# 1 Problème: génération de nombres pseudo-aléatoires, une méthode peu conventionnelle...

Dans ce problème, nous allons proposer une méthode nouvelle de génération de nombres pseudo-aléatoires.
On construit une suite $(u_n)_{n\in\mathbb{N}}$ par récurrence de la manière suivante:

$$\forall n \in \mathbb{N}, u_{n+1} = \overline{\text{bin}(u_n)}^{10}$$

et avec $u_0 \in \mathbb{N}$.
Dans un soucis de simplicité, nous appelerons $u_0$ la *graine* de cette suite.
Par exemple, pour une graine de 4, on obtient:

$$u_0 = 4$$

$$u_1 = \overline{\text{bin}(u_0)}^{10} = \overline{100}^{10} = 100$$

$$u_2 = \overline{\text{bin}(u_1)}^{10} = 1100100$$

$$u_3 = \overline{\text{bin}(u_2)}^{10} = 100001100100101000100$$

$$u_4 = 1010110101111001011010001101011011001100110110011001100101110100100$$

$$\vdots$$

On observe que cette suite à tendance à grandir très vite !

## 1.1 Questions préliminaires

On notera $|u_n|$ la nombre de chiffres dans l'écriture en base 10 $u_n$
Par exemple, $|4| = 1, |10020| = 5$, etc.
Attention ! $|00000| = |0| = 1$, $|010| = |10| = 2$

**Q1. Donner un encadrement de $|u_n|$ en fonction de $u_n$. Montrer que $|u_n| \approx \log_2(u_n)$.**

*Afin de ne pas commettre de barbarisme mathématique, on dira que, soit $(v_n)_{n\in\mathbb{N}}$ et $(\omega_n)_{n\in\mathbb{N}}$ des suites, $v_n \approx \omega_n$ si $\frac{v_n}{\omega_n} \underset{n\to\infty}{\to} 1$*

**Q2. Etudier la limite de la suite $u_n$ en fonction de la graine**

**Q3. On observe/admet que**

$$\forall n \in \mathbb{N}, \quad u_{n+1} \approx 10^{|u_n|}$$

**Montrer que:**

$$\forall u_0 \in \mathbb{N}_{\geq 2}, \forall n \in \mathbb{N}, \quad |u_n| \approx u_0 \log_2(10)^n$$

**Q4. Expliquer comment la croissance exponentielle de $|u_n|$ entraine une limite à l'utilisation de cette méthode en C**

## 1.2 Implémentation

Afin de résoudre ce problème, nous allons "sectionner" le nombre obtenu, en ne gardant que les $\delta \in \mathbb{N}$ premiers chiffres à chaque étape, en commençant par le deuxième. On appelera ce nombre *l'indice de sectionnement* de l'algorithme.
Par exemple, pour $u_n = 10100010100101001$ et $\delta = 4$, on gardera seulement 0100

On appelera $\delta$-compatible tout entier qui peut s'écrire à l'aide de $\delta$ chiffres et ne comportant que des 0 ou des 1
Par exemple, pour $\delta = 5$, 10100 ou 10 sont $\delta$-compatible. En revanche, 1020 ou 100001000100010 ne le sont pas.

**Question bonus: Pourquoi commence-t-on par le deuxième caractère et non pas le premier ?**

**Q5. Trouver le $\delta \in \mathbb{N}$ qui maximise le nombre obtenu (afin d'avoir un aléatoire de "bonne qualité") tout en prenant en compte la limite exposée Q4**

*Rappel:* $2^{31} - 1 = 2147483647$ *(cette valeur est à apprendre par coeur !)*

**Q6. Ecrire une fonction bin prenant en argument un entier, supposé positif, et renvoyant une chaine de caractère représentant sa décomposition binaire. En donner la complexité.**

**Q7. Ecrire une fonction extract prenant en argument une chaine de caractère et un entier $\delta$, et renvoyant les $\delta$ premiers caractères à partir du deuxième de la chaine passée en argument. Si la chaine n'est pas assez grande, retourner la chaine sans modifications. En donner la complexité et prouver sa correction.**

**Q8. Ecrire une fonction aleatoire, prenant en argument un entier $\alpha$, un entier $\delta$ et une entier $\aleph$ et renvoyant l'entier généré "aléatoirement" par notre méthode avec pour graine $\alpha$ et pour indice de sectionnement $\delta$, en l'itérant $\aleph$ fois (c'est à dire, $(\text{aleatoire}(\alpha, \delta))^{\aleph}$ au sens de $\circ$).**

*Explication alternative de $\aleph$:*

$$\text{aleatoire}(\alpha, \delta, \aleph) = \text{aleatoire}(\text{aleatoire}(\cdots \text{aleatoire}(\alpha, \delta, 1), \delta, 1))$$

*ou bien:*

$$\text{aleatoire}(\alpha, \delta, \aleph) = \text{aleatoire}(\text{aleatoire}(\alpha, \delta, 1), \delta, \aleph - 1))$$

## 1.3 Vérifications

Afin d'évaluer la qualité des algorithmes de génération de pseudo-aléatoire, il peut être utile d'étudier les cycles qui existent au sein de ceux-ci.
On peut démontrer par la méthode des tiroirs et des chaussettes qu'il existe au moins un cycle de taille maximale. En réalité, un tel cycle est presque impossible à atteindre, et l'algorithme produira plutôt beaucoup de petits cycles qui ne sont pas reliés. On peut assimiler cela à la notion de spé d'**attracteur** et de **piège** (qui ne sera très rapidement abordée ici).

On appelle **cycle** toute famille $(x_0, \cdots, x_n)$ d'entiers $\delta$-compatibles tels que

$$\forall i \in [|0; n-1|], \quad \text{aleatoire}(x_i, \delta, 1) = x_{i+1}$$

et $x_0 = x_n$

Par exemple, si on a:

$$\text{aleatoire}(4, \delta, 1) = 1010$$
$$\text{aleatoire}(1010, \delta, 1) = 1110$$
$$\text{aleatoire}(1110, \delta, 1) = 1000$$
$$\text{aleatoire}(1000, \delta, 1) = 1010$$

Alors $(1010, 1110, 1000)$ forme un cycle.

On appelle **piège** toute famille $x_0 \cdots x_n$ d'entiers $\delta$-compatibles tels que

$$\forall \aleph \in \mathbb{N}, \forall i \in [|0; n|], \exists j \in [|0; n|] \quad \text{aleatoire}(x_i, \delta, \aleph) = x_j$$

On note Pr l'ensemble des entiers $\delta$-compatibles qui appartiennent à un piège.

On appelle **attracteur** tout entier $x$ $\delta$-compatible tels que

$$\exists n \in \mathbb{N}, \quad \text{aleatoire}(x, \delta, n) \quad \text{appartienne à un piège}$$

On note Atr l'ensemble des entiers $\delta$-compatibles qui sont attracteurs.

On observe que tout entier $\delta$-compatible appartenant à un piège est un attracteur.
On définit alors la notion **d'attracteur stricte** comme étant tout attracteur qui n'appartient pas

à un piège.
On note AtrS l'ensemble des entiers $\delta$-compatibles qui sont des attracteurs strictes.

**Q9. Montrer que pour toute famille d'entiers $\delta$-compatibles $(x_0 \cdots x_n)$,**

$$(x_0 \cdots x_n) \textbf{ est un cycle} \Longleftrightarrow (x_0 \cdots x_n) \textbf{ est un piège}$$

**Q10. Montrer que pour tout entiers $\delta$-compatibles $x$,**

$$x \textbf{ n'appartient pas à un un cycle} \Longrightarrow x \textbf{ est un attracteur stricte}$$

**En déduire que**

$$\textbf{Pr} \bigsqcup \textbf{AtrS} = \{x \in \mathbb{N} | x \quad \delta\textbf{-compatible}\} = \mathbb{N}_\delta \quad \textbf{(notation)}$$

**Q11. Montrer que pour tout $x \in \mathbb{N}_\delta \setminus \{0; 1\}$ ,**

$$|x| < \delta \Longrightarrow x \textbf{ est un attracteur stricte.}$$

*Indication: la stricte croissance d'$u_n$ pour une graine $> 2$ peut servir...*

**Q12. Donner une majoration de Card(Pr) en fonction de Card($\mathbb{N}_\delta$) et $\delta$**

**Q13. Ecrire une fonction cycle, prenant en argument un entier $\delta$-compatible, supposé positif, et renvoyant la taille du cycle que forme cet entier par l'algorithme de notre "nouvelle méthode"**

**Q14. Ecrire une fonction max, renvoyant la graine maximisant la taille du cycle pour des nombres $\delta$-compatibles**