon})$ pour tout $\varepsilon > 0$