Un partage sans fin ou sans faim ?

Année 2021 – 2022

Théo Carcassier, Salomé Déves, Pierre Graviou, Inès Hemery,
Nabil Kodo, Adrien Marion-Soyer, élèves en classe de seconde.

Établissement: Lycée Marguerite de Navarre, Bourges

Enseignant·es : Nathalie Herminier, Olivier Créchet

Chercheur : Benjamin Nguyen, INSA Centre-Val de Loire

1. PRÉSENTATION DU SUJET : Un partage sans fin ou sans faim ?

1.1. Sujet

À l'automne, une population d'écureuils fait le plein de noisettes pour passer l'hiver. Chaque écureuil effectue sa récolte personnelle. Pour autant, afin que personne ne manque de rien, un système de partage particulier est mis en place. Quand deux écureuils se rencontrent, ils comparent leurs récoltes. L'écureuil qui a le plus de noisettes donne autant de noisettes que l'autre écureuil en a. Puis ils recommencent ce procédé jusqu'à ce que les deux écureuils aient le même nombre de noisettes.

1.2. Exemple

Tic a 18 noisettes et Tac a 15 noisettes. Alors Tic donne 15 noisettes à Tac. Il se retrouve donc avec 3 noisettes et Tac a maintenant 30 noisettes. $3 \neq 30$ donc ils recommencent le partage et ainsi de suite.

1.3. Problématique

Y-a-t-il des situations où le partage ne s'arrête pas ? Dans le cas où il se termine, combien d'étapes ont été nécessaires ?

1.4. Extensions

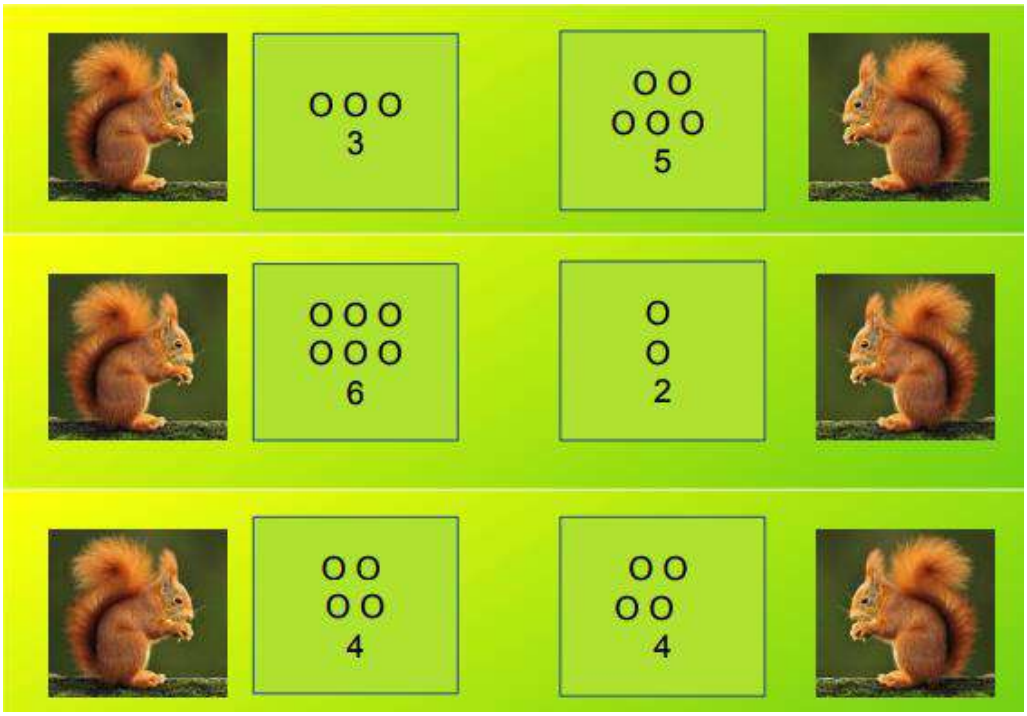
On étudiera d'abord le cas où il n'y a que deux écureuils. Puis on pourra étudier un système à 3 écureuils puis à n écureuils, les rencontres se faisant de façon aléatoire.

2. RECHERCHES

2.1. Test

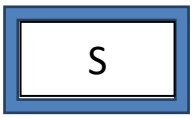
Dans un premier temps, nous sommes rentrés dans une phase de test : nous avons essayé plusieurs nombres, choisis complètement au hasard, pour comprendre comment le partage réussit ou comment il échoue. Nous avons essayé avec des nombres simples dans un premier temps. Evidemment, nous avons choisi des nombres entiers et dont la somme est paire puisque nous partons du principe qu'on ne peut obtenir des demi-noisettes.

2.2. Exemple de test :

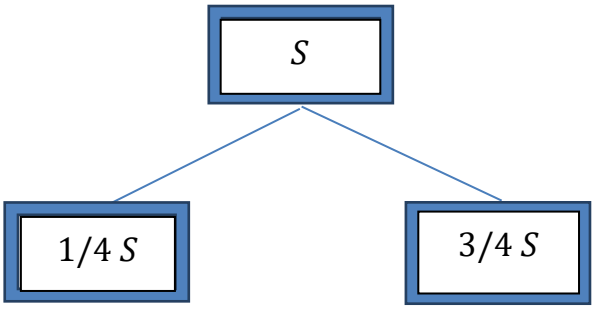


2.3. Arbres de valeurs

Nous avons également fait des arbres de valeur permettant de connaître toutes les répartitions possibles qui aboutiront à un partage équitable pour une somme de noisettes.
Pour les construire on place tout en haut la somme totale de noisettes, que l'on nomme S.

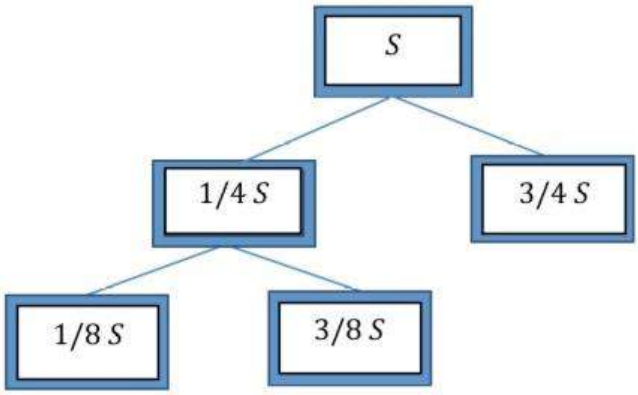


Puis on place ¼ et ¾ de la somme S dans les branches suivantes car c'est le seul moyen d'obtenir un cas d'égalité.



On effectue cela à nouveau pour les branches obtenues (1) jusqu'à ce que la dernière ligne soit entièrement composée de nombres impairs.

On note que l'on ne peut pas continuer car on obtiendrait des demi-noisettes ce qui n'est pas possible.



On obtiendra alors toutes les possibilités de répartition.

3. PROGRAMMES

Nous avons donc écrit un programme qui nous permet de simuler et d'automatiser les échanges entre écureuils.

Il s'articule de la manière suivante :

Il demande les valeurs qui lui sont nécessaires pour faire les calculs, à savoir le nombre de noisettes qui sont réparties entre tous les écureuils.

```
somme=int(input("Combien y a-t-il de noisettes au total?"))
ecureuil1=1
ecureuil2=somme-1
Léchec=[]
Lréussite=[]
```

Il effectue le partage en enregistrant le nombre de noisettes de chaque écureuil à chaque étape et reconnaît un partage interminable en vérifiant si les valeurs qui rentrent dans les listes ont déjà été rentrées et il reconnaît un partage terminé en voyant que les 2 écureuils ont le même nombre de noisettes.

```
test=1
nb_etapes=0
nbr_noisettes_inf=ecureuil1
nbr_noisettes_sup=ecureuil2
while nbr_noisettes°sup!=nbr_noisettes_inf and test=1:
    if nbr_noisettes_inf>nbr_noisettes_sup:
      nbr_noisettes_sup,nbr_noisettes_inf=nbr_noisettes_inf,nbr_noisettes_sup
    Lsup.append(nbr_noisettes_sup)
    Linf.append(nbr_noisettes_inf)
    #On fait le partage
    nbr_noisettes_sup=nbr_noisettes_sup-nbr_noisettes_inf
    nbr_noisettes_inf=nbr_noisettes_inf*2
    nb_etapes=nb_etapes+1

    for i in range(len(Lsup)):
        if Lsup[i]==nbr_noisettes_sup or Lsup[i]==nbr_noisettes_inf:

            test=0
```

Enfin, il termine par ranger les répartitions qui aboutissent dans une liste de réussite et celles qui n'aboutissent pas dans une liste échec, il passe à la répartition suivante en ajoutant 1 noisette à l'écureuil qui en a le moins et en enlevant 1 à celui qui en a le plus puis il réitère le partage. Il affiche lorsqu'il a testé toutes les répartitions réparties dans les listes échec et réussite.

```
    if nbr_noisettes_sup=nbr_noisettes_inf:
        Lréussite.append([ecureuil1,ecureuil2])
    elif test==0:
        Léchec.append([ecureuil1,ecureuil2])

while ecureuil1!=ecureuil2:
    partage(ecureuil1,ecureuil2) (2)
    ecureuil1=ecureuil+1
    ecureuil2=ecureuil2-1
print("Les répartitions qui aboutissent sont", Lréussite)
print("Les répartitions qui n'aboutissent pas sont", Léchec)
```

En exemple, si l'on prend 96 noisettes, le programme renvoie les réponses suivantes :

```
Les répartitions qui aboutissent sont [[3, 93], [6, 90], [9, 87], [12, 84], [15, 81], [18, 78
], [21, 75], [24, 72], [27, 69], [30, 66], [33, 63], [36, 60], [39, 57], [42, 54], [45, 51]]
Les répartitions qui n'aboutissent pas sont [[1, 95], [2, 94], [4, 92], [5, 91], [7, 89], [8,
 88], [10, 86], [11, 85], [13, 83], [14, 82], [16, 80], [17, 79], [19, 77], [20, 76], [22, 74
], [23, 73], [25, 71], [26, 70], [28, 68], [29, 67], [31, 65], [32, 64], [34, 62], [35, 61],
[37, 59], [38, 58], [40, 56], [41, 55], [43, 53], [44, 52], [46, 50], [47, 49]]
```

Il nous a donc permis de confirmer notre théorie sur les puissances de 2

4. PUISSANCE DE 2

Lorsqu'on construisait nos arbres, nous avons remarqué qu'avec une puissance de 2 comme somme de noisettes, toutes les répartitions de noisettes aboutissaient. Nous avons également noté grâce à notre programme, qu'en prenant 96 comme nombre de départ, les seules répartitions qui aboutissaient étaient composées de multiples de 3. Nous nous sommes ensuite rendu compte que le nombre 96 était égal à 32×3, 32 étant la plus grande puissance de 2 qui divise 96. On en a donc déduit qu'il y avait un lien entre les puissances de 2 et le nombre initial de noisettes.
On a alors obtenu la conjecture suivante : « On peut déterminer par le calcul toutes les répartitions qui aboutissent entre 2 écureuils avec une somme de noisettes donnée grâce aux puissances de 2 ».

Partie démonstration

Après avoir émis la conjecture qu'il existe un lien entre les puissances de 2 et la somme totale de noisettes, nous avons cherché à la démontrer.

Soit S un entier que l'on décompose en un produit de la plus grande puissance de 2 qui le divise : $S = 2^x \times r$.
L'ensemble des répartitions de départ qui aboutissent à une situation d'égalité est constitué de tous les multiples de r compris entre 1 et S et seulement ces multiples.

Nous sommes revenus à nos arbres de valeurs et nous avons même cherché à exprimer la somme S de toutes les cases sous une autre forme, qui est « $2^x \times r \times N$ » avec N impair et où l'on trouve r en divisant la somme totale par la plus grande puissance de 2 possible, comme expliqué précédemment, ce qui donne par exemple pour $S = 96$:

Ligne 0

$$96$$
$$2^5 \times 3 \times 1$$

A partir de là, on peut déduire le contenu de la case suivante de l'arbre, qui correspondra à la situation d'égalité [48;48] :

Ligne 0

$$96$$
$$2^5 \times 3 \times 1$$

Ligne 1

$$48 \mid 48$$
$$2^4 \times 3 \times 1 | 2^4 \times 3 \times 1$$

Pour trouver cela, on a simplement divisé 96 par 2, ce qui revient, dans le cas de notre expression, à passer de $S = 2^x \times r \times 1$ à $S = 2^{x-1} \times r \times 1$. En effet, diviser une puissance de 2 par 2 revient à soustraire 1 à son exposant pour trouver la puissance juste inférieure.
Si l'on continue maintenant avec la ligne suivante, qui est le cas $[24; 72]$ (3), on doit montrer que l'on ne perd pas cette forme. En effet, on sait que 48 a été divisé par 2, ce qui nous donne la première partie de la case :

Ligne 1

$$48 \mid 48$$
$$2^4 \times 3 \times 1 | 2^4 \times 3 \times 1$$

Ligne 2

$$24 \mid 72$$
$$2^3 \times 3 \times 1 | \ldots\ldots$$

Ici, l'expression « $2^4 \times 3 \times 1$» a seulement été divisée par 2 (ce qui donne $2^{4-1} \times 3 \times 1$, soit $2^3 \times 3 \times 1$.
On a alors cherché à retrouver la forme $r\ 2^x \times r \times N$ avec 72.
72 vient de $48 + 24$, or nous avons exprimé ces deux valeurs ailleurs dans l'arbre :

- $48 = 2^4 \times 3 \times 1$
- $24 = 2^3 \times 3 \times 1$
- On obtient ainsi l'expression $2^4 \times 3 \times 1 + 2^3 \times 3 \times 1$.

En utilisant la factorisation, nous avons réussi à retrouver la forme $2^x \times r \times N$:

$$2^4 \times 3 \times 1 + 2^3 \times 3 \times 1 = 2^3(2^1 \times 3 \times 1 + 3 \times 1)$$
$$= 2^3 \times 3(2 \times 1 + 1)$$

Ici, $2 \times 1 + 1$ est bien impair

Ligne 1

$48 \mid 48$
$2^4 \times 3 \times 1 | 2^4 \times 3 \times 1$

Ligne 2

$24 \mid 72$
$2^3 \times 3 \times 1 | 2^3 \times 3 \times 3$

Nous avons ensuite généralisé ce procédé afin de montrer qu'il s'adapte à tous les cas.

Si l'on prend n'importe quelle case de l'arbre qui ne se trouve pas sur la dernière ligne, on pourra exprimer les deux termes qu'elle contient sous la forme $2^n \times r \times N$ où l'on trouve n en soustrayant le numéro de la ligne à la puissance x dans l'expression de la somme de départ :
par exemple, à la ligne k, la puissance de 2 dans les expressions des termes vaudra 2^{x-k}.

En effet, si une case contient deux termes sous cette forme, alors les deux cases qu'elle va donner pourront également être exprimées de cette façon :

Ligne k

$2^{x-k} \times r \times N_1 \mid 2^{x-k} \times r \times N_2$

N_1 était la plus petite valeur (=celle qui a doublé) à l'étape précédente

N_2 était la plus petite valeur (=celle qui a doublé) à l'étape précédente

$2^{x-k-1} \times r \times N_1 \left| \begin{array}{l} 2^{x-k} \times r \times N_2 + 2^{x-k-1} \times r \times N_1 \\ = 2^{x-k-1}(2^1 \times r \times N_2 + r \times N_1) \\ = 2^{x-k-1} \times r(2^1 \times N_2 + N_1) \end{array} \right.$

$2^{x-k-1} \times r \times N_2 \left| \begin{array}{l} 2^{x-k} \times r \times N_1 + 2^{x-k-1} \times r \times N_2 \\ = 2^{x-k-1}(2^1 \times r \times N_1 + r \times N_2) \\ = 2^{x-k-1} \times r(2^1 \times N_1 + N_2) \end{array} \right.$

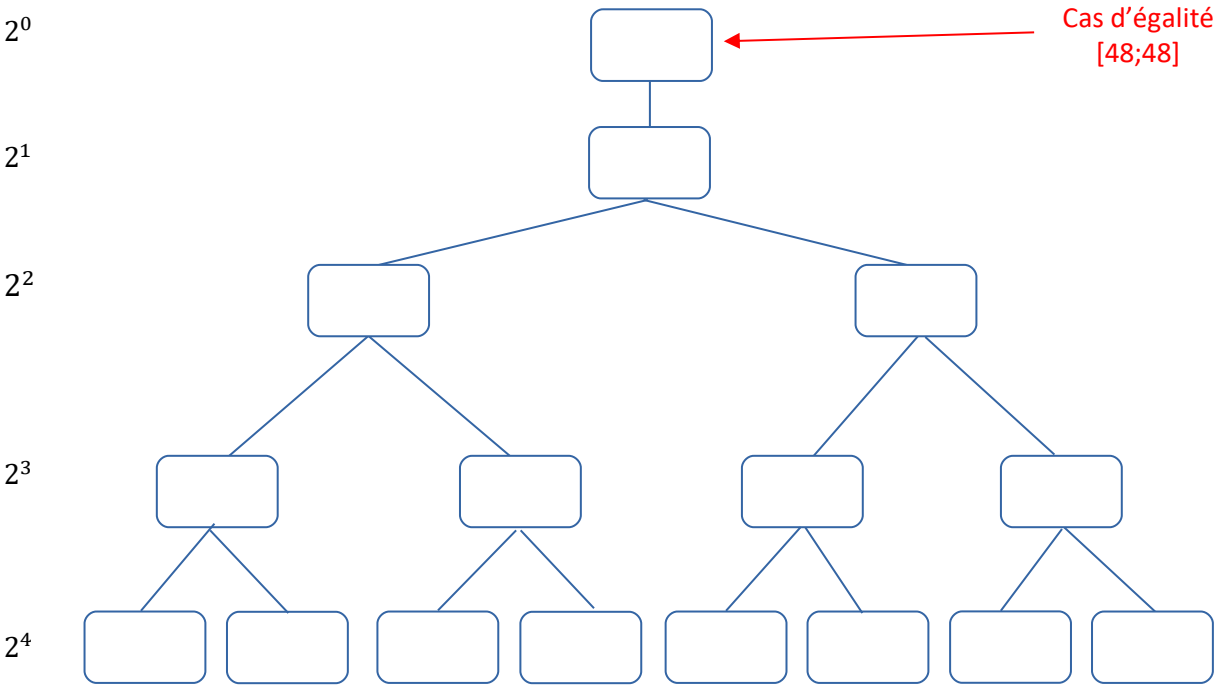
Ici, $2^1 \times N_1$ est pair, et N_2 est impair, la somme d'un nombre pair et d'un nombre impair est toujours impaire, donc $2^1 \times N_1 + N_2$ est impair.

On a montré ici qu'une case de l'arbre qui contient deux termes sous la forme $2^n \times r \times N$ donne deux cases dont on peut exprimer le contenu sous la forme $2^{n-1} \times r \times N$. Par récurrence, on peut ainsi déduire qu'à la dernière ligne de l'arbre, l'exposant de la puissance de 2 est nul, on a donc le produit de :

- 2^0, qui est impair,
- de r, lui aussi impair car obtenu en divisant S par la plus grande puissance de 2 possible,
- et de N, qui est également toujours impair.

Or, le produit de trois facteurs impairs est toujours impair, ce qui explique que les cases de la dernière ligne de l'arbre ne contiennent que des nombres impairs.

Il nous a fallu ensuite démontrer que l'on trouvait autant de nombres dans l'arbre que de multiples de r compris entre 1 et la somme, car cela signifierait que l'ensemble des répartitions qui aboutissent à une situation d'égalité est constitué de tous les multiples de r compris entre 1 et la somme.
Pour cela, nous avons cherché à connaître le nombre de termes que contient notre arbre (chaque case en contient 2). Celui-ci est constitué de x lignes successives, dont la première contient 1 terme. Ce nombre est ensuite multiplié par 2 à chaque ligne :



On peut ainsi dire que le nombre de cases dans l'arbre est égal à la somme des puissances de 2 de 2^0 à 2^n, où n est égal au nombre de lignes moins 1.

$$S = 2^0 + 2^1 + 2^2 + \cdots + 2^n$$
$$= 2^{n+1} - 1$$

Il faut également ajouter que la représentation de l'arbre que nous avons donné plus tôt en exemple est inexacte, car il faudrait y admettre la somme S qui est au début de l'arbre et est aussi un multiple de r. On ajoute ainsi 1 à notre expression $2^{n+1} - 1$.
Cela nous donne que pour un arbre ayant n lignes, le nombre de multiples de r qu'il contient est égal à 2^{n+1}.

Notre somme de départ s'écrit sous la forme $S = 2^x r$, on trouve donc 2^x multiples de r entre 1 et S. D'après le théorème que nous avons énoncé plus tôt, l'arbre de répartitions de S contient x lignes, mais la première ligne est la ligne 2^0, car elle ne contient qu'un terme. On fait donc la somme des puissances de 2 jusqu'à $x - 1$. Si l'on applique la formule, cela donne 2^{x-1+1}, donc 2^x.

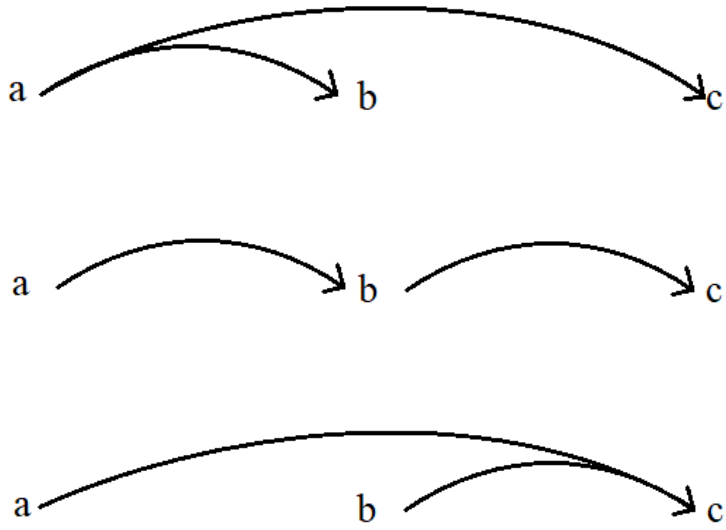
On trouve donc dans notre arbre un nombre de multiples de r égal au total de multiples de r compris entre 1 et la somme S. On peut donc dire que **l'ensemble des multiples de r compris entre 1 et la somme S, et seulement ces multiples constituent les répartitions qui aboutissent à une situation d'égalité (4).**

5. AVEC 3 ÉCUREUILS

Après avoir répondu au sujet, nous avons cherché à étendre notre problématique : Peut-on modifier le protocole de partage pour qu'il s'adapte à un autre nombre d'écureuils ?
Dans ce cas, peut-on trouver des similitudes avec les échanges à deux écureuils ?
Peut-on trouver un protocole universel pour n écureuils ?

Nous avons tout d'abord cherché un système d'échanges avec 3 écureuils. Celui à 2 écureuils peut être schématisé ainsi :

A partir de cela, nous avons établi 3 possibilités de partage :

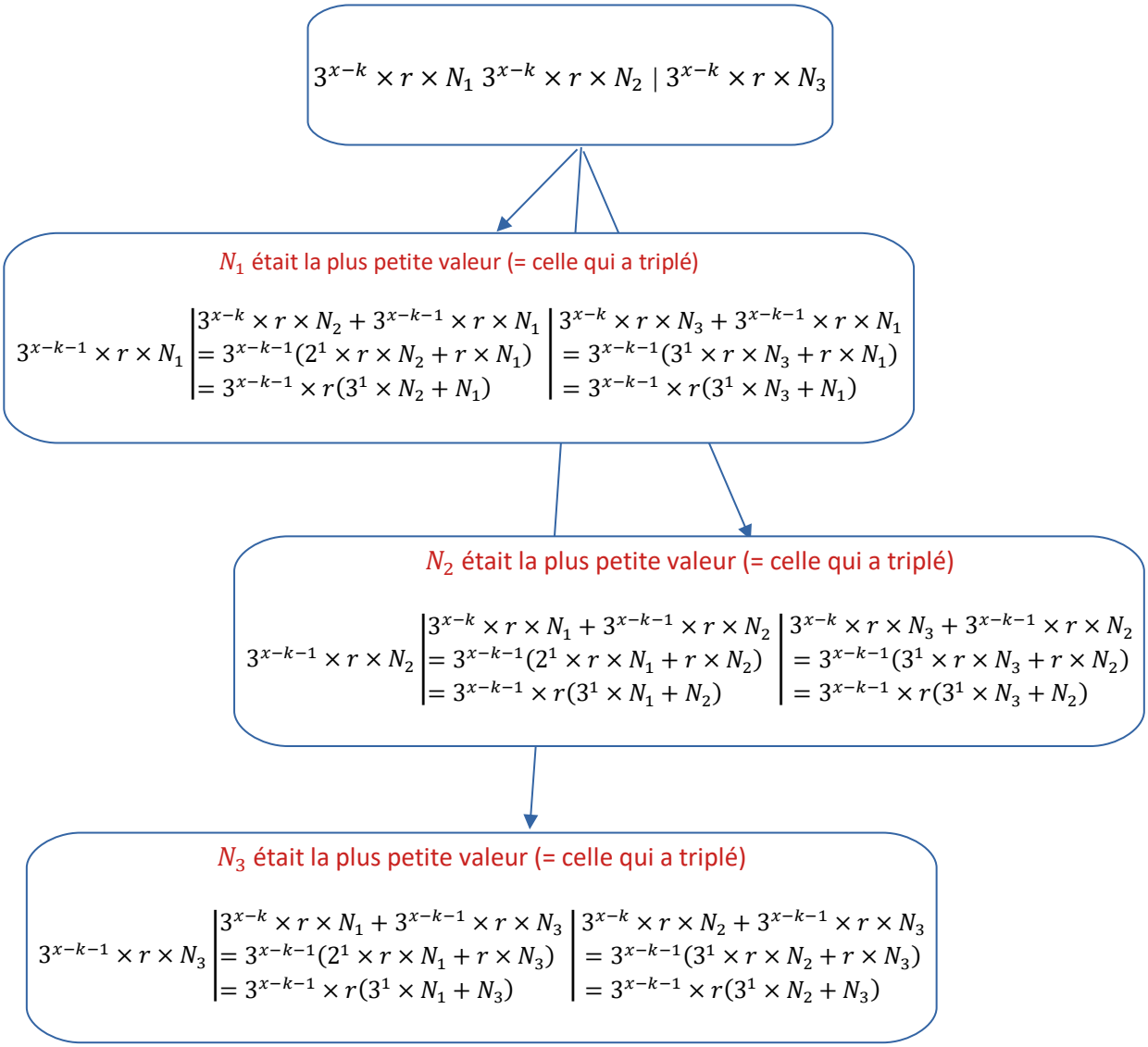


Dans le premier système, l'écureuil qui a le plus grand nombre de noisettes donne aux deux autres autant de noisettes que chacun en a. Or, si la somme des deux plus petits nombres est supérieure au plus grand, alors l'écureuil a, qui est celui qui a le plus de noisettes terminera l'échange avec un nombre négatif de noisettes.
La première proposition n'est donc pas exploitable.

Dans le deuxième système, les écureuils a et b effectuent un échange, puis les écureuils b et c en effectuent un second (on suppose que les nombres a, b et c sont rangés dans l'ordre croissant). Nous avons d'abord exploré ce système mais le problème ici est qu'il n'est pas directement lié au nombre d'écureuils : par exemple, lorsque l'on a 2 écureuils, on effectue des échanges entre 2 écureuils, or, dans notre cas, le nombre d'écureuils est de 3 mais l'on continue à effectuer des échanges entre deux écureuils séparément, ce qui pose un problème si l'on cherche à retrouver des schémas qui se répéteraient par rapport au cas des 2 écureuils.

Dans le cas du troisième protocole, cela fonctionne, car on a bien un lien entre le nombre d'écureuils et le fait que les échanges aient lieu entre les 3 écureuils simultanément. Nous avons donc poursuivi nos recherches à partir de ce système (5).

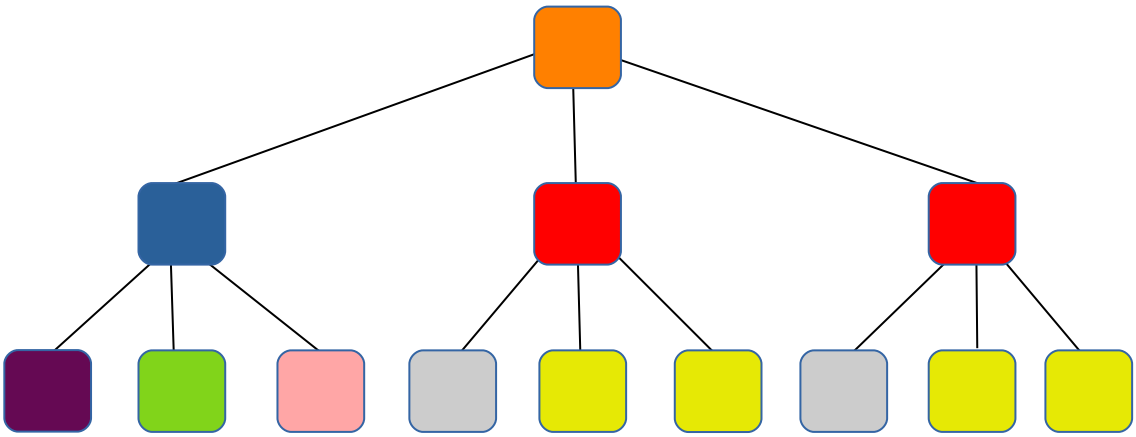
Nous avons essayé d'adapter la démonstration pour deux écureuils au cas avec 3 écureuils : (6)



Cela fonctionne donc à nouveau car tous les termes de l'arbre sont des multiples de r.

Nous avons constaté que cela fonctionne également avec 3 écureuils, cependant, la fin de la démonstration pour 2 écureuils ne peut pas être appliquée ici, car, en calculant le nombre de cases d'un arbre à 3 écureuils ou plus, on ne prend pas en compte les cas de branches identiques qui n'existaient pas à 2, car 2 écureuils ne pouvaient pas avoir le même nombre de noisettes.

Voici à quoi ressemble un arbre à 3 écureuils si l'on met en évidence à chaque ligne les cases qui contiennent des termes identiques (de même couleur) :

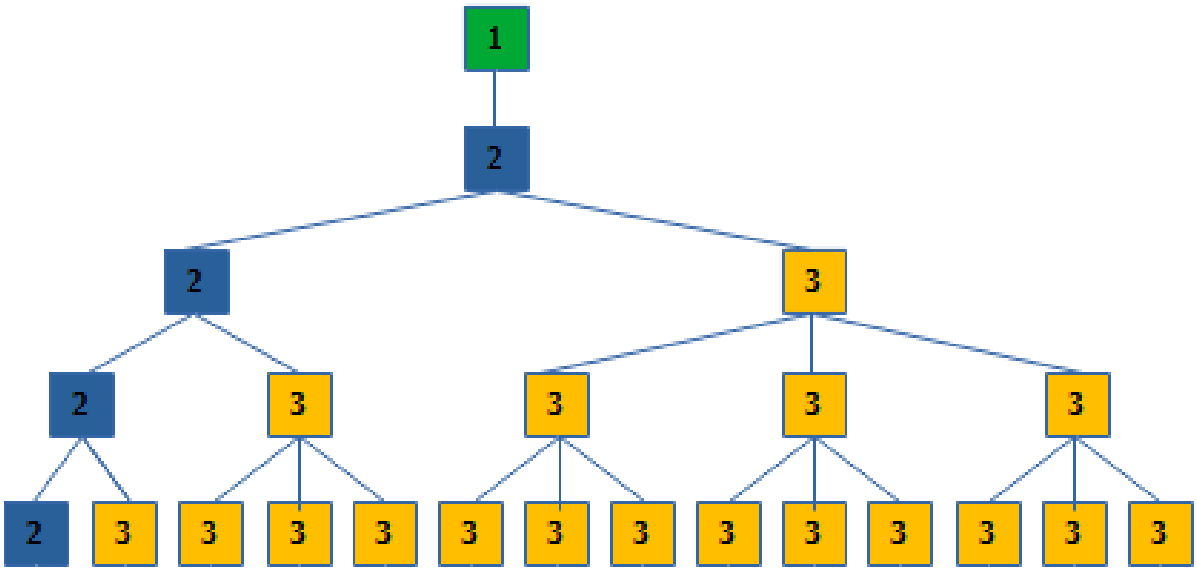


Nous nous sommes donc posé les questions suivantes : combien de couleurs différentes trouve-t-on dans cet arbre et est-il possible d'anticiper ce nombre sans le tracer ?
Nous avons changé la forme de nos arbres en n'y représentant seulement, par des nombres, le nombre de nouvelles répartitions (7) créées à chaque ligne en partant des règles suivantes :

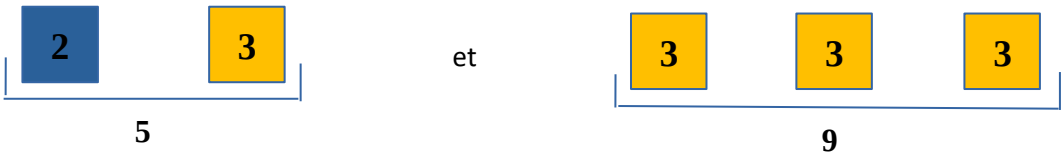
- une case qui contient 3 termes identiques sera notée 1 car elle ne contient qu'un seul exemplaire de nombre.
- une case qui contient 2 termes identiques et 1 autre différent des 2 premiers est notée 2 car elle contient 2 exemplaires de nombres.
- une case qui contient 3 termes différents est notée 3 car elle contient 3 exemplaires de nombres.

- une case 1 donne une case 2.
- une case 2 donne une case 2 et une case 3.
- une case 3 donne trois cases 3.

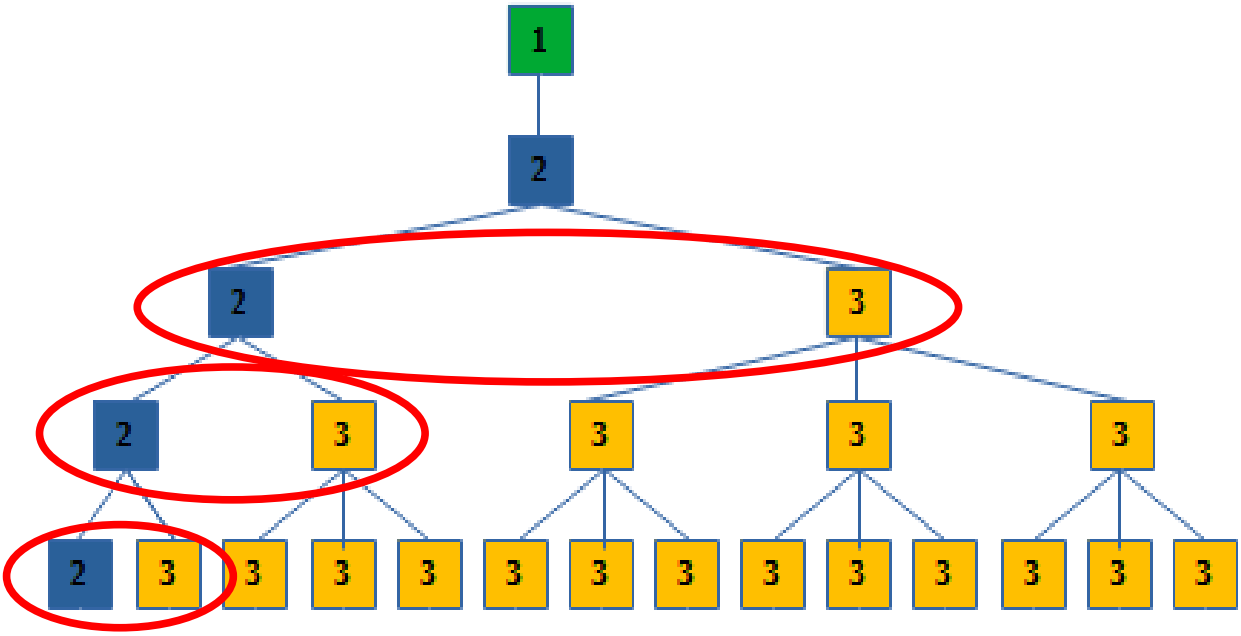
La première case de l'arbre est une case 1 car il s'agit de la situation d'égalité, chaque écureuil possède donc le même nombre de noisettes. On peut ensuite construire notre arbre :



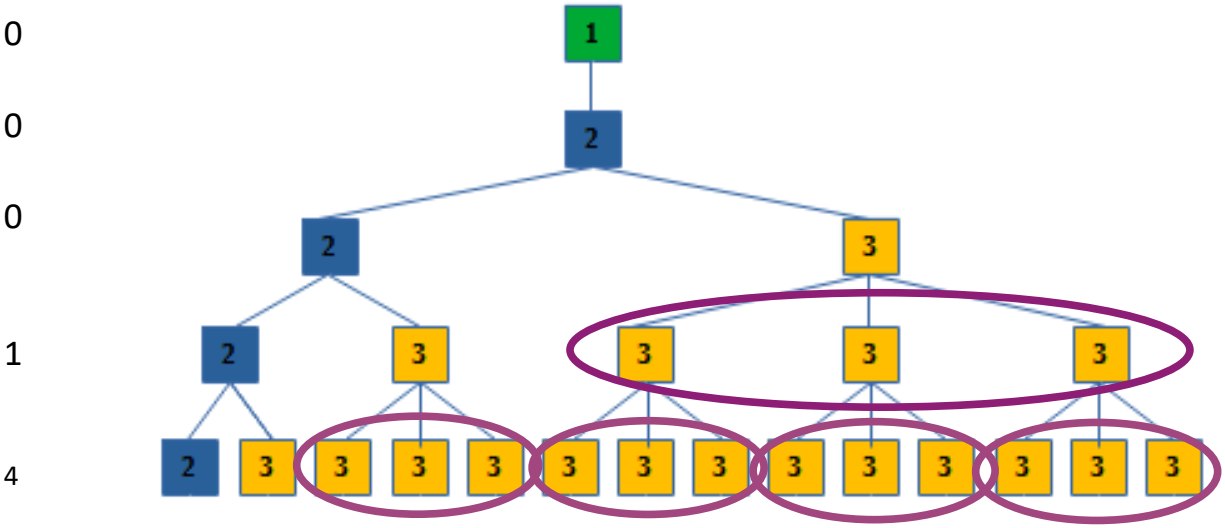
On cherche maintenant à déterminer la somme des nombres présents dans l'arbre, qui équivaudra au nombre de répartitions de noisettes différentes que l'on peut trouver dans l'arbre (8). On peut tout d'abord noter la présence de groupes de cases récurrents :



On trouve un « groupe 5 » à chaque ligne à partir de la troisième ligne.



On a également des « groupes 9 », dont le nombre varie selon les lignes (à partir de la quatrième ligne)



On peut ensuite préciser de nouvelles règles pour continuer :

- un groupe 5 donne un autre groupe 5 et un groupe 9
- un groupe 9 donne trois autres groupes 9

Ainsi, à chaque ligne

- un groupe 5 est créé ;
- le nombre de groupes 9 vaut 3 fois le nombre de groupes 9 à la ligne précédente (chacun en a créé 3), plus 1, créé par le groupe 5 de la ligne précédente.

On peut noter la somme totale (et déduire sa suite), de cette façon :

$$L1 = 1$$
$$L2 = 2$$
$$L3 = 1 \times 5 + 0 \times 9$$
$$L4 = 1 \times 5 + 1 \times 9$$
$$L5 = 1 \times 5 + (3 \times 1 + 1) \times 9 = 1 \times 5 + 4 \times 9$$
$$L6 = 1 \times 5 + (3 \times 4 + 1) \times 9 = 1 \times 5 + 13 \times 9$$
$$L7 = 1 \times 5 + (3 \times 13 + 1) \times 9 = 1 \times 5 + 40 \times 9$$
$$L8 = 1 \times 5 + (3 \times 40 + 1) \times 9 = 1 \times 5 + 121 \times 9$$

Pour calculer la somme totale de ces nombres, nous avons cherché à faire la somme des nombres que l'on multiplie par 9 à chaque ligne. Leur liste est la suivante :

$$1$$
$$4$$
$$13$$
$$40$$
$$121$$
$$\dots$$

Nous avons remarqué qu'ils correspondaient à la moitié des puissances de 3 auxquelles on a soustrait 1 (9) :

$$1 = \frac{(3^1 - 1)}{2}$$
$$4 = \frac{(3^2 - 1)}{2}$$
$$13 = \frac{(3^3 - 1)}{2}$$
$$40 = \frac{(3^4 - 1)}{2}$$
$$121 = \frac{(3^5 - 1)}{2}$$

On peut trouver la somme de ces nombres en simplifiant l'expression suivante :

$$\frac{(3^1 - 1) + (3^2 - 1) + \dots + (3^n - 1)}{2}$$
$$= \frac{3 + 3^2 + 3^3 + \cdots + 3^n}{2} - \frac{n}{2}$$
$$= 3 \times \frac{3^n - 1}{3 - 1} \times \frac{1}{2} - \frac{n}{2} = \frac{3(3^n - 1)}{4} - \frac{n}{2}$$
$$= \frac{3^{n+1} - 3 - 2n}{4}$$

$S = (3^{n+1} - 3 - 2n)/4.$

Nous avons donc conclu que, pour une somme totale de noisettes dont l'arbre possède n lignes, le nombre total de répartitions qui aboutissent est égal à :

$$3 + 5(n-2) + 9\left[\left(3^{n-1} - 3 - 2(n-2)\right)/4\right]$$ (10).

Notes d'édition

(1) L'interprétation de cet arbre n'est pas claire. Si on a noté dans les cases les possibilités pour le premier écureuil, on devrait avoir $\frac{1}{2}S$ au-dessus de $\frac{1}{4}S$ et $\frac{3}{4}S$, et surtout le $\frac{3}{8}S$ à la ligne suivante ne s'explique pas : alors l'autre écureuil a $\frac{5}{8}S$ et après échange le premier en a $\frac{3}{4}S$ et non $\frac{1}{4}S$.

(2) la définition de `partage(ecureuil1,ecureuil2)` est le bloc de programme précédent (`test=1`…) complété par les 4 premières lignes de celui-ci. La condition d'arrêt `while ecureuil1!=ecureuil2` devrait être `while ecureuil1<ecureuil2`, car si on a un nombre total de noisettes impair, on ne trouvera jamais `ecureuil1=ecureuil2`.

(3) On ne distingue pas les deux répartitions symétriques $24|72$ et $72|24$.

(4) Il se pourrait a priori qu'on ait le bon nombre de multiples de r mais qu'on ne les obtienne pas tous, si certains figurent dans plusieurs cases. Il faut encore montrer que cela ne se produit pas.
Or un multiple de la forme $2^n \times r \times N$ ne peut être présent que sur la ligne $x - n$. S'il y figure deux fois, les nombres des deux cases au-dessus, ligne $x - n - 1$, sont aussi identiques et on peut ainsi remonter jusqu'à ce que les branches se rejoignent. Mais c'est impossible puisque les deux cases obtenues en dessous d'une case ont des plus petites valeurs différentes (sauf au départ pour la case $2^{x-1} \times r|2^{x-1} - 1 \times r$ mais on n'a mis qu'une case en-dessous pour les deux répartitions symétriques).

(5) Dans ce troisième système, chacun des écureuils a et b donne à l'écureuil c autant de noisettes que celui-ci en avait avant échange. Mais le cas où deux des écureuils ont le même nombre de noisettes et le troisième en a un plus grand nombre n'est pas envisagé. Or ce cas peut très bien se produire même si ce n'est pas la situation au départ.
Au §1.4, il était plutôt suggéré d'étudier un système plus proche du deuxième système, avec des rencontres des écureuils deux par deux aléatoires. Cette piste aurait pu être explorée.

(6) On écrit le nombre total de noisettes $3^x \times r$, où r n'est pas divisible par 3. Dans les calculs suivants, on voit que si on a dans une case des termes de la forme $3^{x-k} \times r \times N$ avec N non multiple de 3, on trouve en dessous case des termes de la même forme avec l'exposant $x - k - 1$ au lieu de $x - k$.

(7) En dessous d'une case donnée, comme pour le cas de 2 écureuils, on ne met de nouveau qu'une case pour deux répartitions identiques à l'ordre des écureuils près.

(8) Chacun de ces nombres est égal au nombre de cases *en dessous de sa case* : pour le nombre de cases dans l'arbre, il faudrait donc calculer la somme des nombres présents hormis la dernière ligne, et lui ajouter 1 pour la première ligne. De plus, pour le nombre de répartitions il faudrait montrer que les cases correspondent à des répartitions toutes différentes – pour cela il suffit d'adapter la démonstration pour 2 écureuils (note 4) compte-tenu de la remarque ci-dessus sur les puissances de 3 (note 6).

(9) En remplaçant de ligne en ligne dans les parenthèses on se ramène à une somme de puissances de 3 consécutives, comme celle calculée un peu plus loin : $13 = 3 \times 4 + 1 = 3 \times (3 \times 1 + 1) + 1 = 3^2 + 3 + 1$, $3 \times 13 + 1 = 3 \times (3^2 + 3 + 1) + 1 = 3^3 + 3^2 + 3 + 1$, …
Le nombre de cases 3 dans l'arbre s'obtient avec la même formule et il aurait été plus simple de l'utiliser directement.

(10) Cette formule n'est pas exacte. À la ligne k, on a $(3^{k-3} - 1)/2$ groupes 9 et donc dans la formule de la somme il faut remplacer n par $n - 3$ et non par $n - 2$. Mais il y a surtout l'erreur signalée en note 8. En sommant jusqu'à la ligne $n - 1$ et en ajoutant 1 on trouve alors la bonne valeur $(3^{n-1} + 2n + 1)/4$.