# Complexité de fonctions récursives, premières récurrences

## I. Notations de Landau

Étant donné deux suites $(u_n)$ et $(v_n)$, on a :

- ⋆ $u_n = \mathrm{O}(v_n)$ s'il existe un $C > 0$ tel que $u_n \leq C v_n$ pour tout $n \in \mathbf{N}$ (ou, au moins, à partir d'un certain rang) ;
- ⋆ $u_n = \Omega(v_n)$ si $v_n = \mathrm{O}(u_n)$, autrement dit s'il existe un $C > 0$ tel que $C v_n \leq u_n$ pour tout $n \in \mathbf{N}$ (ou, au moins, à partir d'un certain rang) ;
- ⋆ $u_n = \Theta(v_n)$ si $u_n = \mathrm{O}(v_n)$ et $u_n = \Omega(v_n)$, autrement dit s'il existe des réels $0 < C \leq D$ tels que $C v_n \leq u_n \leq D v_n$ pour tout $n \in \mathbf{N}$ (ou, au moins, à partir d'un certain rang). Notons qu'il s'agit d'une relation d'équivalence.

Dans la suite, on ne donnera pas systématiquement les preuves pour O, pour $\Omega$ et pour $\Theta$.

## II. Un seul appel récursif

On a vu pour l'instant des fonctions récursives basées sur des parcours de liste et sur les entiers. En terme de complexité, il est clair que l'on a comme relations :

**Factoriel** $c_n = c_{n-1} + \Theta(1)$

**Tri par insertion** $c_n = c_{n-1} + \Theta(n)$

Ce type de récurrence est facile à résoudre.

**Proposition 1**

Si $(c_n)$ vérifie la relation

$$\forall\, n \in \mathbf{N}^\star,\ c_n = c_{n-1} + b_n$$

avec $b_n = \Theta\left(n^\alpha\right)$ pour un réel $\alpha \geq 0$ fixé, alors $c_n = \Theta\left(n^{\alpha+1}\right)$.

**Preuve**

Notons tout d'abord que :

$$\forall\, n \in \mathbf{N},\ c_n = c_0 + \sum_{k=1}^{n} b_k$$

⋆ À une constante multiplicative près, on a $b_k \leq n^\alpha$ d'où $c_n \leq c_0 + n \times n^\alpha$
⋆ De même, à une constante multiplicative près, au moins $n/2$ termes de la somme sont supérieurs ou égaux à $(n/2)^\alpha$, d'où $c_n \geq (n/2)^{\alpha+1}$.

Sinon, on peut faire un encadrement à l'aide d'intégrales, en remarquant que :

$$\int_{n-1}^{n} t^\alpha \,\mathrm{d}t \leq n^\alpha \leq \int_{n}^{n+1} t^\alpha \,\mathrm{d}t$$

et donc, par relation de Chasles :

$$\frac{n^{\alpha+1}}{\alpha+1} = \int_0^n t^\alpha \,\mathrm{d}t \leq \sum_{k=1}^{n} k^\alpha \leq \int_1^{n+1} t^\alpha \,\mathrm{d}t = \frac{(n+1)^{\alpha+1}-1}{\alpha+1}$$

Ainsi, le calcul de la factoriel est de complexité $\Theta(n)$ (en nombre de multiplications), et le tri par insertion (ainsi que le tri par sélection) est en $\Theta\left(n^2\right)$.

**Remarque** Concernant la complexité de la factorielle, la taille des entiers manipulés croît très vite, donc on ne peut considérer que les multiplications s'effectuent en temps constant. Cependant, dans un grand nombre de cas pratiques, on effectuera ces calculs modulo un entier fixé, auquel cas les multiplications sont bien en temps constant.

## III. Cas général

On s'intéresse maintenant aux récurrences de la forme :

$$u_{n+1} = a \cdot u_n + b_n$$

avec $a \in \mathbf{N}^\star$ et $(b_n)$ à valeurs positives.

On peut se ramener au cas $a = 1$ en posant $v_n = \dfrac{u_n}{a^n}$ pour tout $n \in \mathbf{N}$ puisqu'alors on a pour tout $n$, $v_{n+1} = v_n + \dfrac{b_n}{a^n}$ et donc

$$\forall\, n \in \mathbf{N},\ v_n = v_0 + \sum_{k=0}^{n-1} \frac{b_k}{a^k} \quad \textbf{et} \quad \forall\, n \in \mathbf{N},\ u_n = a^n\left(u_0 + \sum_{k=0}^{n-1} \frac{b_k}{a^k}\right)$$

On en déduit que :

⋆ si la série de terme général $\dfrac{b_n}{a^n}$ converge, alors $u_n = \Theta\left(a^n\right)$ ;
⋆ si $\dfrac{b_n}{a^n} = \Theta(1)$, alors $u_n = \Theta\left(na^n\right)$ ;

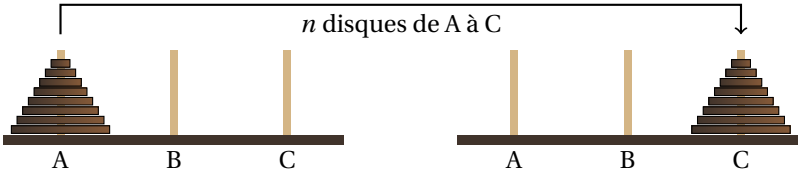
⋆ si $b_n = \Theta\big((\lambda a)^n\big)$ pour $\lambda > 1$, alors $u_n = \Theta\big(\lambda^n\big)$.

**Remarque** En s'inspirant du cas pour $a = 1$, si $\dfrac{b_n}{a^n} = \Theta\big(n^\alpha\big)$, alors $u_n = \Theta\big(n^{\alpha+1}a^n\big)$.

## 1. Un exemple : les tours de Hanoï

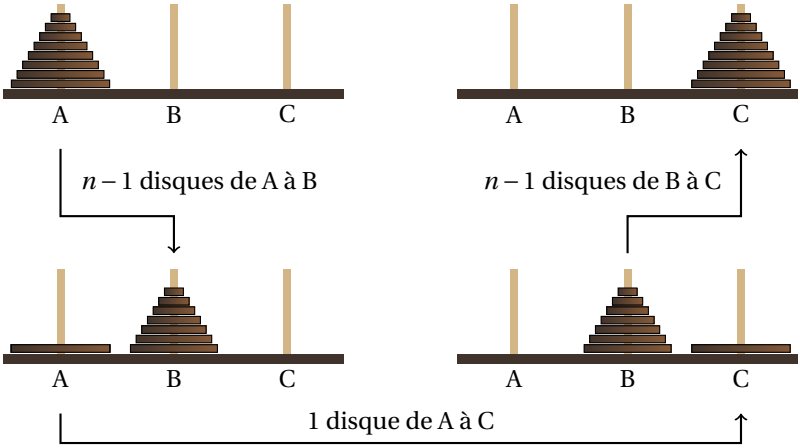
Il s'agit d'un casse-tête où l'on a trois tiges A, B et C et $n$ disques sur la tige A, de taille croissante du haut vers le bas. Le but est de transférer, en moins de coups possible, tous les disques vers la tige C en respectant les règles suivantes :

⋆ on ne peut déplacer plus d'un disque à la fois ;
⋆ on ne peut placer un disque que sur un autre disque plus grand que lui ou sur un emplacement vide.




Pour résoudre le problème avec $n$ disques, il suffit de savoir le faire avec $n-1$ disques. En effet,

1. on transfère $n-1$ disques de la tige A à la tige B,
2. on transfère le disque $n$ de la tige A à la tige C,
3. on transfère les $n-1$ autres disques de la tige B à la tige C.




Le programme correspondant, qui affiche les déplacements des disques, est alors très simple :

```
let rec hanoi a b c n =
  if n >= 1 then begin
    hanoi a c b (n - 1);
    Printf.printf "%s -%d-> %s\n" a n c;
    hanoi b a c (n - 1)
  end
;;
```

Si $c_n$ désigne le nombre d'étapes nécessaires pour transférer $n$ disques, alors clairement :

$$c_0 = 0 \qquad \forall\, n \in \mathbf{N},\ c_{n+1} = c_n + 1 + c_n = 2c_n + 1$$

On peut résoudre exactement cette récurrence :

$$\forall\, n \in \mathbf{N},\ c_n = 2^n - 1$$

Néanmoins, à titre d'exercice, on peut utiliser la méthode précédente. On a :

$$\forall\, n \in \mathbf{N},\ c_n = 2c_{n-1} + 1 = ac_{n-1} + b_n$$

avec $a = 2$ et $b_n = \Theta(1) = \Theta\left(1^n\right)$. Ainsi, on a

$$\sum_{k=1}^{n} \frac{b_k}{2^k} = \Theta(1)$$

et donc

$$c_n = 2^n\left(c_0 + \sum_{k=1}^{n} \frac{b_k}{2^k}\right) = \Theta\left(2^n\right)$$