Introduction

- Rappel : fonction factorielle.

```
let rec fact n=match n with
  | 0 -> 1
  | _ -> n* fact (n-1)
;;
```

Introduction

- Rappel : fonction factorielle.

```
let rec fact n=match n with
  | 0 -> 1
  | _ -> n* fact (n-1)
;;
```

- Analyse :
  - Terminaison : pas de suites à valeurs dans $\mathbb{N}$ strict. décroissante. Un appel récursif n'est fait que pour $n > 0$ et sur une valeur dans $\mathbb{N}$ strictement inférieure.
  - Correction : récurrence immédiate sur n que `fact n` calcule bien $n!$.
  - Complexité : $C(n)$ complexité de `fact n`. On a

$$C(n) = \underbrace{C(n-1)}_{\text{appel rec.}} + \underbrace{O(1)}_{\text{hors appel rec.}} \implies C(n) = O(n)$$

Terminaison : relations d'ordre

Relation d'ordre

$(E, \preceq)$. $\preceq$ relation d'ordre si :

- réflexivité : $\forall x \in E,\ x \preceq x$.
- transitivité : $\forall x, y, z \in E,\ x \preceq y$ et $y \preceq z \ \Rightarrow x \preceq z$.
- antisymétricité : $\forall x, y \in E \quad x \preceq y$ et $y \preceq x \Rightarrow \ x = y$.

Ordre total

Si de plus $\forall x, y \quad x \preceq y$ ou $y \preceq x$, l'ordre est total.

Éléments minimaux, plus petit élément

- x élément minimal : $\forall y \quad y \preceq x \Rightarrow x = y$;
- x plus petit élément : $\forall y \quad x \preceq y$.

Terminaison : relations d'ordre

Relation d'ordre

$(E, \preceq)$. $\preceq$ relation d'ordre si :

- réflexivité : $\forall x \in E$, $x \preceq x$.
- transitivité : $\forall x, y, z \in E$, $x \preceq y$ et $y \preceq z \Rightarrow x \preceq z$.
- antisymétricité : $\forall x, y \in E \quad x \preceq y$ et $y \preceq x \Rightarrow \quad x = y$.

Ordre total

Si de plus $\forall x, y \quad x \preceq y$ ou $y \preceq x$, l'ordre est total.

Éléments minimaux, plus petit élément

- x élément minimal : $\forall y \quad y \preceq x \Rightarrow x = y$;
- x plus petit élément : $\forall y \quad x \preceq y$.

Attention

Un plus petit élément est minimal, mais la réciproque n'est en général pas vrai si l'ordre n'est pas total.

Terminaison : ordre bien fondé

Def.

$(E, \preceq)$ *bien fondé* si pas de suite infinie strictement décroissante $x_0 \succ x_1 \succ \cdots x_i \succ \cdots$.
Si $\preceq$ total, on dit que $(E, \preceq)$ bien ordonné.

Ex.

$(\mathbb{N}, \leq)$ bien ordonné, $(\mathbb{Q}_+, \leq)$ ne l'est pas.

Ordres sur $\mathbb{N}^2$:

- Ordre produit : $(a, b) \preceq (c, d) \Leftrightarrow a \leq c$ et $b \leq d$
  pas total, bien fondé.
- Ordre lexico : $(a, b) \preceq (c, d) \Leftrightarrow a < c$ ou $(a = c$ et $b \leq d)$
  bien ordonné (exo).
- Ordre lexico gradué :
  $(a, b) \preceq (c, d) \Leftrightarrow a + c < b + d$ ou $(a + b = c + d$ et $a \leq c)$
  bien ordonné.

Terminaison : caractérisation des ensembles bien ordonnés

Prop.

$(E, \preceq)$ bien ordonné $\Leftrightarrow$ toute partie non vide admet un plus petit élément.

Preuve

- $\Rightarrow$. Se donner une partie non vide A, $x_0 \in A$. Si A n'a pas de ppe, construire une suite strictement décroissante.
- $\Leftarrow$. $\{x, y\}$ a un ppe $\Rightarrow$ ordre total. S'il existait $x_0 \succ x_1 \succ \cdots$ infinie, $\{x_i \mid i \in \mathbb{N}\}$ sans ppe.

Terminaison : principe d'induction

Def.

Prédicat sur E : application de E vers $\{\text{Vrai}, \text{Faux}\}$.

Terminaison : principe d'induction

Def.

Prédicat sur E : application de E vers $\{\text{Vrai, Faux}\}$.

Principe d'induction

Th. $(E, \preceq)$ bien fondé, $\mathcal{M}$ éléments minimaux. $\mathcal{P}$ prédicat. Si :

- $\forall x \in \mathcal{M}, \quad \mathcal{P}(x)$,
- $\forall x \in E \backslash \mathcal{M}, \quad (\forall y \prec x, \mathcal{P}(y)) \Rightarrow \mathcal{P}(x)$

Alors, pour tout $x \in E$, $\mathcal{P}(x)$.

Terminaison : principe d'induction

Def.

Prédicat sur E : application de E vers $\{$Vrai, Faux$\}$.

Principe d'induction

Th. $(E, \preceq)$ bien fondé, $\mathcal{M}$ éléments minimaux. $\mathcal{P}$ prédicat. Si :

- $\forall x \in \mathcal{M}, \quad \mathcal{P}(x)$,
- $\forall x \in E \backslash \mathcal{M}, \quad (\forall y \prec x, \mathcal{P}(y)) \Rightarrow \mathcal{P}(x)$

Alors, pour tout $x \in E$, $\mathcal{P}(x)$.

Preuve

Supposer $\exists x_0 \in E$ t.q non$(P(x_0))$ et construire une suite infinie strictement décroissante.

Terminaison : principe d'induction

Def.

Prédicat sur E : application de E vers $\{\text{Vrai, Faux}\}$.

Principe d'induction

Th. $(E, \preceq)$ bien fondé, $\mathcal{M}$ éléments minimaux. $\mathcal{P}$ prédicat. Si :

- $\forall x \in \mathcal{M}, \quad \mathcal{P}(x)$,
- $\forall x \in E \backslash \mathcal{M}, \quad (\forall y \prec x, \mathcal{P}(y)) \Rightarrow \mathcal{P}(x)$

Alors, pour tout $x \in E$, $\mathcal{P}(x)$.

Preuve

Supposer $\exists x_0 \in E$ t.q non$(P(x_0))$ et construire une suite infinie strictement décroissante.

Rem.

Généralise la récurrence (forte) sur $\mathbb{N}$.

Terminaison : application du principe d'induction

Terminaison d'une fonction récursive

f définie sur $\mathcal{A}$. Si $\exists\, \varphi : \mathcal{A} \to (E, \preceq)$ bien fondé tq :

- f termine sur $\mathcal{M}_{\mathcal{A}} = \{x \mid \varphi(x) \in \mathcal{M}\}$ avec $\mathcal{M}$ éléments minimaux de E ;
- $\forall\, x \in \mathcal{A} \backslash \mathcal{M}_{\mathcal{A}}$, l'appel $f(x)$ ne fait qu'un nombre fini (éventuellement aucun) d'appels à f, sur des y t.q $\varphi(y) \prec \varphi(x)$, et la terminaison des $f(y)$ entraîne celle de $f(x)$.

Alors $f(x)$ termine $\forall\, x \in \mathcal{A}$

Terminaison : application du principe d'induction

Terminaison d'une fonction récursive

f définie sur $\mathcal{A}$. Si $\exists \varphi : \mathcal{A} \to (E, \preceq)$ bien fondé tq :

- f termine sur $\mathcal{M}_{\mathcal{A}} = \{x \mid \varphi(x) \in \mathcal{M}\}$ avec $\mathcal{M}$ éléments minimaux de E ;
- $\forall x \in \mathcal{A} \backslash \mathcal{M}_{\mathcal{A}}$, l'appel $f(x)$ ne fait qu'un nombre fini (éventuellement aucun) d'appels à f, sur des y t.q $\varphi(y) \prec \varphi(x)$, et la terminaison des $f(y)$ entraîne celle de $f(x)$.

Alors $f(x)$ termine $\forall x \in \mathcal{A}$

Preuve

Principe d'induction sur $(E, \preceq)$ avec la propriété :
$\mathcal{P}(z)$: $\forall x \in \mathcal{A}$ t.q $\varphi(x) = z, f(x)$ termine.

Terminaison : application du principe d'induction

Terminaison d'une fonction récursive

f définie sur $\mathcal{A}$. Si $\exists\, \varphi : \mathcal{A} \to (E, \preceq)$ bien fondé tq :

- f termine sur $\mathcal{M}_{\mathcal{A}} = \{x \mid \varphi(x) \in \mathcal{M}\}$ avec $\mathcal{M}$ éléments minimaux de E ;
- $\forall\, x \in \mathcal{A} \backslash \mathcal{M}_{\mathcal{A}}$, l'appel $f(x)$ ne fait qu'un nombre fini (éventuellement aucun) d'appels à f, sur des y t.q $\varphi(y) \prec \varphi(x)$, et la terminaison des $f(y)$ entraîne celle de $f(x)$.

Alors $f(x)$ termine $\forall\, x \in \mathcal{A}$

Preuve

Principe d'induction sur $(E, \preceq)$ avec la propriété :
$\mathcal{P}(z)$: $\forall x \in \mathcal{A}$ t.q $\varphi(x) = z, f(x)$ termine.

Def. cas terminal

Si $f(x)$ ne fait aucun appel rec, on dit que x est un cas terminal.

Terminaison : exemples

En pratique...

φ sera simple : identité, projection sur un argument de f, longueur d'une liste/tableau, etc...

Terminaison : exemples

En pratique...

φ sera simple : identité, projection sur un argument de f, longueur d'une liste/tableau, etc...

factorielle

Prendre $\varphi = \mathrm{id}$!

Terminaison : exemples

En pratique...

φ sera simple : identité, projection sur un argument de f, longueur d'une liste/tableau, etc...

factorielle

Prendre $\varphi = \text{id}$!

Parcours de liste

```
let rec parcours l=match l with
  | [] -> ()
  | _::q -> parcours q
;;
```

$\varphi : \ell \mapsto$ nombre d'éléments de ℓ.

Terminaison : exemples

Calcul de $\binom{n}{p}$ pour $n, p \in \mathbb{N}$.

```
let rec binome n p=match (n,p) with
  | _,0 -> 1
  | 0,_ -> 0
  | _ -> (n*binome (n-1) (p-1))/p ;;
```

Propositions ?

Terminaison : exemples

Calcul de $\binom{n}{p}$ pour $n, p \in \mathbb{N}$.

```
let rec binome n p=match (n,p) with
  | _,0 -> 1
  | 0,_ -> 0
  | _ -> (n*binome (n-1) (p-1))/p ;;
```

Propositions ?

Calcul de $\binom{n}{p}$ pour $n, p \in \mathbb{N}$.

```
let rec binome n p=match (n,p) with
  | _,0 -> 1
  | 0,_ -> 0
  | _ -> binome (n-1) p + binome (n-1) (p-1) ;;
```

Propositions ?

Terminaison : exemples

Fonction d'Ackermann

```
let rec ack n p=match (n,p) with
  | 0,_ -> p+1
  | _,0 -> ack (n-1) 1
  | _ -> ack (n-1) (ack n (p-1))
;;
```

Terminaison : exemples

Fonction d'Ackermann

```
let rec ack n p=match (n,p) with
   | 0,_ -> p+1
   | _,0 -> ack (n-1) 1
   | _ -> ack (n-1) (ack n (p-1))
;;
```

Prendre $\varphi = \mathrm{id}$, avec $\mathbb{N}^2$ muni de l'ordre lexico.
Remarque : cette fonction croît très très vite : `ack 4 1` vaut 65533, `ack 4 2` vaut $2^{65556} - 3$.

Fonctions de Morris (ne termine pas)

```
let rec morris n p=match (n,p) with
   | 0,_ -> 1
   | _ -> morris n (p-1) + morris (n-1) (morris n p) ;;
```

Correction

C'est facile, une fois la terminaison montrée. Même cadre que pour la terminaison : f définie sur $\mathcal{A}$. $\varphi : \mathcal{A} \to (E, \preceq)$, qu'on suppose surjective.

Théorème

Considérons sur l'ensemble E le prédicat suivant : $\mathcal{P}(z)$: « les $f(x)$ pour $\varphi(x) = z$ ont la bonne valeur ». Supposons que

- $\forall x \in \mathcal{M}_{\mathcal{A}}$, $\mathcal{P}(x)$.
- pour tout x dans $\mathcal{A} \backslash \mathcal{M}_{\mathcal{A}}$, le calcul de $f(x)$ ne requiert qu'un nombre fini d'appels d'arguments $(y_i)_{1 \leq i \leq N}$ qui vérifient $\varphi(y_i) \prec \varphi(x)$ et

$$(\forall i \leq N, \mathcal{P}(\varphi(y_i))) \Longrightarrow \mathcal{P}(\varphi(x))$$

Alors pour tout $x \in \mathcal{A}$, $\mathcal{P}(x)$

Preuve immédiate par induction.

Exemple : tri par sélection

```
let mini q=
 let rec aux m reste p=match p with
   | [] -> m, reste
   | x::r when m<=x -> aux m (x::reste) r
   | x::r -> aux x (m::reste) r
 in aux (List.hd q) (List.tl q)
;;


let tri_selection q=match q with
  | [] -> []
  | _ -> let m,p=mini q in m::(tri_selection p)
;;
```

Complexité : introduction

- résoudre une récurrence de la forme
$$C(n) = \underbrace{\sum C(n_i)}_{\text{coût appels rec.}} + \underbrace{O(f(n))}_{\text{coût hors appels rec.}}$$

Complexité : introduction

- résoudre une récurrence de la forme
$$C(n) = \underbrace{\sum C(n_i)}_{\text{coût appels rec.}} + \underbrace{O(f(n))}_{\text{coût hors appels rec.}}$$
- Exemples :
  - factorielle : $C(n) = C(n-1) + O(1)$;
  - tri par sélection récursif : $C(n) = C(n-1) + O(n)$;
  - tri fusion : $C(n) = C(\lfloor n/2 \rfloor) + C(\lceil n/2 \rceil) + O(n)$;
  - Hanoi : $C(n) = 2C(n-1) + O(1)$.

Premiers résultats

Prop (remplacer un terme par un terme plus simple)

Soit $(b_n), (b'_n)$ deux suites réelles positives. Si $b_n = O(b'_n)$ alors $\sum_{k=0}^{n} b_k = O\left(\sum_{k=0}^{n} b'_k\right)$.

Preuve : distinguer les cas $\sum_{k=0}^{+\infty} b'_k < +\infty$ et $\sum_{k=0}^{+\infty} b'_k = +\infty$.

Premiers résultats

Prop (remplacer un terme par un terme plus simple)

Soit $(b_n), (b'_n)$ deux suites réelles positives. Si $b_n = O(b'_n)$ alors $\sum_{k=0}^{n} b_k = O\left(\sum_{k=0}^{n} b'_k\right)$.

Preuve : distinguer les cas $\sum_{k=0}^{+\infty} b'_k < +\infty$ et $\sum_{k=0}^{+\infty} b'_k = +\infty$.

Sommations classiques

Soient $\alpha > 0$, $q > 1$. Alors $\sum_{k=0}^{n} k^\alpha = \Theta(n^{\alpha+1})$, $\sum_{k=0}^{n} q^k = \Theta(q^n)$.

Preuve :

- $\sum_{k=0}^{n} k^\alpha \leq \int_0^{n+1} t^\alpha \mathrm{d}t$
- $\sum_{k=0}^{n} q^k = \frac{q^{n+1}-1}{q-1} = \Theta(q^n)$.

Applications :

- Factorielle : $\forall n > 0,\ C(n) = C(n-1) + O(1)$. On télescope :

$$C(n) = \sum_{k=1}^{n} (\underbrace{C(k) - C(k-1)}_{O(1)}) + \underbrace{C(0)}_{O(1)} = O(n)$$

Applications :

- Factorielle : $\forall n > 0,\ C(n) = C(n-1) + O(1)$. On télescope :

$$C(n) = \sum_{k=1}^{n} (\underbrace{C(k) - C(k-1)}_{O(1)}) + \underbrace{C(0)}_{O(1)} = O(n)$$

- Tri sélection récursif : $\forall n > 0,\ C(n) = C(n-1) + O(n)$. On télescope :

$$C(n) = \sum_{k=1}^{n} (\underbrace{C(k) - C(k-1)}_{O(k)}) + \underbrace{C(0)}_{O(1)} = O(n^2)$$

Applications :

- Factorielle : $\forall n > 0,\ C(n) = C(n-1) + O(1)$. On télescope :

$$C(n) = \sum_{k=1}^{n} (\underbrace{C(k) - C(k-1)}_{O(1)}) + \underbrace{C(0)}_{O(1)} = O(n)$$

- Tri sélection récursif : $\forall n > 0,\ C(n) = C(n-1) + O(n)$. On télescope :

$$C(n) = \sum_{k=1}^{n} (\underbrace{C(k) - C(k-1)}_{O(k)}) + \underbrace{C(0)}_{O(1)} = O(n^2)$$

- Plus généralement, $C(n) = C(n-1) + O(n^\alpha) \Rightarrow C(n) = O(n^{\alpha+1})$, pour $\alpha \geq 0$.

Sommations plus complexes

Théorème

Soit (u_n) vérifiant $u_n = au_{n-1} + b_n$, avec $b_n > 0$, $a > 0$, $u_0 \geq 0$.
Si $b_n = O(b^n)$, on a suivant les cas :

- si $b < a$, alors $u_n = O(a^n)$;
- si $b = a$, alors $u_n = O(na^n)$;
- si $b > a$, alors $u_n = O(b^n)$.

Théorème

Soit (u_n) vérifiant $u_n = au_{n-1} + b_n$, avec $b_n > 0$, $a > 0$, $u_0 \geq 0$.
Si $b_n = O(b^n)$, on a suivant les cas :

- si $b < a$, alors $u_n = O(a^n)$;
- si $b = a$, alors $u_n = O(na^n)$;
- si $b > a$, alors $u_n = O(b^n)$.

Preuve à savoir faire !

- On écrit $\frac{u_n}{a^n} = \frac{u_{n-1}}{a^{n-1}} + \frac{b_n}{a^n}$. On peut alors télescoper :
  $\frac{u_n}{a^n} = \sum_{k=1}^{n-1}(u_k - u_{k-1}) + u_0 = \sum_{k=1}^{n-1} \frac{b_k}{a^k} + u_0$. D'où
  $u_n = O\left(a^n \sum_{k=1}^{n-1} \frac{b_k}{a^k}\right)$.
- Avec $b_k = O(b^k)$, on discute facilement suivant les cas.

Sommations plus complexes

Théorème

Soit (u_n) vérifiant $u_n = au_{n-1} + b_n$, avec $b_n > 0$, $a > 0$, $u_0 \geq 0$.
Si $b_n = O(b^n)$, on a suivant les cas :

- si $b < a$, alors $u_n = O(a^n)$;
- si $b = a$, alors $u_n = O(na^n)$;
- si $b > a$, alors $u_n = O(b^n)$.

Preuve à savoir faire !

- On écrit $\frac{u_n}{a^n} = \frac{u_{n-1}}{a^{n-1}} + \frac{b_n}{a^n}$. On peut alors télescoper : $\frac{u_n}{a^n} = \sum_{k=1}^{n-1}(u_k - u_{k-1}) + u_0 = \sum_{k=1}^{n-1} \frac{b_k}{a^k} + u_0$. D'où $u_n = O\left(a^n \sum_{k=1}^{n-1} \frac{b_k}{a^k}\right)$.
- Avec $b_k = O(b^k)$, on discute facilement suivant les cas.

Tours de Hanoï

$C(n) = 2C(n-1) + O(1) \Rightarrow C(n) = O(2^n)$ (premier cas ici).

Récurrences « Diviser pour régner »

À résoudre

$u_n = au_{\lfloor \frac{n}{2} \rfloor} + bu_{\lceil \frac{n}{2} \rceil} + b_n$ pour $n \geq 2,\ a, b \in \mathbb{N}$ non tous deux nuls.

Récurrences « Diviser pour régner »

À résoudre

$u_n = au_{\lfloor \frac{n}{2} \rfloor} + bu_{\lceil \frac{n}{2} \rceil} + b_n$ pour $n \geq 2$, $a, b \in \mathbb{N}$ non tous deux nuls.

Tri fusion

$a = b = 1,\ b_n = O(n)$.

Récurrences « Diviser pour régner »

À résoudre

$u_n = au_{\lfloor \frac{n}{2} \rfloor} + bu_{\lceil \frac{n}{2} \rceil} + b_n$ pour $n \geq 2$, $a, b \in \mathbb{N}$ non tous deux nuls.

Tri fusion

$a = b = 1$, $b_n = O(n)$.

Simplication

Si u'_n la même suite définie avec (b'_n) à la place de (b_n), $b'_n > 0$, $b_n = O(b'_n)$, alors $u_n = O(u'_n)$.

Récurrences « Diviser pour régner »

À résoudre

$u_n = au_{\lfloor \frac{n}{2} \rfloor} + bu_{\lceil \frac{n}{2} \rceil} + b_n$ pour $n \geq 2$, $a, b \in \mathbb{N}$ non tous deux nuls.

Tri fusion

$a = b = 1$, $b_n = O(n)$.

Simplication

Si u'_n la même suite définie avec (b'_n) à la place de (b_n), $b'_n > 0$, $b_n = O(b'_n)$, alors $u_n = O(u'_n)$.

Croissance

Si (b_n) croissante, alors (u_n) croissante.

Théorème « Diviser pour régner »

Théorème

À résoudre $u_n = au_{\lfloor \frac{n}{2} \rfloor} + bu_{\lceil \frac{n}{2} \rceil} + b_n$, avec $a, b \in \mathbb{N}$, non tous deux nuls, $u_1 \geq 0$. On pose $\alpha = \log_2(a+b)$. Si $(b_n)_{n\geq 1}$ strictement positive vérifiant $b_n = O(n^\beta)$, on a :

- $\beta < \alpha$, $u_n = O(n^\alpha)$.
- $\beta = \alpha$, $u_n = O(n^\alpha \log(n))$.
- $\beta > \alpha$, $u_n = O(n^\beta)$.

Théorème « Diviser pour régner »

Théorème

À résoudre $u_n = au_{\lfloor \frac{n}{2} \rfloor} + bu_{\lceil \frac{n}{2} \rceil} + b_n$, avec $a, b \in \mathbb{N}$, non tous deux nuls, $u_1 \geq 0$. On pose $\alpha = \log_2(a+b)$. Si $(b_n)_{n\geq 1}$ strictement positive vérifiant $b_n = O(n^\beta)$, on a :

- $\beta < \alpha$, $u_n = O(n^\alpha)$.
- $\beta = \alpha$, $u_n = O(n^\alpha \log(n))$.
- $\beta > \alpha$, $u_n = O(n^\beta)$.

Preuve

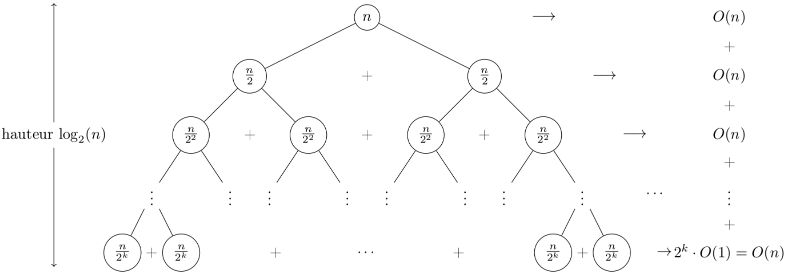
- On peut étudier $b_n = n^\beta$ exactement.
- On montre le résultat pour n puissance de 2, on conclut par croissance de (u_n) (si $n \leq t \leq 2n$, et $u_n = O(n^\alpha)$ pour n puissance de 2, alors $u_t = O(u_{2n}) = O((2n)^\alpha) = O(n^\alpha) = O(t^\alpha)$, de même pour les autres cas).
- Sur les puissances de 2, cas précédent : $v_p = u_{2^p}$, $v_p = (a+b)v_{p-1} + b_{2^p} = (a+b)v_{p-1} + O(2^{p\beta})$. Comparer $(a+b)$ et 2^β revient à comparer $\log_2(a+b)$ et β.

Preuve ad-hoc pour le tri fusion

À savoir faire !

- $C(n) = C(\lfloor n/2 \rfloor) + C(\lceil n/2 \rceil) + O(n)$, on résout pour n une puissance de 2.
- Pour $p \geq 1$, on a donc $C(2^p) = 2C(2^{p-1}) + O(2^p)$. On divise par 2^p.
- $\frac{C(2^p)}{2^p} = \frac{C(2^{p-1})}{2^{p-1}} + O(1)$ pour $p \geq 1$. On télescope.
- $\frac{C(2^p)}{2^p} = \sum_{k=1}^{p} \underbrace{\left(\frac{C(2^k)}{2^k} - \frac{C(2^{k-1})}{2^{k-1}}\right)}_{O(1)} + \underbrace{C(1)}_{O(1)} = O(p)$.
- D'où $C(2^p) = O(p2^p)$. Avec $n = 2^p$, $p = \log_2(n)$.
- Ainsi $C(n) = O(n \log n)$ pour n puissance de 2.
- En admettant la croissance de C, on a pour $n \leq t \leq 2n$ et n une puissance de 2 :
  $C(t) \leq C(2n) = O(2n \log(2n)) = O(n \log n) = O(t \log t)$.

En image...


hauteur $\log_2(n)$
n
$\frac{n}{2}$ + $\frac{n}{2}$
$\frac{n}{2^2}$ + $\frac{n}{2^2}$ + $\frac{n}{2^2}$ + $\frac{n}{2^2}$
$\frac{n}{2^k}$ + $\frac{n}{2^k}$ + $\cdots$ + $\frac{n}{2^k}$ + $\frac{n}{2^k}$
$\longrightarrow O(n)$
$+$
$\longrightarrow O(n)$
$+$
$\longrightarrow O(n)$
$+$
$\cdots$
$+$
$\rightarrow 2^k \cdot O(1) = O(n)$