Nbr occ :

Recherche borne droite

int $Bd(T, g, d, v)$:

- Si $g > d$ Alors return $g-1$
- $m = (g+d)/2$
- Si $T[m] > v$ Alors return $Bd(T, g, m-1, v)$
- sinon return $Bd(T, m+1, d, v)$

" " gauche

$d+1$

$T[m] < v$ $Bd(T, m+1, d, v)$

$Bd(T, m-1, d, v)$

Complexité m̂ que reche dico standard

ex :

CC de cet Algo :

1) Le calcul s'arrête

2) --- avec un appel où $g = d+1$

3) $d - g + 1 \geq 0$ (invariant)

Montrons que

$Bd(T, g, d, v)$ = le ⊕ grand indice $g \leq i \leq d$ t.q. $T[i] \leq v$

(et $g-1$ si il n'en existe pas)

↳ Montrons par induction sur la taille de la zone de recherche

$(d-g+1)$

Cas $T[m] > v$:

la zone des v est à gauche de m.

→ $Bd(T, g, m-1, v)$ appel

Cas $T[m] \leq v$:

// regarde si on est dans la nbr de vals.

Complexité : $O(\log_2 n)$
// Tri fusion en TD
- Algorithme de Karatsuba

entrée : deux entiers a et b de n chiffres dans une base r
sortie : le produit de $a \times b$

↳ algo naïf $O(n^2)$ opérations

On a : $m = \frac{n}{2}$

$$\begin{cases} a = \alpha_1 . r^m + \alpha_0 \\ b = \beta_1 . r^m + \beta_0 \end{cases} \qquad \alpha_0, \alpha_1, \beta_0, \beta_1 < r^m$$

$$a.b = \alpha_1 \beta_1 . r^{2m} + (\alpha_0 \beta_1 + \alpha_1 \beta_0) r^m + \beta_0 \alpha_0$$

diviser pour régner : nn efficace
$\varphi(n) = 4\varphi(\frac{n}{2}) + \Theta(n)$
↳ $\Theta(n^2)$

Karatsuba : $3\varphi(\frac{n}{2}) + O(n)$
↳ $n^{1.58}$ ~ complexité

$$a.b = \underbrace{\alpha_1 \beta_1}_{K_2} r^{2m} + \underbrace{(\alpha_0 \beta_1 + \alpha_1 \beta_0)}_{K_1} r^m + \underbrace{\alpha_0 \beta_0}_{K_0}$$

↳ $K_{1_\Delta} = (\alpha_1 + \alpha_0)(\beta_1 + \beta_0) - K_2 - K_0$
ou
$K_{1_\nabla} = K_2 + K_0 - (\alpha_1 - \alpha_0)(\beta_1 - \beta_0)$

Exemple : r=10 n=4
12|87 × 42|21

n^2 Algo div pour régner
!= V
n^2 Algo naïfs
!= V
$n^{1.58}$ K_{1_Δ}
!= V
$n^{1.58}$ K_{1_∇}

Θ variat algo
↓
Θ opti.

Master theorem

théorème :

Soient 2 rationnels $a \geq 1$ et $b > 1$,

Soit $f(n)$ une fonction positive

Soit $T(n) = \begin{cases} a \cdot T\left(\frac{n}{b}\right) + f(n) & \text{si } n > 1 \\ \Theta(1) & \text{si } x = 1 \end{cases}$ ← ce qu'on doit trv dès qu'on a un algo style tri fusion. ↳ diviser en ss pbs

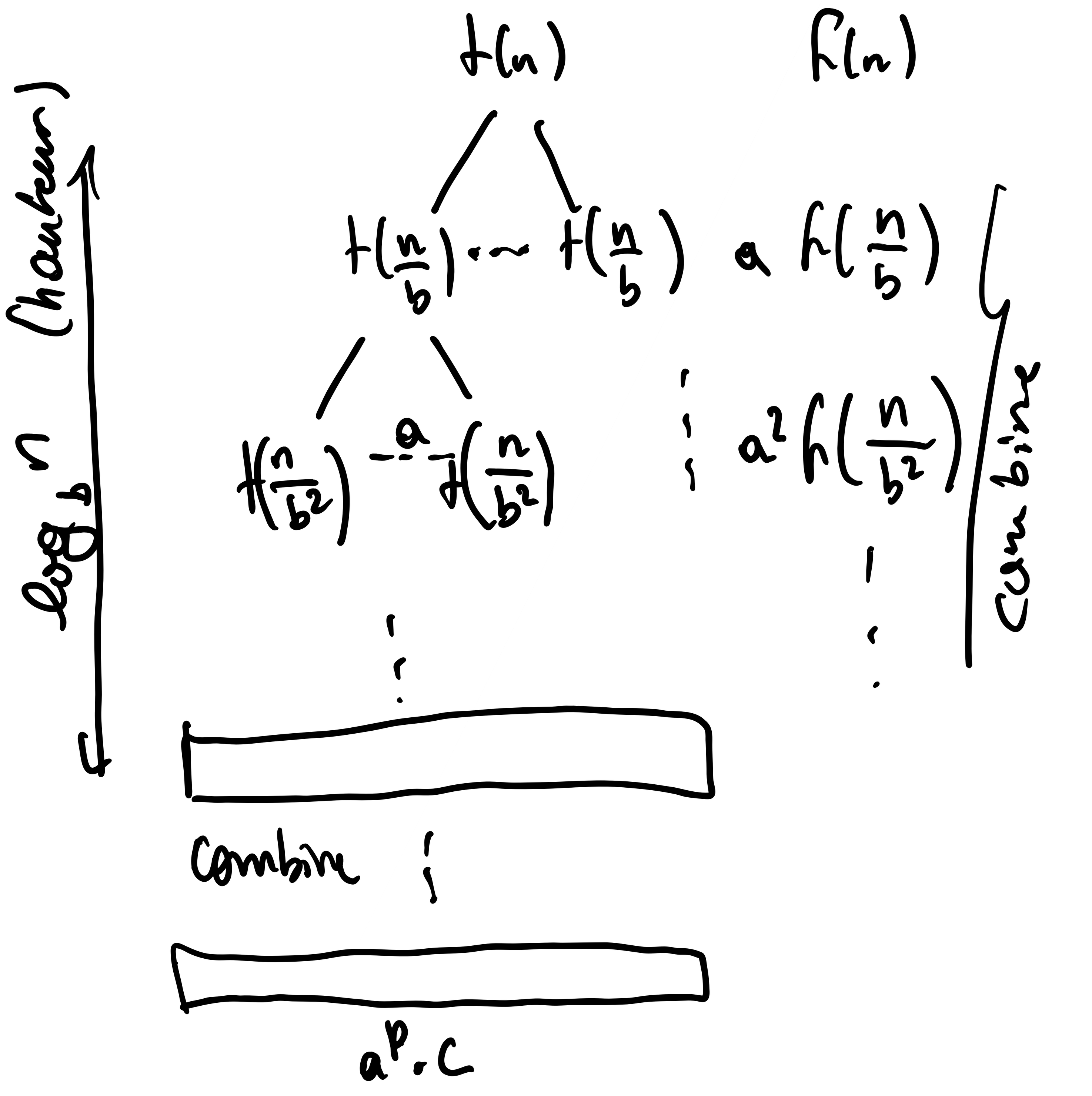
diviser pr régner

Alors on a :

① Si $f(n) = O(n^{\log_b(a) - \varepsilon})$ pour $\varepsilon > 0$, on a : $T(n) = \Theta(n^{\log_b a})$

② Si $f(n) = \Theta(n^{\log_b a})$, on a : $T(n) = \Theta(n^{\log_b a} \cdot \log n)$

③ Si $f(n) = \Omega(n^{\log_b a + \varepsilon})$ pour $\varepsilon > 0$ et si $\exists c < 1$ tq $a \cdot f\left(\frac{n}{b}\right) \leq c \cdot f(x)$ pour n assez gd,
alors : $T(n) = \Theta(f(n))$



coût total ⟶ $T(n) = \sum_{i=0}^{p-1} a^i f\left(\frac{n}{b^i}\right) + a^p \cdot c$

NB : $a^p = a^{\log_b n} = (b^{\log_b a})^{\log_b n} = (b^{\log_b n})^{\log_b a} = n^{\log_b a}$

Karatsuba $n^{1.58}$

// Th calcul dem sur page web proh. cas 1, 2 et 3.

Appliquer Master Théorème .

Recherche dichotomique : $t(n = t(\frac{n}{2}) + c$

$\left\{\begin{array}{l} a = 1 \quad b = 2 \\ f(n) = \theta(1) \end{array}\right.$ $\log_2 1 = 0$ cas 2 : $f(n) = \theta(1)$ $t(n) = \theta(n^{\log_b a} . \log n) = \theta(\log n)$

Tri fusion : $t(n) = 2t(\frac{n}{2}) + f(n)$

$a = 2 \quad b = 2$ $\log_2 2 = 1$ cas 2 $f(n) = \theta(n)$

$f(n) = \theta(n)$

$t(n) = \theta(n^{\log_b a} . \log n) = \theta(n \log n)$

Karatsuba : $t(n) = 3t(\frac{n}{2}) + f(n)$

$a = 3 \quad b = 2$ $\log_2 3 \simeq 1,58$ cas 1 ($\varepsilon \in]0 ; \log_2 3 - 1[$)

$f(n) = \theta(n)$

$t(n) = \theta(n^{\log_b a}) = \theta(n^{\log_2 3})$

Utiliser Master théorème :

1) trouver a et b
2) évaluer $\log_b a$ .. comparer $f(n)$ et $n^{\log_b a}$

↳ en déduire le cas à considérer...

(*p)++;   != *p++   => * et ++ pas m̂ priorité.

```
int *p = new int;
delete p;
```
} gestion memoire

en

```
int *q = new int[10];
delete []q;
```

! emplacement de car. !