Les tours de Hanoï

Année 2020-2021

Autrice : Emma DEBRAY (Terminale générale).
Établissement : Lycée Paul Guérin, Niort (79).
Encadrés par : Fabien AOUSTIN, Thomas FORGET.
Chercheur : Abdallah EL HAMIDI, Laboratoire des Sciences de l'Ingénieur pour l'Environnement, LaSIE, UMR CNRS 7356, La Rochelle Université.

Dans cet article, on s'intéresse au jeu appelé « tours de Hanoï ». On détermine le nombre de coups minimum pour résoudre le jeu, on donne un programme qui donne la solution optimale et on établit quelques statistiques. On s'intéresse ensuite à plusieurs variantes du jeu où certains mouvements sont interdits.

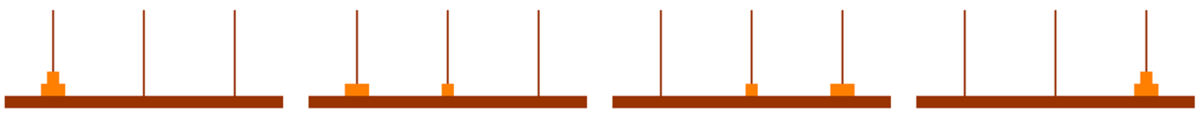
1) Présentation du jeu :

Le jeu des tours de Hanoï a été inventé par le jeune mathématicien Édouard Lucas (1842-1891) en 1883. Il serait inspiré de la tour sacrée du Brahma où on trouvait 64 disques en or que des prêtres déplaçaient en permanence selon les mêmes règles que le jeu. On raconte que la fin du monde coïnciderait avec le dernier déplacement de disque ; en déplaçant un disque par seconde, il se sera écoulé environ 585 milliards d'années, soit plusieurs fois l'âge estimé de notre univers.
Voici les règles du jeu. On dispose de trois piquets. Sur le premier piquet, on place un nombre n d'anneaux. Chaque anneau a en dessous de lui un anneau plus grand. Le but est de déplacer cette pile de n anneaux jusqu'au troisième piquet en respectant les deux contraintes suivantes :
- on ne peut déplacer qu'un seul anneau à la fois ;
- un anneau ne peut pas être posé sur un autre anneau plus petit.

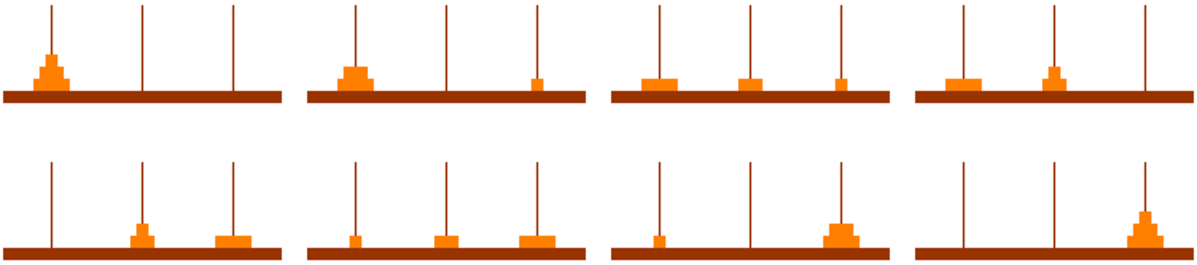
2) Résolution de la version classique du jeu :

2-a) Nombre de coups optimal :

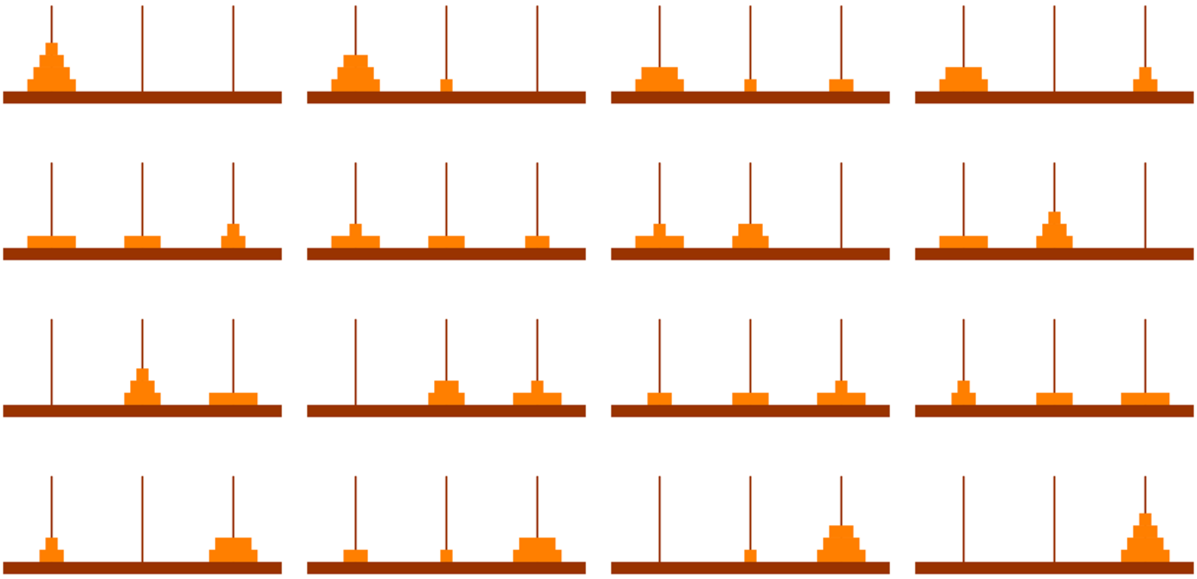
Nous avons tout d'abord tenté de résoudre le jeu avec un petit nombre n d'anneaux.
Pour $n = 1$, il n'y a évidemment qu'un seul coup à jouer.
Pour $n = 2$, on peut résoudre le jeu en 3 coups.



Pour $n = 3$, on peut résoudre le jeu en 7 coups.



Pour $n = 4$, on peut résoudre le jeu en 15 coups.



Démontrons par récurrence que le nombre de coups minimal pour résoudre le jeu avec n anneaux est égal à $2^n - 1$.

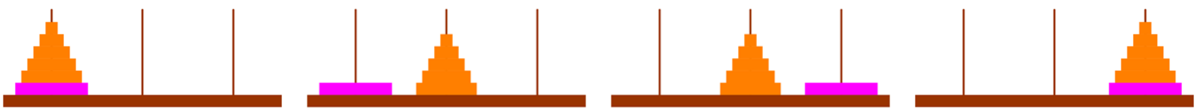
Initialisation :

Le nombre de déplacements pour un anneau est égal à 1 et $2^1 - 1 = 2 - 1 = 1$ donc la récurrence est fondée au rang 1.

Hérédité :

Supposons que pour un certain entier n supérieur ou égal à 1, le nombre minimal de coups à jouer pour déplacer les n anneaux d'un piquet à l'autre soit égal à $2^n - 1$. Montrons alors que le nombre minimal de déplacements pour $n + 1$ anneaux est égal à $2^{n+1} - 1$.

Avec $n + 1$ anneaux, on doit d'abord déplacer n anneaux du piquet de gauche au piquet du milieu. Cela nécessite $2^n - 1$ déplacements. On peut alors déplacer le plus grand anneau du piquet de gauche au piquet de droite ce qui nécessite 1 déplacement. Il faut ensuite redéplacer les n anneaux du piquet du milieu vers le piquet de droite ce qui nécessite $2^n - 1$ déplacements (1).



Au final, le nombre de déplacement est égal à :

$$(2^n - 1) + 1 + (2^n - 1) = 2^n + 2^n - 1 = 2^{n+1} - 1.$$

On a démontré que la propriété est héréditaire.

Conclusion :

Finalement, quel que soit l'entier n supérieur ou égal à 1, le nombre minimal de déplacements pour résoudre le jeu est bien égal à $2^n - 1$.

2-b) Description des solutions optimales :

On va noter chaque déplacement de la façon suivante (2) :
- *a* représente le déplacement du piquet 1 vers le piquet 2 ;
- *A* représente le déplacement du piquet 2 vers le piquet 1 ;
- *b* représente le déplacement du piquet 1 vers le piquet 3 ;
- *B* représente le déplacement du piquet 3 vers le piquet 1 ;
- *c* représente le déplacement du piquet 2 vers le piquet 3 ;
- *C* représente le déplacement du piquet 3 vers le piquet 2.

Les solutions trouvées précédemment se notent donc ainsi :
- pour $n = 1$, on a le mot-solution : *b*.
- pour $n = 2$, on a le mot-solution : *abc*.
- pour $n = 3$, on a le mot-solution : *baCbAcb*.
- pour $n = 4$, on a le mot-solution : *abcaBCabcABcabc*.

On peut alors écrire un programme qui prend en entrée :
- le nombre n d'anneaux ;
- le numéro du piquet de départ ("piquetdepart")
- le numéro du piquet d'arrivée ("piquetdarrivee")
et qui renvoie le mot décrivant la solution la plus courte, en $2^n - 1$ coups.

On commence par donner les six cas avec un seul anneau.
Ensuite, si le nombre d'anneaux est plus grand que 1, on écrit :
- le mot-solution qui décrit le déplacement des $n - 1$ plus petits anneaux du piquet de départ vers le piquet intermédiaire ;
- la lettre qui correspond au déplacement du grand anneau du piquet de départ vers le piquet d'arrivée ;
- le mot-solution qui décrit le déplacement des $n - 1$ plus petits anneaux du piquet intermédiaire vers le piquet d'arrivée.

Voici le programme écrit dans le langage Python.

```python
def tourdehanoi(n,piquetdepart,piquetdarrivee):
    if n==1:
        if piquetdepart==1 and piquetdarrivee==2:
            return('a')
        if piquetdepart==2 and piquetdarrivee==1:
            return('A')
        if piquetdepart==1 and piquetdarrivee==3:
            return('b')
        if piquetdepart==3 and piquetdarrivee==1:
            return('B')
        if piquetdepart==2 and piquetdarrivee==3:
            return('c')
        if piquetdepart==3 and piquetdarrivee==2:
            return('C')
    else:
        if piquetdepart==1 and piquetdarrivee==2:
            return(tourdehanoi(n-1,1,3)+'a'+tourdehanoi(n-1,3,2))
        if piquetdepart==2 and piquetdarrivee==1:
            return(tourdehanoi(n-1,2,3)+'A'+tourdehanoi(n-1,3,1))
        if piquetdepart==1 and piquetdarrivee==3:
            return(tourdehanoi(n-1,1,2)+'b'+tourdehanoi(n-1,2,3))
        if piquetdepart==3 and piquetdarrivee==1:
            return(tourdehanoi(n-1,3,2)+'B'+tourdehanoi(n-1,2,1))
        if piquetdepart==2 and piquetdarrivee==3:
            return(tourdehanoi(n-1,2,1)+'c'+tourdehanoi(n-1,1,3))
        if piquetdepart==3 and piquetdarrivee==2:
            return(tourdehanoi(n-1,3,1)+'C'+tourdehanoi(n-1,1,2))
```

On a par exemple :

```
>>> tourdehanoi(8,1,3)
'abcaBCabcABcabcaBCaBcABCabcaBCabcABcabcABCaBcABcabcaBCabcABcabcaBCaBcABCabcaBCaBcABcabcABCaBcAB
CabcaBCabcABcabcaBCaBcABCabcaBCabcABcabcABCaBcABcabcaBCabcABcabcABCaBcABCabcaBCaBcABcabcABCaBcA
BcabcaBCabcABcabcaBCaBcABCabcaBCabcABcabcABCaBcABcabcaBCabcABcabc'
>>> len(tourdehanoi(8,1,3))
255
>>> tourdehanoi(8,1,3).count('a')
58
```

2-c) Quelques statistiques intéressantes :

En comptant le nombre d'apparitions de chaque mouvement dans les solutions optimales pour déplacer les n anneaux du piquet 1 vers le piquet 3, on a le tableau suivant (3) :

n	Nombre de coups	Nombre d'apparitions du					
		mouvement a	mouvement b	mouvement c	mouvement A	mouvement B	mouvement C
1	1	0	1	0	0	0	0
2	3	1	1	1	0	0	0
3	7	1	3	1	1	0	1
4	15	4	3	4	1	2	1
5	31	4	9	4	6	2	6
6	63	15	9	15	6	12	6
7	127	15	31	15	27	12	27
8	255	58	31	58	27	54	27
9	511	58	117	58	112	54	112
10	1 023	229	117	229	112	224	112
11	2 047	229	459	229	453	224	453

12	4 095	912	459	912	453	906	453
13	8 191	912	1 825	912	1 818	906	1 818
14	16 383	3 643	1 825	3 643	1 818	3 636	1 818
15	32 767	3 643	7 287	3 643	7 279	3 636	7 279
16	65 535	14 566	7 287	14 566	7 279	14 558	7 279
17	131 071	14 566	29 133	14 566	29 124	14 558	29 124
18	262 143	58 257	29 133	58 257	29 124	58 248	29 124
19	524 287	58 257	116 515	58 257	116 505	58 248	116 505
20	1 048 575	233 020	116 515	233 020	116 505	233 010	116 505
21	2 097 151	233 020	466 041	233 020	466 030	233 010	466 030
22	4 194 303	932 071	466 041	932 071	466 030	932 060	466 030
23	8 388 607	932 071	1 864 143	932 071	1 864 131	932 060	1 864 131
24	16 777 215	3 728 274	1 864 143	3 728 274	1 864 131	3 728 262	1 864 131
25	33 554 431	3 728 274	7 456 549	3 728 274	7 456 536	3 728 262	7 456 536
26	67 108 863	14 913 085	7 456 549	14 913 085	7 456 536	14 913 072	7 456 536

En calculant les fréquences d'apparitions des mouvements, on a les résultats suivants :

n	Nombre de coups	Fréquences d'apparition du					
		mouvement a	mouvement b	mouvement c	mouvement A	mouvement B	mouvement C
1	1	0	1	0	0	0	0
2	3	0,3333333333	0,3333333333	0,3333333333	0	0	0
3	7	0,1428571429	0,4285714286	0,1428571429	0,1428571429	0	0,1428571429
4	15	0,2666666667	0,2	0,2666666667	0,6666666667	0,1333333333	0,6666666667
5	31	0,1290322581	0,2903225806	0,1290322581	0,1935483871	0,6451612903	0,1935483871
6	63	0,2380952381	0,1428571429	0,2380952381	0,9523809524	0,1904761905	0,9523809524
7	127	0,1181102362	0,2440944882	0,1181102362	0,2125984252	0,9448818898	0,2125984252
8	255	0,2274509804	0,1215686275	0,2274509804	0,1058823529	0,2117647059	0,1058823529
9	511	0,1135029354	0,228962818	0,1135029354	0,2191780822	0,1056751468	0,2191780822
10	1 023	0,2238514174	0,1143695015	0,2238514174	0,1094819159	0,2189638319	0,1094819159
11	2 047	0,1118710308	0,2242305813	0,1118710308	0,2212994626	0,1094284319	0,2212994626
12	4 095	0,2227106227	0,1120879121	0,2227106227	0,1106227106	0,2212454212	0,1106227106
13	8 191	0,1113417165	0,2228055183	0,1113417165	0,2219509217	0,1106092052	0,2219509217
14	16 383	0,2223646463	0,1113959592	0,2223646463	0,1109686871	0,2219373741	0,1109686871
15	32 767	0,11117893	0,2223883786	0,11117893	0,2221442305	0,1109653005	0,2221442305
16	65 535	0,2222629129	0,1111924926	0,2222629129	0,1110704204	0,2221408408	0,1110704204
17	131 071	0,1111306086	0,2222688467	0,1111306086	0,2222001816	0,111069573	0,2222001816
18	262 143	0,2222336664	0,1111339994	0,2222336664	0,111099667	0,222199334	0,111099667
19	524 287	0,1111166212	0,2222351498	0,1111166212	0,2222160763	0,1110994551	0,2222160763
20	1 048 575	0,2222254011	0,1111174689	0,2222254011	0,1111079322	0,2222158644	0,1111079322
21	2 097 151	0,1111126476	0,222225772	0,1111126476	0,2222205268	0,1111078792	0,2222205268
22	4 194 303	0,2222230964	0,1111128595	0,2222230964	0,1111102369	0,2222204738	0,1111102369
23	8 388 607	0,111111535	0,2222231891	0,111111535	0,2222217586	0,1111102237	0,2222217586
24	16 777 215	0,2222224606	0,1111115879	0,2222224606	0,1111108727	0,2222217454	0,1111108727
25	33 554 431	0,111111227	0,2222224838	0,111111227	0,2222220964	0,1111108694	0,2222220964
26	67 108 863	0,2222222868	0,1111112403	0,2222222868	0,1111110465	0,2222220931	0,1111110465

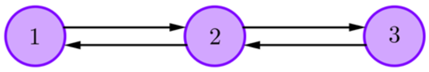
On peut conjecturer que si n est pair, alors :
- les fréquences d'apparition des mouvements a, B et c convergent vers $2/9$;
- les fréquences d'apparition des mouvements A, b et C convergent vers $1/9$.

On peut aussi conjecturer que si n est impair, les résultats sont échangés.

3) Étude de certaines variantes du jeu :

3-a) La variante « à petits pas » :

On modifie maintenant le jeu de la façon suivante. Le but est toujours de déplacer n anneaux du premier au troisième piquet sauf que les coups directs entre le piquet 1 et le piquet 3 sont interdits. On autorise seulement les coups entre les piquets 1 et 2 et entre les piquets 2 et 3.



On peut résoudre ce nouveau jeu à la main pour de petites valeurs de n.
On peut aussi modifier le programme précédent.

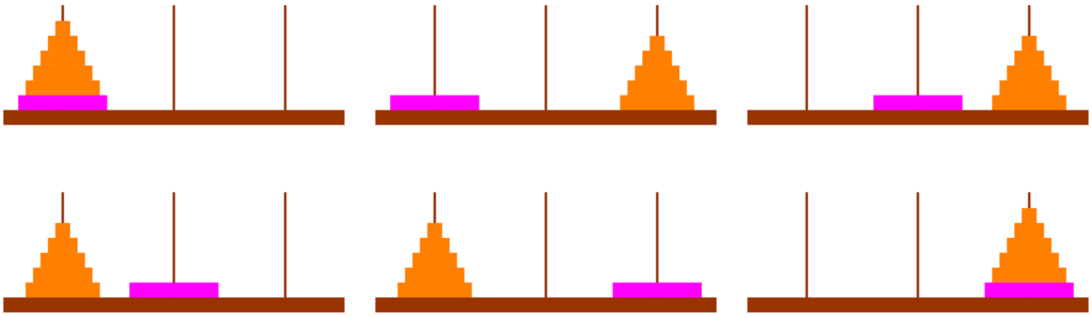
```python
def tourdehanoisansbB(n,piquetdepart,piquetdarrivee):
    if n==1:
        if piquetdepart==1 and piquetdarrivee==2:
            return('a')
        if piquetdepart==2 and piquetdarrivee==1:
            return('A')
        if piquetdepart==1 and piquetdarrivee==3:
            return('ac')
        if piquetdepart==3 and piquetdarrivee==1:
            return('CA')
        if piquetdepart==2 and piquetdarrivee==3:
            return('c')
        if piquetdepart==3 and piquetdarrivee==2:
            return('C')
    else:
        if piquetdepart==1 and piquetdarrivee==2:
            return(tourdehanoisansbB(n-1,1,3)+'a'+tourdehanoisansbB(n-1,3,2))
        if piquetdepart==2 and piquetdarrivee==1:
            return(tourdehanoisansbB(n-1,2,3)+'A'+tourdehanoisansbB(n-1,3,1))
        if piquetdepart==1 and piquetdarrivee==3:
            return(tourdehanoisansbB(n-1,1,3)+'a'+tourdehanoisansbB(n-1,3,1)+'c'+tourdehanoisansbB(n-1,1,3))
        if piquetdepart==3 and piquetdarrivee==1:
            return(tourdehanoisansbB(n-1,3,1)+'C'+tourdehanoisansbB(n-1,1,3)+'A'+tourdehanoisansbB(n-1,3,1))
        if piquetdepart==2 and piquetdarrivee==3:
            return(tourdehanoisansbB(n-1,2,1)+'c'+tourdehanoisansbB(n-1,1,3))
        if piquetdepart==3 and piquetdarrivee==2:
            return(tourdehanoisansbB(n-1,3,1)+'C'+tourdehanoisansbB(n-1,1,2))
```

On obtient alors les résultats suivants :

n	Nombre de déplacements minimum
1	2
2	8
3	26
4	80
5	242
6	728
7	2 186
8	6 560

On observe que pour passer d'un terme à l'autre, il faut le multiplier par 3 et ajouter 2.
Cela s'explique de la façon suivante. On note u_n le nombre minimum de coups à jouer pour déplacer les n anneaux du piquet 1 au piquet 3.

Pour déplacer $n + 1$ anneaux, on doit d'abord déplacer n anneaux du piquet 1 vers le piquet 3, ce qui consiste à effectuer u_n déplacements.
On peut alors déplacer le dernier anneau vers le piquet 2, soit 1 déplacement.
Puis on repasse les n anneaux du piquet 3 vers le piquet 1, soit u_n déplacements.
Ensuite, on peut déplacer le dernier anneau du piquet 2 vers le piquet 3, soit 1 déplacement.
Enfin, on peut déplacer les n anneaux du piquet 1 vers le piquet 3, soit u_n déplacements.



Finalement, on a bien : $u_{n+1} = 3u_n + 2$.

La suite (u_n) est une suite arithmético-géométrique.
On pose $v_n = u_n + 1$. Pour cette suite auxiliaire, on a :

$$\begin{aligned} v_{n+1} &= u_{n+1} + 1 \\ &= 3u_n + 2 + 1 \\ &= 3u_n + 3 \\ &= 3(u_n + 1) \\ &= 3 \times v_n \end{aligned}$$

La suite (v_n) est donc une suite géométrique de raison 3.
On a : $v_1 = u_1 + 1 = 2 + 1 = 3$. Donc $v_n = v_1 \times q^{n-1} = 3 \times 3^{n-1} = 3^n$ d'où $3^n = u_n + 1$ et $u_n = 3^n - 1$.

On peut aussi démontrer cette formule par récurrence.

Initialisation :
Le nombre de déplacements pour un anneau est égal à 2 (d'abord du piquet 1 vers le piquet 2, puis du piquet 2 vers le piquet 1) et $3^1 - 1 = 3 - 1 = 2$ donc la récurrence est fondée au rang 1.

Hérédité :
Supposons que pour un certain entier n supérieur ou égal à 1, $u_n = 3^n - 1$.
Montrons alors que $u_{n+1} = 3^{n+1} - 1$. On a :

$$\begin{aligned} u_{n+1} &= 3u_n + 2 \\ &= 3(3^n - 1) + 2 \\ &= 3 \times 3^n - 3 + 2 \\ &= 3^{n+1} - 1. \end{aligned}$$

On a démontré que la propriété est héréditaire.

Conclusion :
Finalement, quel que soit l'entier n supérieur ou égal à 1, le nombre minimal de déplacements pour résoudre la variante à petits pas est bien égal à $3^n - 1$.

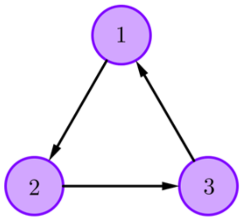
3-b) Remarque sur la variante « à petits pas » :

On peut compter le nombre de positions différentes correctes pour disposer les anneaux.
Pour chaque anneau, il y a trois piquets possibles. Une fois qu'on sait quels anneaux se trouvent sur un piquet, il suffit de les empiler du plus grand au plus petit.
On peut représenter la situation par un n-uplet avec 3 choix (les 3 piquets pour les n anneaux).
D'après le principe multiplicatif, il y a 3^n positions différentes correctes.

Avec le résultat trouvé précédemment (le nombre de coups minimum pour déplacer les n anneaux du piquet 1 au piquet 3 avec les déplacements interdits entre les piquets 1 et 3), on remarque qu'on passe par toutes les positions : il y a $3^n - 1$ déplacements à effectuer ce qui donne $3^n - 1 + 1 = 3^n$ positions avec la position initiale (4).

3-c) La variante cyclique :

Dans cette variante, on n'autorise ici que les mouvements du piquet 1 vers le piquet 2, du piquet 2 vers le piquet 3 et du piquet 3 vers le piquet 1.



On a cette fois-ci deux suites. On note :

- w_n le nombre de déplacements pour aller du piquet k au piquet $k+2$ en allant dans le même sens (de 1 vers 3, de 2 vers 1 ou de 3 vers 2) ;
- z_n le nombre de déplacements pour aller du piquet k au piquet $k+1$ en allant dans le même sens (de 1 vers 2, de 2 vers 3 ou de 3 vers 1).

À la main, on trouve les résultats suivants :

n	w_n	z_n
1	2	1
2	7	5
3	21	15
4	59	

On observe que pour $n = 4$ on fait $2 \times w_3 + 1 \times z_3 + 2$.
De façon générale, on a : $w_n = w_{n-1} + 1 + z_{n-1} + 1 + w_{n-1} = 2w_{n-1} + z_{n-1} + 2$.
On observe aussi que : $z_n = w_{n-1} + 1 + w_{n-1} = 2w_{n-1} + 1$ et donc : $z_{n-1} = 2w_{n-2} + 1$.
Ainsi, on a :

$$\begin{aligned} w_n &= 2w_{n-1} + z_{n-1} + 2 \\ &= 2w_{n-1} + 2w_{n-2} + 1 + 2 \\ &= 2w_{n-1} + 2w_{n-2} + 3. \end{aligned}$$

On a une suite récurrente d'ordre 2.

On cherche une suite constante C qui vérifie cette relation.
On a alors $C = 2C + 2C + 3$ et donc $C = -1$.

On pose alors $W_n = w_n - (-1) = w_n + 1$. On a donc aussi $w_n = W_n - 1$.
On a alors, de $w_n = 2w_{n-1} + 2w_{n-2} + 3$:

$$\begin{aligned} W_n - 1 &= 2(W_{n-1} - 1) + 2(W_{n-2} - 1) + 3 \\ &= 2W_{n-1} - 2 + 2W_{n-2} - 2 + 3 \\ &= 2W_{n-1} + 2W_{n-2} - 1 \\ W_n &= 2W_{n-1} + 2W_{n-2}. \end{aligned}$$

Si on suppose que (W_n) est géométrique, on a $W_n = W_0 \times q^n$ et :

$$\begin{aligned} W_n &= 2W_{n-1} + 2W_{n-2} \\ W_0 \times q^n &= 2W_0 \times q^{n-1} + 2W_0 \times q^{n-2} \\ q^n &= 2q^{n-1} + 2q^{n-2} \\ q^2 \times q^{n-2} &= 2q \times q^{n-2} + 2q^{n-2} \\ q^2 &= 2q + 2 \\ 0 &= -q^2 + 2q + 2. \end{aligned}$$

On a : $\Delta = 2^2 - 4 \times (-1) \times 2 = 12$.

Il y a deux solutions : $q_1 = \frac{(-2-\sqrt{12})}{-2} = 1 + \sqrt{3}$ et $q_2 = 1 - \sqrt{3}$

On a alors : $W_n = \alpha(1+\sqrt{3})^n + \beta(1-\sqrt{3})^n$ (5).

Pour trouver les valeurs de α et β on utilise :

$$\begin{aligned} W_1 &= \alpha(1+\sqrt{3}) + \beta(1-\sqrt{3}) = w_1 + 1 = 2 + 1 = 3 \\ W_2 &= \alpha(1+\sqrt{3})^2 + \beta(1-\sqrt{3})^2 = w_2 + 1 = 7 + 1 = 8. \end{aligned}$$

On résout maintenant le système :

$$\begin{cases} \alpha(1+\sqrt{3}) + \beta(1-\sqrt{3}) & = 3 \quad \times (1+\sqrt{3}) \\ \alpha(1+\sqrt{3})^2 + \beta(1-\sqrt{3})^2 & = 8 \end{cases} \Leftrightarrow \begin{cases} \alpha(1+\sqrt{3})^2 + \beta(1-3) & = 3+\sqrt{3} \\ \alpha(1+\sqrt{3})^2 + \beta(1-\sqrt{3})^2 & = 8 \end{cases}$$

En soustrayant la lignes (1) de la ligne (2) on trouve :

$$\begin{aligned} (1-\sqrt{3})^2\beta - (1-3)\beta &= 8 - (3+\sqrt{3}) \\ (1-2\sqrt{3}+3)\,\beta + 2\beta &= 5 - 3\sqrt{3} \\ (1-2\sqrt{3}+3+2)\,\beta &= 5 - 3\sqrt{3} \\ (6-2\sqrt{3})\,\beta &= 5 - 3\sqrt{3} \\ \beta = \frac{5-3\sqrt{3}}{6-2\sqrt{3}} &= \frac{3-2\sqrt{3}}{6}. \end{aligned}$$

En reprenant la première relation, on a :

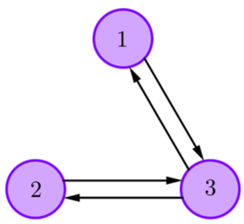
$$\begin{aligned} (1+\sqrt{3})\alpha + (1-\sqrt{3}) \times \frac{3-2\sqrt{3}}{6} &= 3 \\ (1+\sqrt{3})\alpha + \frac{9-5\sqrt{3}}{6} &= 3 \\ (1+\sqrt{3})\alpha &= \frac{9+5\sqrt{3}}{6} \\ \alpha &= \frac{3+2\sqrt{3}}{6} \end{aligned}$$

Finalement, on a :

$$w_n = W_n - 1 = \frac{3+2\sqrt{3}}{6}(1+\sqrt{3})^n + \frac{3-2\sqrt{3}}{6}(1-\sqrt{3})^n - 1.$$

3-d) Sans échanges entre les piquets 1 et 2 :

Dans cette variante, on supprime les mouvements entre les piquets 1 et 2.



On prend deux suites différentes :

- v_n est le nombre de mouvements pour aller du piquet 1 au piquet 3 avec n anneaux sans échanges directs entre les piquets 1 et 2 ;
- u_n est la suite utilisée pour trouver le nombre de déplacements minimum en interdisant les échanges directs entre les piquets 1 et 3.

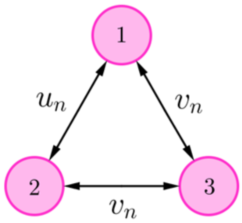
On a déjà vu que $u_n = 3^n - 1$.

En trouvant les premiers résultats à la main pour 1, 2 ou 3 anneaux, on observe que :

$$v_n = u_{n-1} + 1 + v_{n-1}.$$

En effet, pour déplacer les n anneaux du piquet 1 au piquet 3, il faut :

– déplacer $n-1$ anneaux du piquet 1 au piquet 2 sans échanges entre les piquets 1 et 2, ce qui revient au même que d'appliquer la variante "à petit pas", soit u_{n-1} déplacements ;
– déplacer le grand anneau du piquet 1 au piquet 3, soit 1 déplacement ;
– déplacer les $n-1$ anneaux du piquet 2 au piquet 3 sans échanges entre les piquets 1 et 2, ce qui revient au même que de les déplacer du piquet 1 au piquet 3, soit v_{n-1} déplacements.



On a donc :

$$v_n = 3^{n-1} - 1 + 1 + v_{n-1} = 3^{n-1} + v_{n-1},$$

$$\begin{aligned} v_1 &= 1 \\ v_2 &= 3 + 1 \\ v_3 &= 3^2 + 3 + 1 \\ v_4 &= 3^3 + 3^2 + 3 + 1 \end{aligned}$$

et donc :

$$\begin{aligned} v_n &= 1 + 3 + 3^3 + 3^2 + \cdots + 3^{n-1} \\ &= \frac{1 - 3^n}{1 - 3} \\ &= \frac{3^n - 1}{2}\ . \end{aligned}$$

On remarque qu'on trouve la moitié du nombre minimum de déplacements en interdisant les échanges entre les piquets 1 et 3.

Bibliographie :

Pour la présentation du jeu :
– Le beau livre des maths, Clifford A. Pickover, Dunod, 2019.
Pour les suites récurrentes d'ordre 2 :
– *Récurrence linéaire et épidémie de bosse des maths*, Gilles Cohen, Tangente Hors-série n° 76, novembre 2020.

Notes d'édition

(1) Il existe d'autres façons de déplacer les anneaux mais cette stratégie le réalise effectivement en un nombre minimal de coups, et c'est la seule. En effet, il faut déplacer au moins une fois l'anneau le plus grand ; il faut pour cela déplacer avant les n autres anneaux sur un autre piquet, et il faut les déplacer à nouveau sur le piquet d'arrivée après y avoir mis le grand anneau.
Pour des raisons analogues, les stratégies données pour les variantes avec certains déplacements interdits sont aussi optimales.

(2) On pourrait introduire cette notation par une fonction Python de façon à éviter l'énumération des cas dans le programme.

(3) Ici, on compte les nombres d'apparitions des différents mouvements dans les solutions fournies par le programme (au moyen de l'instruction tourdehanoi(n,1,3).count()). Mais on peut remarquer qu'on peut aussi les obtenir plus directement par des équations de récurrence linéaires déduites de la construction.

(4) Il est clair qu'on ne repasse jamais par la même position du fait que la stratégie optimale ne peut pas comporter de boucles (on le voit aussi à la façon dont la solution est construite). Les 3^n positions visitées sont donc toutes différentes et on trouve bien toutes les positions possibles.

(5) En effet, quels que soient α et β les suites (W_n) et $\left(\alpha(1+\sqrt{3})^n + \beta(1-\sqrt{3})^n\right)$ satisfont la même équation de récurrence d'ordre 2 ; en choisissant les valeurs de α et β telles que leurs deux premiers termes coïncident, on obtient par récurrence que leurs termes de même indice n sont égaux pour tout n.