La propriété d'Archimède généralisée

Marc Lorenzi

14 juillet 2025

Résumé

Étant donnés une fonction $f : \mathbb{N} \longrightarrow \mathbb{N}$ strictement croissante et un entier naturel t, il existe un unique entier naturel n tel que $f(n) \leq t < f(n+1)$. L'entier n peut être trouvé efficacement par dichotomie.

1 Encadrements

1.1 Motivation

La propriété d'Archimède pour les entiers naturels s'énonce comme suit.

Proposition 1. *Soient $a, b \in \mathbb{N}$ tels que $b \neq 0$. Il existe un unique $n \in \mathbb{N}$ tel que*

$$nb \leq a < (n+1)b$$

Pour montrer l'existence de n, on considère l'ensemble

$$E = \{n \in \mathbb{N} : nb > a\}$$

On montre ensuite que E n'est pas vide. Comme toute partie non vide de $\mathbb{N}$ possède un plus petit élément, E possède un plus petit élément m. On montre ensuite que $n = m - 1$ satisfait la double inégalité. L'unicité de n résulte quant à elle d'une propriété remarquable de l'ordre naturel sur $\mathbb{N}$: si $n', n'' \in \mathbb{N}$ et $n' < n'' + 1$, alors $n' \leq n''$.

Lorsqu'on regarde attentivement la preuve du théorème d'Archimède, on se rend compte qu'elle fonctionne parce que la fonction $f : n \longmapsto nb$ est strictement croissante.

1.2 Généralisation

Dans tout l'article, on se donne une fonction $f : \mathbb{N} \longrightarrow \mathbb{N}$ strictement croissante.

Proposition 2. *Pour tout $n \in \mathbb{N}$, $f(n) \geq n$.*

Démonstration. Faisons une récurrence sur n. On a $f(0) \in \mathbb{N}$, donc $f(0) \geq 0$. Soit $n \in \mathbb{N}$. Supposons $f(n) \geq n$. Comme f croît strictement, $f(n+1) > f(n)$ et donc $f(n+1) > n$. De là, comme nous travaillons avec des entiers, $f(n+1) \geq n+1$. □

Proposition 3. *Soit $t \in \mathbb{N}$. Il existe un unique $n \in \mathbb{N}$ tel que*

$$(1)\ f(n) \leq t < f(n+1)$$

Démonstration. Pour l'unicité, supposons qu'il existe deux tels entiers, n' et n''. On a

$$f(n') \leq t < f(n''+1)$$

donc $n' < n''+1$ car f est strictement croissante. Ainsi, $n' \leq n''$. De même, $n'' \leq n'$, donc $n' = n''$. Montrons maintenant l'existence. Soit

$$E = \{n \in \mathbb{N} : f(n) > t\}$$

On a $f(t+1) \geq t+1 > t$ donc $t+1 \in E$. Ainsi, $E \neq \emptyset$. Comme $E \subseteq \mathbb{N}$, E possède un plus petit élément m. On a $f(m) > t \geq 0$ donc $m \geq 1$. Soit $n = m-1$. Par la minimalité de m, $f(n) \leq t$. De plus, $t < f(m) = f(n+1)$. □

Remarque. Dans la preuve que nous venons de faire, nous avons vu que $0 \leq n \leq t$. Ceci nous donne un premier encadrement de n.

Notation. Pour tout $t \in \mathbb{N}$ nous noterons $\varphi_f(t)$ l'unique entier naturel n vérifiant (1). Nous avons donc défini une fonction $\varphi_f : \mathbb{N} \longrightarrow \mathbb{N}$. Remarquons que

$$\varphi_f(t) = \max\{n \in \mathbb{N} : f(n) \leq t\} = \min\{n \in \mathbb{N} : f(n) > t\} - 1$$

2 Une suite d'entiers non nuls

2.1 La fonction ω

Soit $\omega : \mathbb{N}^* \longrightarrow \mathbb{N}^*$ définie par

$$\omega(x) = \left\lceil \frac{x}{2} \right\rceil$$

Proposition 4. *Pour tout $x \in \mathbb{N}^*$,*

$$x \leq 2\omega(x) \leq x+1$$

Démonstration. Par définition de la fonction plafond,

$$\omega(x) - 1 < \frac{x}{2} \leq \omega(x)$$

et donc

$$x \leq 2\omega(x) < x+2$$

Comme $2\omega(x)$ et $x+2$ sont entiers, $\omega(x) \leq x+1$ □

Proposition 5. *Pour tout $x \geq 2$, $1 \leq \omega(x) < x$.*

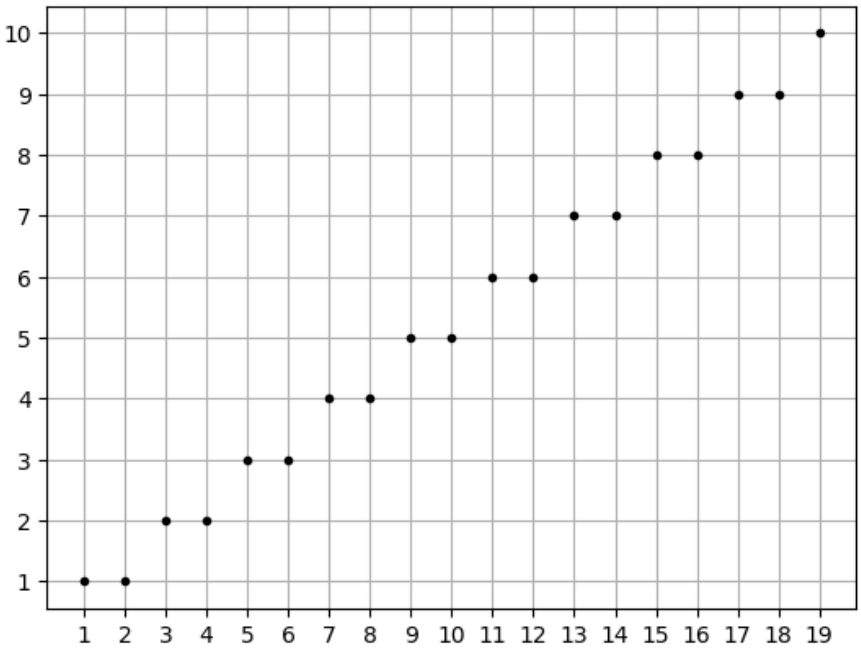

FIGURE 1 – La fonction ω

Démonstration. Soit $x \geq 2$. On a par la proposition précédente $2 \leq x \leq 2\omega(x)$ donc $1 \leq \omega(x)$. On a aussi

$$\omega(x) \leq \frac{x}{2} + \frac{1}{2} < x$$

□

2.2 Itérations

Soit $d \in \mathbb{N}^*$. On définit la suite $(d_n)_{n\geq 0}$ en posant pour tout $n \in \mathbb{N}$, $d_n = \omega^n(d)$. Par exemple, pour $d = 123$, la suite $(d_n)_{n\geq 0}$ est $(123, 62, 31, 16, 8, 4, 2, 1, 1, 1, \ldots)$.

Proposition 6. *Soit $d \in \mathbb{N}^*$. Il existe $n \in \mathbb{N}$ tel que $d_n = 1$.*

Démonstration. Supposons le contraire. On a donc pour tout $n \in \mathbb{N}$, $d_n \geq 2$. De là, par la proposition 5, $d_{n+1} = \omega(d_n) < d_n$. Ainsi, la suite $(d_n)_{n\geq 0}$ est une suite strictement décroissante d'entiers naturels, contradiction. □

Notation. Pour tout $d \in \mathbb{N}^*$, $C(d)$ est le plus petit entier naturel n tel que $d_n = 1$.

Remarquons que pour tout $d \geq 1$, et tout $n \in \mathbb{N}$, $\omega^n(\omega(d)) = \omega^{n+1}(d)$. On en déduit facilement que pour tout $d \geq 2$,

$$C(\omega(d)) = C(d) - 1$$

Proposition 7. *Pour tout $d > 1$,*

$$2^{C(d)-1} + 1 \leq d \leq 2^{C(d)}$$

Démonstration. Montrons le résultat par récurrence forte sur d. Tout d'abord, $\omega(2) = 1$ donc $C(2) = 1$. On a bien la double inégalité, qui est en fait une égalité.

Soit $d \geq 3$. Supposons que pour tout $d' \in [\![2, d]\!]$ on ait la double inégalité. Par l'hypothèse de récurrence,

$$2^{C(\omega(d))-1} + 1 \leq \omega(d) \leq 2^{C(\omega(d))}$$

et donc, puisque $C(\omega(d)) = C(d) - 1$,

$$2^{C(d)-2} + 1 \leq \omega(d) \leq 2^{C(d)-1}$$

En multipliant par 2, il vient

$$2^{C(d)-1} + 2 \leq 2\omega(d) \leq 2^{C(d)}$$

Par la proposition 4, $d \leq 2\omega(d)$. On a donc

$$d \leq 2^{C(d)}$$

On a aussi $2\omega(d) \leq d + 1$, donc

$$2^{C(d)-1} + 2 \leq d + 1$$

d'où

$$2^{C(d)-1} + 1 \leq d$$

□

Corollaire 8. *Pour tout $d > 1$,*

$$\lg d \leq C(d) \leq \lg(d-1) + 1$$

Démonstration. Il suffit de passer au logarithme en base 2 dans la proposition précédente. □

2.3 Antécédents

Proposition 9. *Soit $x \geq 2$. L'entier x a exactement deux antécédents par la fonction ω.*

Démonstration. Soit $y \in \mathbb{N}^*$. Supposons $\omega(y) = x$. On a donc

$$x - 1 < \frac{y}{2} \leq x$$

ou encore

$$2x - 2 < y \leq 2x$$

Comme x et y sont entiers, on en déduit que $y = 2x - 1$ ou $y = 2x$. Inversement, ces deux entiers ont bien x pour image par ω. □

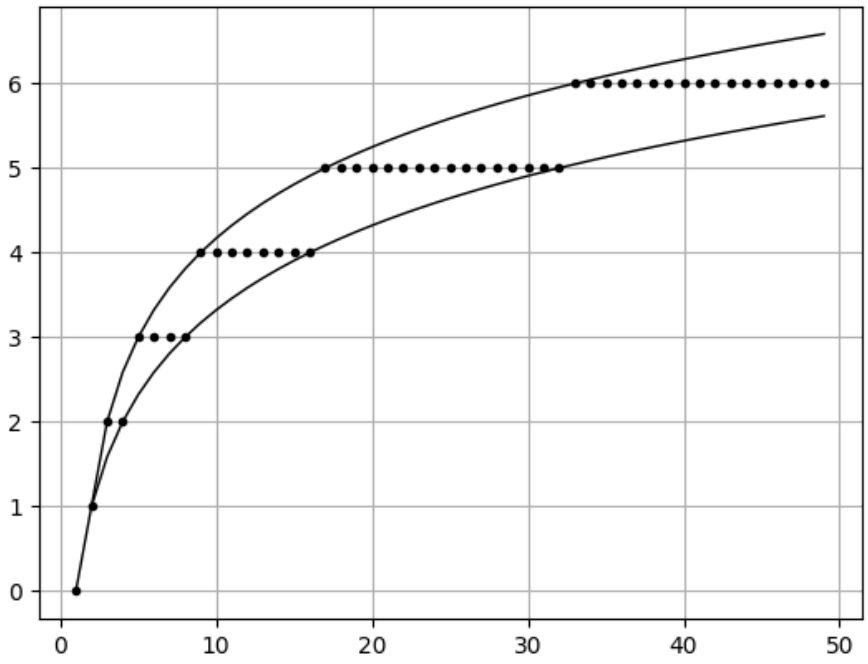

FIGURE 2 – La fonction C

3 L'algorithme

3.1 Description

La fonction φ prend en paramètres une fonction f strictement croissante et un entier naturel non nul t. Elle renvoie $\varphi_f(t)$.

```
1 def phi(f, t):
2     a = 0
3     b = t + 1
4     while b - a > 1:
5         c = (a + b) // 2
6         if f(c) <= t: a = c
7         else: b = c
8     return a
```

3.2 Terminaison et complexité

Appelons $N = N(t)$ le nombre d'évaluations de la ligne 4. Si la fonction termine, on a $N \in \mathbb{N}^*$. Sinon, $N = +\infty$. Numérotons les évaluations par les entiers de l'intervalle $[\![0, N[\![$. Pour tout $k \in [\![0, N[\![$, notons a_k et b_k les valeurs de a et b lors de l'évaluation de la ligne k. Si cette évaluation réussit, soit c_k la valeur de c après l'évaluation de la ligne 5.

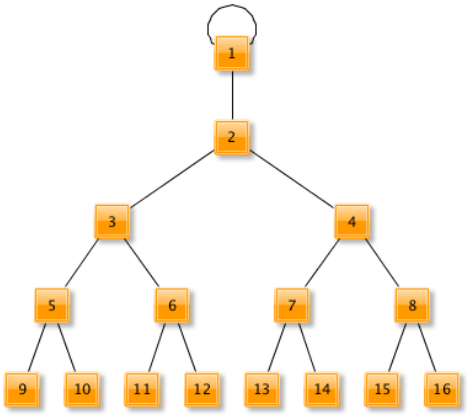

FIGURE 3 – La dynamique de la fonction ω

Proposition 10. *Pour tout $k \in [\![0, N[\![$, $a_k \leq \varphi_f(t) < b_k$.*

Démonstration. Notons $n = \varphi_f(t)$.

Montrons le résultat par récurrence sur k. Par la remarque de la section 1, $0 \leq n \leq t$, donc $a_0 \leq n < b_0$. Soit $1 \leq k < N$. Supposons que $a_{k-1} \leq n < b_{k-1}$. L'évaluation numéro $k-1$ de la ligne 4 réussit. L'entier c_{k-1} est donc défini et

$$c_{k-1} = \left\lfloor \frac{a_{k-1} + b_{k-1}}{2} \right\rfloor$$

- Cas 1, $f(c_{k-1}) \leq t$. On a alors $a_k = c_{k-1}$ et $b_k = b_{k-1}$. De là,

  $$f(a_k) \leq t < f(n+1)$$

  Comme f est strictement croissante, $a_k < n+1$, d'où $a_k \leq n$. On a aussi $n < b_{k-1} = b_k$.
- Cas 2, $f(c_{k-1}) > t$. On a alors $a_k = a_{k-1}$ et $b_k = c_{k-1}$. Immédiatement,

  $$f(a_k) = f(a_{k-1}) \leq t$$

  De plus,

  $$f(b_k) = f(c_{k-1}) > t \geq f(n)$$

  Par la croissance stricte de f, $b_k > n$.

□

Lemme 11. *Pour tous $a, b \in \mathbb{N}$ tels que $a \leq b$,*

$$b - \left\lfloor \frac{a+b}{2} \right\rfloor = \left\lceil \frac{b-a}{2} \right\rceil$$

$$\left\lfloor \frac{a+b}{2} \right\rfloor - a \leq \left\lceil \frac{b-a}{2} \right\rceil$$

Démonstration. Si a et b ont la même parité, $(a+b)/2$ et $(b-a)/2$ sont entiers et on a donc égalité. Supposons a et b de parités distinctes. Les deux entiers $a+b$ et $b-a$ sont impairs. On a alors

$$\left\lfloor \frac{a+b}{2} \right\rfloor = \frac{a+b}{2} - \frac{1}{2}$$

$$\left\lceil \frac{b-a}{2} \right\rceil = \frac{b-a}{2} + \frac{1}{2}$$

De là,

$$b - \left\lfloor \frac{a+b}{2} \right\rfloor = b - \frac{a+b}{2} + \frac{1}{2} = \frac{b-a}{2} + \frac{1}{2} = \left\lceil \frac{b-a}{2} \right\rceil$$

et

$$\left\lfloor \frac{a+b}{2} \right\rfloor - a = \frac{a+b}{2} - \frac{1}{2} - a = \frac{b-a}{2} - \frac{1}{2} = \left\lceil \frac{b-a}{2} \right\rceil - 1$$

□

Proposition 12. *Pour tout $t \in \mathbb{N}^*$, l'appel à* `phi(f, t)` *termine et*

$$N(t) \leq 1 + \lg t$$

Démonstration. Pour tout $k \in [\![0, N(t)[\![$, posons

$$\delta_k = b_k - a_k$$

On a $\delta_0 = b_0 - a_0 = t+1$. Soit $k \in [\![1, N(t)[\![$.

- Cas 1, $f(c_{k-1}) \leq t$. Par le lemme 10,

$$\delta_k = b_{k-1} - c_{k-1} = b_{k-1} - \left\lfloor \frac{a_{k-1} + b_{k-1}}{2} \right\rfloor = \left\lceil \frac{\delta_{k-1}}{2} \right\rceil$$

- Cas 2, $f(c_{k-1}) > t$. Toujours par le lemme 10,

$$\delta_k = c_{k-1} - a_{k-1} = \left\lfloor \frac{a_{k-1} + b_{k-1}}{2} \right\rfloor - a_{k-1} \leq \left\lceil \frac{\delta_{k-1}}{2} \right\rceil$$

Par une récurrence immédiate sur k, on a pour tout $k \in [\![0, N(t)[\![$, $\delta_k \leq d_k$ où $(d_k)_{k\in\mathbb{N}}$ est la suite associée à $\delta_0 = t+1$ par la méthode de la section 2. Par la proposition 6, il existe $k \in \mathbb{N}$ tel que $d_k = 1$. Rappelons que $C(t+1)$ est le plus petit tel entier. En supposant que $N(t) \geq C(t+1)$, on a alors

$$1 \leq \delta_{C(t+1)-1} \leq d_{C(t+1)-1} = 1$$

et donc

$$\delta_{C(t+1)-1} = 1$$

Le test de la ligne 4 de la fonction `phi` échoue, et donc $N(t) = C(t+1)$. Ainsi,

$$N(t) \leq C(t+1)$$

Nous avons donc montré que notre fonction termine. De plus, par le corollaire 8,

$$C(t+1) \leq \lg(t) + 1$$

et donc $N(t) \leq \lg(t) + 1$. □

3.3 Correction

Maintenant que nous avons montré la terminaison, il nous reste à prouver que la fonction `phi` renvoie ce que l'on attend.

Proposition 13. *Pour tout $t \in \mathbb{N}^*$, l'évaluation de* `phi(f, t)` *renvoie $\varphi_f(t)$.*

Démonstration. Soit $t \in \mathbb{N}^*$. Par la proposition 10,

$$a_{N(t)-1} \leq \varphi_f(t) < b_{N(t)-1} = a_{N(t)-1} + 1$$

où l'égalité ci-dessus est vraie parce que la fonction termine après $N(t)$ itérations. On a donc

$$a_{N(t)-1} = \varphi_f(t)$$

On en conclut la correction, puisque la fonction renvoie $a_{N(t)-1}$. □

4 Les limites pratiques

Si la fonction f est à croissance rapide, l'algorithme que nous venons de décrire est inefficace, voire impossible à utiliser. En effet, dans ce cas, $\varphi_f(t)$ est négligeable devant t. Avec une fonction f à croissance exponentielle, $\varphi_f(t)$ est de l'ordre du logarithme de t. En revanche, notre fonction effectue des opérations sur des nombres de la taille de $f(t)$.

Prenons un exemple concret. Soit $f : \mathbb{N} \longrightarrow \mathbb{N}$ définie par récurrence par $f(0) = 0$, $f(1) = 1$ et pour tout $n \in \mathbb{N}$, $f(n+2) = f(n+1) + f(n)$. Pour tout $n \in \mathbb{N}$, $f(n)$ est le n^e nombre de Fibonacci. On a

$$f(n) = \frac{\varphi^n - \overline{\varphi}^n}{\sqrt{5}}$$

où

$$\varphi = \frac{1+\sqrt{5}}{2} \quad \text{et} \quad \overline{\varphi} = \frac{1-\sqrt{5}}{2}$$

On vérifie facilement que

$$\varphi_f(t) \sim \log_\varphi t$$

L'utilisation de notre algorithme nécessite lors de la première itération de la boucle `while`, le calcul de $f((t+1)/2)$. Par exemple, si $t = 10^{10} - 1$, il faudrait calculer le 5 milliardième nombre de Fibonacci, ce qui est évidemment catastrophique. Et pourtant, $\varphi_f(t) = 49$, un très petit nombre.

La meilleure solution, pour ce genre de fonction f, est de ne pas utiliser l'algorithme dichotomique, mais plutôt de revenir à un algorithme naïf.

```
def phi_naif(f, t):
    a = 0
    while f(a) <= t:
        a = a + 1
    return a - 1
```

Appliquée à notre exemple, cette fonction renvoie instantanément 49.