BROUILLON - SOMMER LES CARRÉS DES CHIFFRES D'UN NATUREL

CHRISTOPHE BAL

Document, avec son source LATEX, disponible sur la page
https://github.com/bc-writing/drafts.

Table des matières

1. Faire une tête au carré à tous les entiers naturels

Voici un procédé facile à faire à l'aide d'une calculatrice.

> Considérons un entier naturel n.
>
> - Élevons chacun des chiffres de n au carré.
> - Additionnons tous ces carrés. Notons n cette somme.
> - Retournons au premier point.
>
> On peut alors étudier ce processus qui peut être infini a priori.

Voici deux exemples instructifs pour la suite.

Exemple 1.1. *Pour $n = 19$, nous obtenons :*

- $1^2 + 9^2 = 82$
- $8^2 + 2^2 = 68$
- $6^2 + 8^2 = 100$
- $1^2 + 0^2 + 0^2 = 1 \to$ *Rien de nouveau à attendre.*

Exemple 1.2. *Pour $n = 1\,234\,567\,890$, après $1^2 + 2^2 + 3^2 + 4^2 + 5^2 + 6^2 + 7^2 + 8^2 + 9^2 + 0^2 = 285$ nous obtenons :*

Date: 6 Juin 2018 – 28 Mars 2019.

- $2^2 + 8^2 + 5^2 = 93$
- $9^2 + 3^2 = 90$
- $9^2 + 0^2 = 81$
- $8^2 + 1^2 = 65$
- $6^2 + 5^2 = 61$
- $6^2 + 1^2 = 37$
- $3^2 + 7^2 = 58$
- $5^2 + 8^2 = 89$
- $8^2 + 9^2 = 145$
- $1^2 + 4^2 + 5^2 = 42$
- $4^2 + 2^2 = 20$
- $2^2 + 0^2 = 4$
- $4^2 = 16$
- $1^2 + 6^2 = 37 \rightarrow$ *Dèjà rencontré.*

Dans le 1er cas, au bout d'un moment le procédé ne produit que des 1. Ce sera par exemple le cas dès que l'on commence avec une puissance de 10. Quant au 2e exemple, il montre que le mieux que l'on puisse espérer c'est que le procédé devienne périodique à partir d'un moment *(on parle de phénomène ultimement périodique).*

On peut explorer le comportement de ce procédé sur plusieurs valeurs grâce à un programme. Voici un code possible non optimisé écrit en Python 3.7 qui prend un peu de temps pour vérifier que pour tous les naturels $n \in [\![1\,;10^6]\!]$, le procédé devient ultimement périodique.

```
NMAX    = 10**6
MAXLOOP = 10**20

for n in range(1, NMAX + 1):
    nbloops = 0
    results = []

    while nbloops < MAXLOOP and n not in results:
        nbloops += 1
        results.append(n)
        n = sum(int(d)**2 for d in str(n))

    if n not in results:
        print(f"Test raté pour n = {n}.")

print("Tests finis.")
```

Une fois lancé, le code précédent affiche juste `Tests finis`. Il reste à voir ce qu'il se passe dans le cas général. La section qui suit démontre que pour tout naturel n, le procédé sera toujours ultimement périodique.

2. Une preuve

On introduit les notations suivantes.

- Pour un naturel n, $n = [\,c_{d-1}c_{d-2}\cdots c_1c_0\,]_{10} \overset{\text{def}}{=} \sum_{i=0}^{d-1} c_i 10^i$, avec $c_{d-1} \neq 0$, désigne l'écriture décimale propre de n.
- On pose enuite $sq(n) = \sum_{i=0}^{d-1} (c_i)^2$ et $\texttt{taille}(n) = d$ sera appelé *« taille de n »*.

- Pour $(n\,;i) \in \mathbb{N}^2$, on définit $\boxed{n}_{\,0} = n$ et $\boxed{n}_{\,i} = sq^i(n) \overset{\text{def}}{=} sq \circ sq \circ \cdots \circ sq(n)$ avec $(i-1)$ compositions si $i > 0$.

  Autrement dit, nous avons $\boxed{n}_{\,0} = n$ et $\boxed{n}_{\,i+1} = sq\left(\boxed{n}_{\,i}\right)$.
- Enfin on note $V_n = \left\{ \boxed{n}_{\,i} \,|\, i \in \mathbb{N} \right\}$ l'ensemble des valeurs prises par la suite $\left(\boxed{n}_{\,i}\right)_i$.

Fait 2.1. *$\forall n \in \mathbb{N}$, $sq(n) \leqslant 81d$ où $d = \texttt{taille}(n)$.*

Preuve. *Si $n = [\,c_{d-1}c_{d-2}\cdots c_1c_0\,]_{10}$ alors* $sq(n) = \displaystyle\sum_{i=0}^{d-1}(c_i)^2 \leqslant \sum_{i=0}^{d-1} 9^2 = 81d$.

Fait 2.2. *$\forall n \in \mathbb{N}$, notant $d = \texttt{taille}(n)$, nous avons les résultats suivants :*

(1) Si $d \geqslant 4$ alors $sq(n) < n$.

(2) Si $d \leqslant 3$ alors $sq(n) < 10^3$.

Preuve. *Comme $n \geqslant 10^{d-1}$ et compte tenu du fait précédent, nous cherchons à comparer 10^{d-1} et $81d$. Pour cela, regardons ce qu'il se passe pour les premières valeurs de d.*

d	1	2	3	4	5
10^{d-1}	1	10	100	1000	10 000
$81d$	81	162	243	324	405

Or lorsque $d \geqslant 2$ augmente de 1, alors $81d$ augmente de 81 tandis que 10^{d-1} augmente de $9 \times 10^{d-1}$ soit d'au moins 90. En effet, $10^d = 10 \times 10^{d-1} = 10^{d-1} + 9 \times 10^{d-1}$.

Donc dès que $d \geqslant 4$, nous avons $n \geqslant 10^{d-1} > 81d \geqslant sq(n)$ d'où ensuite $n > sq(n)$. Ceci prouve le 1er point. [1].

Le 2nd point pour $d \leqslant 3$ découle directement de $sq(999) = 243$.

Fait 2.3. *$\forall n \in \mathbb{N}$, l'ensemble V_n est fini et donc la suite $\left(\boxed{n}_{\,i}\right)_{i\in\mathbb{N}}$ est ultimement périodique, i.e. périodique à partir d'un certain rang.*

Preuve. *Le 2nd point dépend directement du 1er point via le principe des tiroirs et la définition récursive de la suite $\left(\boxed{n}_{\,i}\right)_i$.*

Pour le 1er point, pour $n \leqslant 999$, on a directement $V_n \subset [0\,;999]$, sinon il suffit de montrer que $V_n \subset \left[0\,;10^{\texttt{taille}(n)}\right]$ pour $n \geqslant 10^4$ via une petite récurrence descendante finie.

3. Coder - Étudier la « période » d'un naturel

Quand il ne se fige pas, le code suivant donne la *« période »* d'un naturel auquel on applique le procédé présenté dans la section 1.

```
n     = 20181209
nmemo = n

results = []
```

1. Pour les fans de Nicolas BOURBAKI, voir la preuve page 6 du fait 4.2 qui traite le cas des puissances quelconques

```
while n not in results:
    results.append(n)
    n = sum(int(d)**2 for d in str(n))

print(f"{nmemo} a la période suivante :")
print(results[results.index(n):])

print()

before = results[:results.index(n)]

if before:
    print("Avant la 1ère période nous avons :")
    print(before)
else:
    print("On commence directement par la période.")
```

Le code précédent, où `n = 20181209`, nous affiche :

```
20181209 a la période suivante :
[16, 37, 58, 89, 145, 42, 20, 4]

Avant la 1ère période nous avons :
[20181209, 155, 51, 26, 40]
```

Amusons-nous maintenant à représenter un histogramme des tailles des « périodes » À l'adresse `https://github.com/bc-writing/drafts`, dans le dossier `squares-digits`, vous trouverez le fichier `squareint-sizeplots.py` qui été utilisé pour obtenir le graphique [2]. Le traitement des données a été amélioré pour éviter de refaire des calculs déjà rencontrés *(pour plus de précisions, se reporter aux commentaires du code)*. Le résultat est donné dans la figure 1 page 5.

Le graphique est frappant ! En effet, il semblerait que l'on ait soit des périodes de taille 1, penser à 0 et 1, soit des périodes de taille 8 comme pour $37-58-89-145-42-20-4-16$. Magie ou coïncidence ? Les résultats de la section 2, dont nous allons reprendre les notations, vont nous permettre de le savoir. Tout d'abord, d'après le fait 4.2, nous avons $\texttt{taille}(sq(n)) < \texttt{taille}(n)$ dès que $\texttt{taille}(n) \geqslant 4$, donc la périodicité n'arrivera que lorsque $\texttt{taille}\left(\boxed{n}_{\,k}\right) \leqslant 3$. De plus, nous savons aussi que $\texttt{taille}(sq(n)) \leqslant 3$ dès que $\texttt{taille}(n) \leqslant 3$. Tout ceci nous permet d'analyser brutalement via un programme ce qu'il se passe pour les périodes des naturels appartenant à $[\![0\,;999]\!]$. Nous pouvons pour cela utiliser le code suivant, qui n'est absolument pas optimisé mais fait le travail immédiatement.

2. À la même adresse dans le dossier `squares-digits` se trouve l'image `befores.png` qui est un histogramme des nombres de termes calculés avant l'apparition de la 1$^{\text{re}}$ *« période »*.

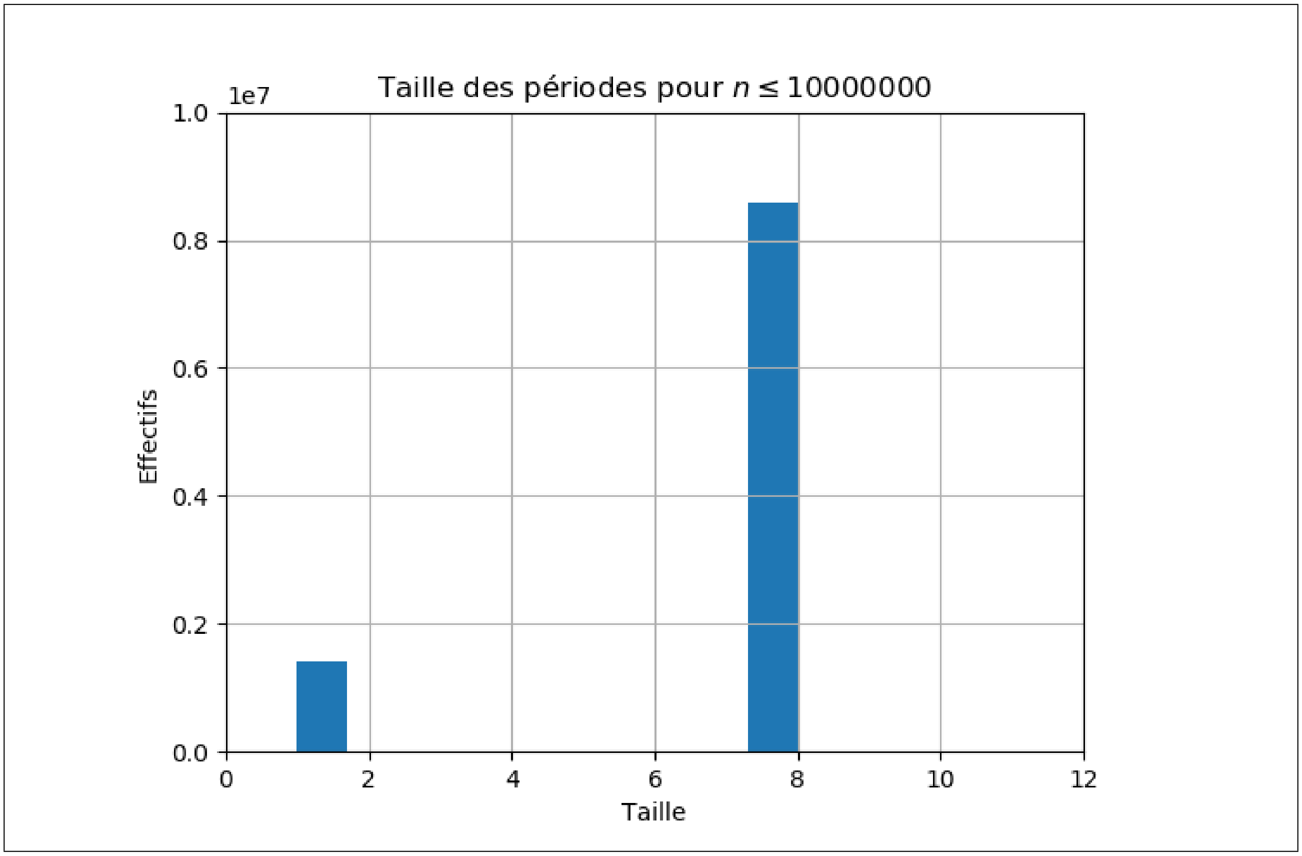

FIGURE 1: Histogramme des tailles des périodes

```
nmax = 999

periodsfound = []

for n in range(nmax + 1):
    results = []

    while n not in results:
        results.append(n)
        n = sum(int(d)**2 for d in str(n))

    period = results[results.index(n):]

    if period not in periodsfound:
        periodsfound.append(period)

for oneperiod in periodsfound:
    print(oneperiod)
```

Le code précédent nous fournit toutes les périodes possibles.

```
[0]
[1]
[4, 16, 37, 58, 89, 145, 42, 20]
[37, 58, 89, 145, 42, 20, 4, 16]
[89, 145, 42, 20, 4, 16, 37, 58]
[16, 37, 58, 89, 145, 42, 20, 4]
[20, 4, 16, 37, 58, 89, 145, 42]
[58, 89, 145, 42, 20, 4, 16, 37]
[42, 20, 4, 16, 37, 58, 89, 145]
[145, 42, 20, 4, 16, 37, 58, 89]
```

Et là cela devient joli car nous notons au passage que trois types de périodes : `[0]`, `[1]` et `[4, 16, 37, 58, 89, 145, 42, 20]` avec toutes ses *« permutées circulaires »*.

4. Peut-on généraliser à un exposant $k \geqslant 3$?

Pour finir, nous allons analyser ce qu'il se passe si l'on somme à la puissance $k \geqslant 3$ au lieu d'élever au carré. Nous reprenons des notations similaires à celles de la section 2.

- Pour un naturel $n = [\,c_{d-1}c_{d-2}\cdots c_1c_0\,]_{10}$ avec $c_{d-1} \neq 0$, on pose $pw(n) = \sum_{i=0}^{d-1} (c_i)^k$ et $\texttt{taille}(n) = d$.
- Pour $(n\,;i) \in \mathbb{N}^2$, on définit $\boxed{n}_{\,0} = n$ et $\boxed{n}_{\,i+1} = pw\left(\boxed{n}_{\,i}\right)$.

Fait 4.1. *$\forall n \in \mathbb{N}$, $pw(n) \leqslant 9^k d$ où $d = \texttt{taille}(n)$.*

Preuve. *Si $n = [\,c_{d-1}c_{d-2}\cdots c_1c_0\,]_{10}$ alors $pw(n) = \sum_{i=0}^{d-1} (c_i)^k \leqslant \sum_{i=0}^{d-1} 9^k = 9^k d$.*

Fait 4.2. *Il existe $d_0 \in \mathbb{N}$ tel que $\forall n \in \mathbb{N}$, $[\,\texttt{taille}(n) \geqslant d_0 \Rightarrow pw(n) < n\,]$.*

Preuve. *Notons $d = \texttt{taille}(n)$ de sorte que $n \geqslant 10^{d-1}$. Compte tenu du fait précédent, nous cherchons à comparer 10^{d-1} et $9^k d$.*

Nous allons procéder de façon analogue au cas $k = 2$ démontré dans la section 2 mais en étant ici plus rigoureux dans la rédaction.

$\exists d_0 \in \mathbb{N}^*$ tel que $10^{d_0-1} > 9^k d_0 > 9^{k-1}$. Ceci s'obtient en utilisant la croissance comparée des fonctions $f(x) = 10^{x-1}$ et $g(x) = 9^k x$.

Montrons par récurrence sur $d \geqslant d_0$ que $10^{d-1} > 9^k d$. Ceci donnera $n \geqslant 10^{d-1} > 9^k d \geqslant pow(n)$ d'où $n > pow(n)$ dès que $d \geqslant d_0$ comme souhaité.

- Initialisation. *Par choix de d_0 , nous avons $10^{d-1} > 9^k d$ si $d = d_0$.*
- Hérédité. *Faisons l'hypothèse que $10^{d-1} > 9^k d$ est vérifiée pour un naturel $d \geqslant d_0$* « fixé quelconque ».

  Nous avons : $10^{(d+1)-1} = 10 \times 10^{d-1} = 10^{d-1} + 9 \times 10^{d-1} > 10^{d-1} + 9^k$ en utilisant au passage $10^{d-1} \geqslant 10^{d_0-1} > 9^{k-1}$.

  Comme $10^{d-1} > 9^k d$, nous avons ensuite $10^{(d+1)-1} > 9^k d + 9^k = 9^k(d+1)$. L'inégalité est donc vérifiée au rang suivant $(d+1)$.

- Conclusion. *Par récurrence sur* $d \geqslant d_0$ *, nous avons* $10^{d-1} > 9^k d$ *pour tout naturel* d *tel que* $d \geqslant d_0$ *.*

Remarque 4.1. *Informatiquement une valeur de* d_0 *peut s'obtenir en testant* $10^{d-1} > 9^k d$ *successivement pour les naturels non nuls* d*.*

Pour gagner du temps, on peut tester les valeurs successives de 2^i *pour* $i = 0, 1, 2, \dots$ *pour obtenir* D *tel que* $10^{D-1} > 9^k D$ *. Si la valeur de* D *est trop grande pour faire des tests brutaux, on peut chercher la valeur minimale de* d *tel que* $10^{d-1} > 9^k d$ *en utilisant une recherche de type dichotomique.*

Fait 4.3. $\forall n \in \mathbb{N}$*, la suite* $\left(\boxed{n}_{\,i}\right)_{i\in\mathbb{N}}$ *est ultimement périodique.*

Preuve. *Tout est en fait contenu dans le fait 4.2, dont on reprend la signification de* d_0*. Expliquons pourquoi.*

- *Le fait 4.2 donne l'existence d'un indice* $i_0 \in \mathbb{N}$ *tel que* $\texttt{taille}\left(\boxed{n}_{\,i_0}\right) < d_0$ (dans le cas contraire, on pourrait construire une suite strictement décroissante de naturels).
- *Si pour tout naturel* $i \in [\![i_0 \,; +\infty [\![$ *,* $\texttt{taille}\left(\boxed{n}_{\,i}\right) < d_0$ *, nous avons l'ultime périodicité via le principe des tiroirs* (si besoin revoir la fin de la section 2).
- *Sinon il existe* $i'_0 \in]\!] i_0 \,; +\infty [\![$ *tel que* $\texttt{taille}\left(\boxed{n}_{\,i'_0}\right) \geqslant d_0$*. Comme dans le premier point, nous pouvons alors trouver* $i_1 \in]\!] i'_0 \,; +\infty [\![$ *tel que* $\texttt{taille}\left(\boxed{n}_{\,i_1}\right) < d_0$*.*
- *En répétant notre raisonnement, on peut aboutir à une situation similaire au 2*^e^ *point, et c'est gagné.*
  Sinon on arrive à construire une suite strictement croissante $(i_k)_k$ *d'indices tels que* $\forall k \in \mathbb{N}$, $\texttt{taille}\left(\boxed{n}_{\,i_k}\right) < d_0$*. Le principe des tiroirs s'applique ici aussi !*

Remarque 4.2. *La preuve précédente montre que pour rechercher toutes les périodes il* « suffit » *d'étudier les naturels appartenant à* $[\![0 \,; 10^{d_0} [\![$ *.*

---

5. AFFAIRE À SUIVRE...
