Début d'article du Zero surprenant

On a un programme python qui stabilise une configuration 3x3
on a un programme python qui ajoute 2 tableaux et stabilise commander config 3 par 3
travail sur un programme qui stabilise une config 4 par 4 et un autre qui ajoutent et stabilise
test au hasard déconfits 4 par 4 pour peut-être tomber sur le 0

plan de l'exposé, de l'article

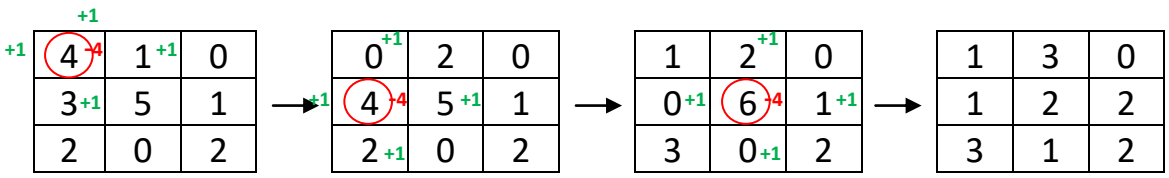
A. Environnement du sujet
   1. Vocabulaire

Définition d'une configuration :
Une **configuration** est un tableau avec n lignes et n colonnes de nombres entiers naturels.

Définition configuration stable :
Une **configuration stable** est une configuration qui ne contient que des chiffres strictement inférieurs à 4 : donc que des 0-1-2-3.

Règle de stabilisation : Si une case contient 4 ou plus, alors cette case donne une unité à chacune de ses quatre voisines (dessus, dessous, à droite et à gauche). Les unités qui sortent de la configuration sont perdues.

Exemple à la main de la stabilisation d'une configuration :



Algorithme Python de stabilisation : ANNEXE fichier Edupython « stabilise config 3-3 et compte les étapes »

```
import numpy as Np
def stabilise():          #stabilise une config 3*3 en la mettant dans une config 5*5#
    c=0
    a = Np.array([[6,6,6],[6,4,6],[6,6,6]])        #config initiale 3*3#
    b= Np.array([[0,0,0,0,0],[0,0,0,0,0],[0,0,0,0,0],[0,0,0,0,0],[0,0,0,0,0]])
    #config full0 dans laquelle on va mettre config initiale#
    print (a)
    for i in range(0,3):            #balaie la ligne i#                        #on met config initiale à l'interieur de full0#
        for j in range(0,3):        #balaie la colonne j#
            b[i+1][j+1]=a[i][j]

    while b[1][1]>3 or b[1][2]>3 or b[1][3]>3 or b[2][1]>3 or b[2][2]>3 or b[2][3]>3 or b[3][1]>3 or b[3][2]>3 or b[3][3]>3:
        #pour faire balayer l'intérieur de b tant qu'il y a des nombres sup à 4#
        for i in range(1,4):            #balaie la ligne i#
            for j in range(1,4):         #balaie la colonne j#
                if b[i][j]>3:             #on stabilise : distribution de cacahuètes aux voisins#
                    b[i][j]=b[i][j]-4
                    b[i-1][j]=b[i-1][j]+1
                    b[i+1][j]=b[i+1][j]+1
                    b[i][j-1]=b[i][j-1]+1
                    b[i][j+1]=b[i][j+1]+1
                    c= c+1            #compteur du nombre d'étapes pour stabiliser#


    for i in range(1,4):            #balaie la ligne i#
        for j in range(1,4):        #balaie la colonne j#
            a[i-1][j-1]=b[i][j]         #on met l'intérieur de b dans a#
    print(a)  #config stabilisée#
    print(c)   #nombre d'étapes#

stabilise()
```

Exemple avec notre programme pour vérifier la config

```
*** Console de processus distant Réinitialisée ***
>>>
[[6 6 6]
 [6 4 6]
 [6 6 6]]
[[2 1 2]
 [1 0 1]
 [2 1 2]]
45
>>>
```

Propriété : Il y a 262 144 configurations stables.

Démonstration : Il y a 9 cases avec 4 possibilités (0, 1, 2 ou 3) par case : donc $4^9 = 262\ 144$ configurations stables différentes.

2. Restriction du domaine de recherche : configurations lourdes

Notations : On notera full 0, (respectivement full 3, full a -où a appartient $\mathbb{N}$) une configuration ne contenant que des 0 (respectivement 3, a).

Règle d'addition de deux configurations :
On peut naturellement additionner deux configurations stables entre elles, en :
1. additionnant case par case le contenu,
2. appliquant la règle de stabilisation jusqu'à obtenir de nouveau une configuration stable.

Définition configuration lourde :
Une configuration lourde est l'addition d'une configuration stable avec un full3.

Exemple à la main pour obtenir une configuration lourde :

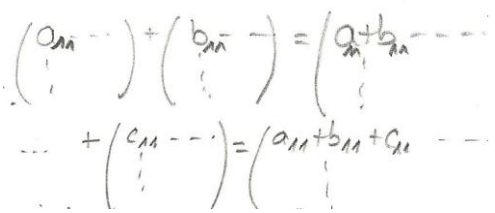
3	3	3		0	0	0		3	3	3				1	3	1
3	3	3	+	0	1	0	=	3	4	3	$\rightarrow \cdots \rightarrow$			3	0	3
3	3	3		0	0	0		3	3	3				1	3	1

(La dernière configuration tout à droite est lourde.)

Remarque : $\mathbb{Z}$ et l'addition forment un groupe commutatif. En effet :
①quand on additionne deux entiers relatifs, on obtient un entier relatif
②associativité
③il y a un zéro
④chaque entier relatif a un opposé : 2 et -2, 3 et -3…
⑤commutativité

Propriété : L'ensemble des configurations lourdes et l'addition forment un groupe commutatif
En effet :
① La somme de deux configurations lourdes est une configuration lourde.
② En notant, a, b et c trois configurations lourdes, (a+b)+c=a+(b+c)
③ Il existe un élément zéro z tel que pour toute configuration lourde a, a+z=a
④ Chaque configuration lourde a admet une configuration lourde opposé b telle que a+b=z :
⑤ En notant a et b deux configurations lourdes, a+b=b+a (commutativité)

Démonstrations :



②associativité
a, b, c 3 lourdes
(a+b) +c = a + (b+c)
Évident car somme de tableau ?
utilise associativité de Z et +
①somme interne
a Appartient au groupe des lourdes ⟺ a+z=a
b Appartient au groupe des lourdes ⟺ b+z=b
est ce que a+b appartient au groupe des lourdes ?
(a+b) + z
= a + (b+z) (associativité)
= a + b
③ ADMIS existence de zéro z tel que +/a lourde, a+z = a ?
en 3x3 (212)
(101)
(212)
④ ADMIS existence d'un inverse tel que +/a lourde, ∃ b lourde telle que a+b=z
⑤commutativité a+b = b+a ? par commutativité de Z dans le tableau !

B. Le sujet

Trouver le zéro z d'une configuration 3×3, 4×4, 5×5, $n \times n$

Pour trouver aléatoirement le zéro des configurations ?
On devrait avec un algorithme, faire balayer toutes les configurations en 4*4 possibles (4^{16} configurations). Pour chacune, on l'ajoute à un Full3 (notée a).
On l'ajoute à elle-même : a+a= b. On vérifie si b=a. Dans ce cas, a est notre zéro.

Durée pour faire ce travail par Python ?
En 3*3 :
On a constaté une moyenne d'étapes de 23-24.
On suppose une moyenne de temps pour 1 étape : 0,00008s

Programme stabilisation d'un tableau aléatoire en 3*3 puis en 4*4 et qui compte le nombre d'étapes
Qui tourne 1000 fois
Et fait la moyenne du nb d'étapes.

On aurait une fréquence du nombre d'étapes par stabilisation sur un échantillon de 1000 stabilisation

En 4*4 :
On suppose une moyenne d'étapes : 59-60 (car sur des full3 on a vu max 90 étapes, le plus souvent 80)
On suppose une moyenne de temps pour 1 étape : 0,00008s
Donc l'ordinateur fait 23600 étapes en
23600 x 2 = 0.00008 secondes
Il fait donc une possibilité de tableau en 3 secondes
il y a en tout 4^{16}, 4 chiffres possibles (0, 1, 2, 3) et dans un tableau 4x4 il y a 16 cases
Donc si le 0 cherché est le dernier à être testé afin de le trouver il nous faudra : 4^{16}x3 = 1.288490189 e10 s
Soit 408 ans 210 jours 19 heures 24 minutes et 47, 93184 secondes.

Pour nous aider, le chercheur nous a donné le zéro z des configurations en 3×3.

2	1	2
1	0	1
2	1	2

Propriété :
La configuration ci-dessus est le zéro z des configurations en 3×3.

Démonstration ?

Propriété : Le zéro z est lourd.

Démonstration de z est lourde :
Il faut trouver une configuration qui ajoutée à Full3 fasse notre zéro z.
333
333 + ? = z
333
Et si on créait un programme qui balaie les 262 144 possibilités en 3x3 ?
Inutile, on est tombé par hasard sur :
333
313 qui est l'origine du zéro. En effet :
333

333 333 666 212
333 + 313 = 646 = … = 101 =z Yes ! donc z est bien lourde.
333 333 666 212

Propriété : Full3 est lourde.

Démonstration de Full3 est lourde :
Full3 + Full0 = Full3 -> Ok full 3 est bien lourde.

Propriété :
Le Zéro des configurations n× n est unique.

Unicité du zéro ?
Notons z et z' deux zéros de nos configurations. Alors pour toute configuration lourde a, on a :
a+z=a et a+z'=a.
Pour z', la dernière égalié donne : z'+z=z'
Or z'+z = z+z'=z donc z=z'.

Propriété : Full0 n'est pas lourde.

Démonstration ?
Sinon Full0 serait notre 0. Du coup, nous aurions deux zéros. Or cela est impossible. Donc Full0 n'est pas lourde.

on s'est dit qu'il fallait trouver la forme des configurations lourdes mais cela nous a semblé dur et déconseillé par le chercheur. Recherchez un critère pour savoir si une config stable était lourde.
Théorème (admis) une configuration et lourdes si et seulement si c + z = c

Propriété : dans Z 0 de lourdes, il ne peut pas y avoir deux 0 adjacents
et on a même :
propriété : dans les lourdes , il ne peut y avoir deux 0 adjacents
Démo
Questions : le 0 est-il unique ? oui démo ok
obtenir 2 lourdes identique à partir de 2 config inégale ? Oui cela nous est déjà arrivé !
Ordre des étapes de stabilisation a-t-il une importance ?
Inverse de 0 est-il 0 ?
faire les démos de propriété du groupe de lourdes muni +