# TIPE : Nombre de Frobenius

Marc FERSZTAND

31 octobre 2021

## Table des matières

# 1 Un problème de la vie courante (Introduction)

Avec uniquement des pièces de 2 et 5 centimes il est possible de faire l'appoint pour $2, 4, 5, 6, 7, 8$, ou $9, ...$ centimes mais il est impossible de faire l'appoint pour 1 ou 3 centimes.

Plus généralement, "Soit $n \in \mathbb{N}$ et $a_1, \ldots, a_n$ des entiers. Sachant que l'on dispose d'une quantité illimité de pièces de valeurs $a_1, \ldots, a_n$, quel est le plus grand montant pour lequel il est impossible de payer comptant ?"

De manière plus formelle "quel est le plus grand entier qui n'est pas combinaison linéaire positive entière des $a_i$ ?"

Malgré l'apparente simplicité de cette question, on ne connaît pas de formule générale et on doit se contenter d'algorithmes calculant la solution.

# 2 Corps principal

## 2.1 modalités de l'action

Dans une première partie théorique j'ai étudié les preuves de la validité de deux algorithmes : celui de Heap et Lynn [4] et celui de Nijenhuis-Dijkstra [2]. J'ai ensuite comparé leurs complexités respectives.
Dans une seconde partie pratique, j'ai implémenté ces deux algorithmes en Python. Les programmes informatiques ainsi réalisés m'ont permis de mesurer de façon expérimentale les temps réels d'exécution et de les comparer aux hypothèses théoriques de complexité temporelle. Je m'en suis ensuite servi pour vérifier une approximation empirique du nombre de Frobenius. Enfin je me suis posé la question suivante : "Qui est un nombre de Frobenius ?".

## 2.2 Résultats théoriques

Soit $0 < a_1 < ... < a_n$ des entiers tels que $pgcd(a_1, ..., a_n) = 1$

### 2.2.1 Premières propriétés

**Définition.** *On dit que k est* ***représentable*** *si*

$$\exists(\lambda_1, ..., \lambda_n) \in \mathbb{N}^n, \sum_{i=1}^{n} \lambda_i a_i = k$$

**Enoncé.** *Le nombre de Frobenius $g = g(a_1, ..., a_n)$ est* ***le plus grand entier non représentable***

**Exemple.** $n = 2$, $(a_1, a_2) = (3, 5)$ *avec en vert les nombres représentables et en rouge ceux qui ne le sont pas :* 1 2 3 4 5 6 **7** 8 9 10 11 12 13 14 15
$g(3, 5) = 7$

On dispose d'une formule dans le cas $n = 2$ :

**Proposition 1** (Formule de Sylvestre)**.** *Soit* $(a, b) \in \mathbb{N}^{*2}$ *tel que* $pgcd(a, b) = 1$ *alors*

$$g(a, b) = ab - a - b$$

### 2.2.2 Algorithme de Heap et Lynn

Considérons la matrice carrée $B \in \mathcal{M}_{a_n}(\mathbb{R})$ de dimension $a_n \times a_n$ définie par :

$$\forall 1 \leqslant i, j \leqslant a_n,\ b_{i,j} = \begin{cases} 1, \text{si } j = i + 1 \text{ ou si } i + 1 - j \in \{a_1, \ldots, a_n\} \\ 0, \text{sinon} \end{cases}$$

On peut lui associer un graphe $G(B)$ dont l'ensemble des sommets est $\{1, \ldots, a_n\}$ avec une arête de $i$ à $j$ si et seulement si $b_{i,j} > 0$

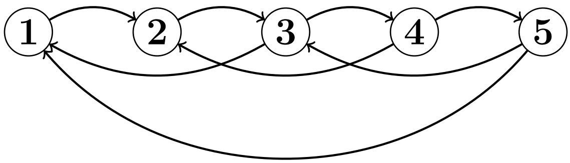



FIGURE 1 – $G(B)$ pour $n = 2, a_1 = 3, a_2 = 5$

En utilisant le lien entre $B$ et $G(B)$, on démontre [4] que :

**Proposition 2** (Principe de l'algorithme)**.** *$B$ est primitive d'indice de primitivité défini par* $\gamma := \min\{m \in \mathbb{N}^*, B^m > 0\}$ *vérifie :*

$$g(a_1, \ldots, a_n) = \gamma - a_n$$

Mon programme calcule $\gamma$ avec $O(\log(\gamma))$ multiplications de matrices, soit une complexité totale de $O(a_n^3 \log(\gamma)) = O(a_n^3 \log(a_n))$.

### 2.2.3 Algorithme de Nijenhuis-Dijkstra

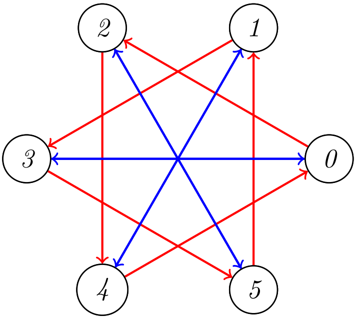
On étudie le graphe pondéré orienté $G = (V, E)$

— de sommets :$V = \{0, 1, \ldots, a_1 - 1\}$

— d'arêtes : $E = \{(u,v) \mid \exists k \in [\![1,n]\!], u - v \equiv a_k \pmod{a_1}\}$

— de poids $w(u,v) = a_k$ si $u - v \equiv a_k \pmod{a_1}$.

**Exemple.** $a_1 = 6, a_2 = 9, a_3 = 20$ *(rouge de poids* $20$ *; bleu de poids* $9$*)*




Soit $v \in [\![0, a_1 - 1]\!]$ et $S_v$ le poids d'un plus court chemin de 0 vers $v$ alors pour $M \equiv v \pmod{a_1}$, on peut montrer [2] que :

$$\text{M représentable} \Leftrightarrow M \geqslant S_v$$

Puis en déduire que :

**Proposition 3.**

$$g(a_1, \ldots, a_n) = \max_{0 \leqslant v < a_1} S_v - a_1$$

Pour la recherche du plus court chemin, j'ai choisi d'utiliser l'algorithme de Dijkstra. Avec l'utilisation d'un tas binaire sa complexité est en $O(|E|\log(|V|)) = O(na_1 \log(a_1))$.

## 2.3 Résultats expérimentaux et leur analyse

### 2.3.1 Complexités temporelles des deux algorithmes

**Résultats** J'ai exécuté mes deux algorithmes pour un nombre maximal de valeurs et j'ai affiché leur temps d'éxecution moyen dans les figures 2 et 3 :

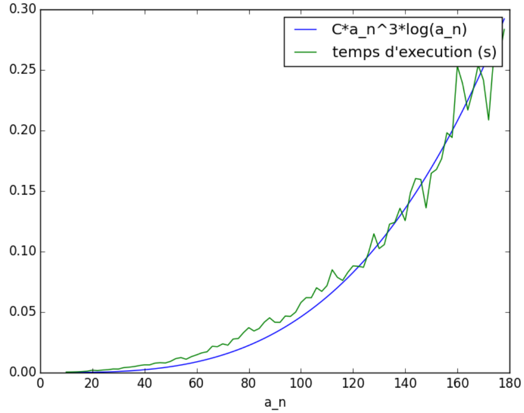



FIGURE 2 – Temps d'exécution de l'algorithme de Heap et Lynn pour 125 valeurs inférieures à 180

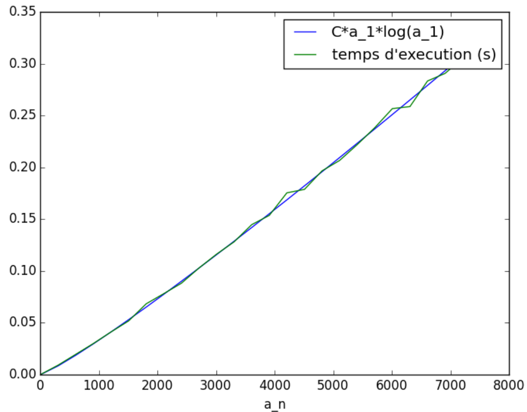



FIGURE 3 – Temps d'exécution de l'algorithme Dijkstra pour 100 valeurs inférieurs à 8000

Table 1 – Comparaison des deux algorithmes

| | Heap et Lynn | Nijenhuis-Dijkstra |
|---|---|---|
| Complexité temporelle | $O(a_n^3 \log(a_n))$ | $O(na_1 \log(a_1))$ |
| Complexité spatiale | $O(a_n^2 \log(a_n))$ | $O(na_1)$ |
| Valeurs pour 0.3 s | $a_n \leqslant 180$ | $a_1 \leqslant 7500$ |
| Valeurs pour 1 min | $a_n \leqslant 500$ | $a_1 \leqslant 500000$ |

**Analyse-Exploitation-Discussion**

— Le second algorithme est beaucoup plus efficace que le premier.

— Les complexités théoriques dans le pire des cas correspondent aux temps moyens d'exécution mesurés.

### 2.3.2 Approximation de $g(a_1, \ldots, a_n)$

**Résultats** J'ai choisi de vérifier informatiquement une approximation de $g + \sum a_i$ donnée par[2]

**Conjecture.**

$$g(a_1, \ldots, a_n) + \sum a_i \approx (\frac{n!}{2} \prod_{i=1}^{n} a_i)^{\frac{1}{n-1}} := L(a_1, \ldots, a_n)$$

En utilisant l'algorithme de Nijenhuis-Dijkstra, j'ai calculé 5000 valeurs du rapport

$$R(n) := \frac{g(a_1, \ldots, a_n) + \sum_{i=1}^{n} a_i}{L(a_1, \ldots, a_n)}.$$

Les valeurs de $R(n)$ sont regroupées dans le tableau 2 et les figures 4 et 5.

— Pour $n = 3$

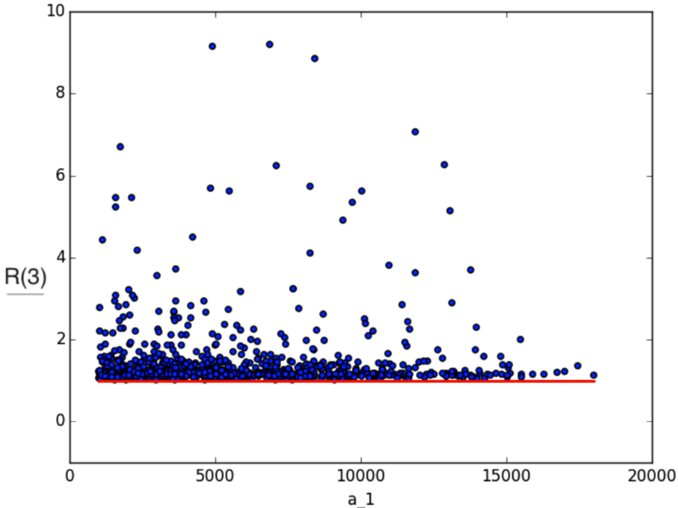



FIGURE 4 – $R(3)$ pour 1000 valeurs entre 1000 et 20000

— pour $n = 4$

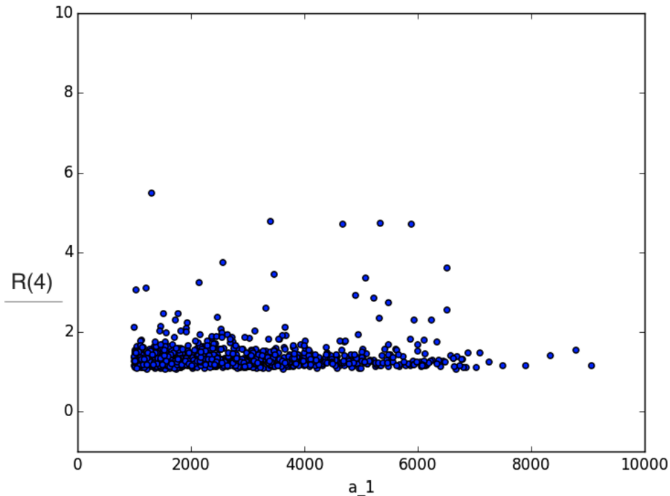



FIGURE 5 – $R(4)$ pour 1000 valeurs entre 1000 et 10000

Table 2 – Etude de $R(n)$ avec 500 à 1000 points par valeurs de $n$

| n | moyenne : $\overline{r}$ | écart relatif : $\frac{\sigma(r)}{\overline{r}}$ | minimum |
|---|---|---|---|
| 3 | 1,46 | 0,9 | 1,009 |
| 4 | 1,36 | 0,25 | 1,08 |
| 5 | 1,32 | 0,13 | 1,13 |
| 6 | 1,32 | 0,07 | 1,17 |
| 7 | 1,35 | 0,07 | 1,21 |
| 8 | 1,37 | 0,03 | 1,23 |
| 9 | 1,40 | 0,03 | 1,3 |
| 10 | 1,44 | 0,02 | 1,38 |

**Analyse-Exploitation-Discussion**

- En moyenne $R(n) \approx 1.4$
- $n = 3$ : rappelons que $R(a_1, a_2, a_3) = \frac{g(a_1,a_2,a_3)+a_1+a_2+a_3}{\sqrt{3a_1a_2a_3}}$.
  Conformément aux résultats démontrés par Davison[3], j'ai remarqué $R(3) \geqslant 1$ et que le coefficient 1 est optimal.
- $n = 4$ et $n = 8$ : Ces cas sont traités dans [2]. Les valeurs obtenues pour la moyenne diffèrent de moins de 2% des miennes.

### 2.3.3 Qui peut être un nombre de Frobenius ?

Les articles de la bibliographie cherchent à déterminer le nombre de Frobenius d'un ensemble donné réciproquement quels entiers peuvent être des nombres de Frobenius ? J'ai envisagé deux versions de cette question :

- Version 1 : $n$ quelconque

  Soit $m \in \mathbb{N}^*$, existe t-il $n \in \mathbb{N}^*$ et $a_1, \ldots, a_n$ entiers tels que $m = g(a_1, \ldots, a_n)$ ? La solution est donnée par la propriété suivante :

  **Proposition 4.**

  $$\forall m \in \mathbb{N}^*, \exists n \in \mathbb{N}^* \exists (a_1, \ldots, a_n) \in \mathbb{N}^{*n}, \begin{cases} pgcd(a_1, \ldots, a_n) = 1 \\ m = g(a_1, \ldots, a_n) \end{cases}$$

  **Démonstration.** — *vrai pour* $m = 1, 2, 3, 4$

  — *si* $m \geqslant 5$. *Alors* $g(m-2, m-1) = (m-3)(m-2) - 1 \geqslant m$ *donc*

  $$g(\{m-2, m-1\} \cup \{x > m, x \ non\ atteignable\}) = m$$

- Version 2 : $n$ fixé

  Soit $(m, n) \in \mathbb{N}^{*2}$, existe-t-il $a_1, \ldots, a_n$ entiers tels que $m = g(a_1, \ldots, a_n)$

— Si $n = 2$ En appliquant la formule de Sylvestre j'ai également redémontré que :

**Proposition 5.**

$$m \text{ est un nombre de Frobenius} \Leftrightarrow m \text{ est impair}$$

— Si $n = 3$ (qui implique le cas général)

**Conjecture.** *tout nombre est un nombre de Frobenius .*

J'ai testé informatiquement cette conjecture pour $m \leqslant 1000$. R. Alfonsin m'a finalement indiqué une démonstration[5] de cette conjecture que je n'ai pas étudiée.

# 3 Conclusion

La théorie des graphes nous fournit deux algorithmes pour calculer le nombre de Frobenius. Le plus efficace (Nijenhuis-Dijkstra) a une complexité temporelle en $O(na_1 \log(a_1))$. J'ai implémenté ces deux algorithmes que j'ai utilisés pour étudier asymptotiquement le nombre de Frobenius et pour déterminer quels entiers étaient des nombres de Frobenius.

# 4 Références

# Références


[1] J.L.R. Alfonsín. *The Diophantine Frobenius Problem.* Oxford Lecture Mathematics and. OUP Oxford, 2005.

[2] A. Nijenhuis D. Beihoffer, J. Hendry and S. Wagon. Faster algorithms for frobenius numbers. *The Electronic Journal of Combinatorics*, 12(27), June 2005.

[3] J.L. Davison. On the linear diophantine problem of frobenius. *Journal of Number Theory*, 48(3) :353–363, 1994.

[4] B. R. Heap and M. S. Lynn. On a linear diophantine problem of frobenius : An improved algorithm. *Numer. Mathematics*, 7(3) :226–231, June 1965.

[5] J. C. Rosales, P. A. García-Sánchez, and J. I. García-García. Every positive integer is the frobenius number of a numerical semigroup with three generators. *MATHEMATICA SCANDINAVICA*, 94(1) :5–12, Mar. 2004.