Étude

Pascal au carrefour des probabilités, de l'algorithmique, de la récurrence et de la combinatoire

Dominique Baroux et Martine Bühler

Le groupe M. : A.T.H. s'est intéressé, dès le début de ses activités, à la correspondance de Pascal et Fermat sur le « problème des partis » et à l'utilisation qu'on peut en faire en classe. La relecture de ces textes, à la lumière de nouvelles préoccupations pédagogiques, nous a amenées à étudier de manière plus approfondie les idées de Pascal. Ce dernier donne en effet une solution « algorithmique » au problème, qui permet des activités riches avec nos élèves, dans les domaines de l'algorithmique et de la programmation, apparus au cours des dernières années dans les programmes de mathématiques. L'article présente la méthode de Pascal pour résoudre le problème, présentée dans une lettre à Fermat et dans le *Traité du Triangle Arithmétique*. La rubrique « Dans nos classes » présente l'utilisation qui peut être faite en classe de ces textes, aussi bien du point de vue mathématique, qu'algorithmique ou informatique. Le *Traité* donne également la construction de ce fameux triangle arithmétique, ses propriétés, le principe de la démonstration par récurrence et son usage pour les combinaisons.

Introduction

En 1654, Pascal et Fermat échangent une correspondance au sujet d'un problème de partage de mise entre les joueurs, quand une partie est interrompue, connu sous le nom de « problème des partis ». Cet échange épistolaire est souvent considéré comme l'acte de naissance des probabilités, une affirmation qu'on pourrait nuancer en tenant compte des études sur les discussions antérieures autour de ce problème (Coumet, 1965 et Meusnier, 2004) et d'autres questions liées aux jeux de hasard, par exemple le problème du Duc de Toscane (Henry, 2011).

Dans une première partie, nous nous intéresserons particulièrement à la solution donnée par Pascal dans une de ses lettres. Cette solution est clairement algorithmique, et prend pour l'enseignement des mathématiques en terminale une nouvelle résonance avec la présence de plus en plus importante de l'algorithmique et de la programmation. Pascal revient sur ce problème dans la troisième partie de son *Traité du Triangle Arithmétique*, publié de façon posthume en 1665, en précisant les principes qui doivent, selon lui, nous guider vers la solution, et donne à nouveau sa solution algorithmique, qui n'utilise pas le triangle arithmétique. Cependant, dans le *Traité*, Pascal donne une solution au problème à l'aide de ce fameux triangle.

Nous examinerons donc ensuite ce *Traité du Triangle Arithmétique*, en expliquant la génération du triangle, et celles de ses propriétés utiles pour la compréhension de la suite. Dans les usages du triangle arithmétique figure une étude des combinaisons, tout à fait utilisable en classe. Pascal y présente en effet clairement la notion de combinaison, et donne la relation de récurrence (dite de Pascal dans les programmes) avec une démonstration sur un exemple générique qui peut permettre aux élèves de comprendre, et peut-être même d'aborder seul·es, la démonstration générale.

Nous pourrons alors étudier la troisième partie du *Traité du Triangle Arithmétique*, consacrée à une autre solution du problème des partis, utilisant ce triangle. Pascal utilise les mêmes principes donnés en première partie de l'article, et montre comment la connaissance du triangle donne immédiatement les règles du partage à effectuer. En développant ces questions, aussi bien celles des propriétés du triangle que celles sur les combinaisons ou la règle des partis, Pascal est amené à donner les principes de la démonstration par récurrence.

Les extraits de textes sont tirés de l'édition des *Œuvres Complètes* de Pascal citée en bibliographie.

La méthode algorithmique de Pascal

Le problème des partis et la lettre de Pascal de juillet 1654

Rappelons le problème, sur lequel on peut faire travailler les élèves, en groupes en classe ou à la maison, comme problème ouvert et/ou narration de recherche [1] . Mieux vaut d'ailleurs avoir déjà réfléchi au problème avant de lire la suite de l'article.

> Deux joueurs jouent à un jeu de pur hasard ; le premier qui gagne un nombre déterminé de parties (par exemple 3) remporte la mise. Or, ils doivent quitter le jeu avant que l'un d'entre eux ait gagné (par exemple, le joueur A a 2 points et le joueur B a 1 point). Comment doivent-ils partager la mise [a] ?
>
> a. Il s'agit d'un problème de partage, d'où le nom de problème des partis, écrit aussi parfois party, le mot « parti » ayant le sens de « partage ».

Dans une lettre à Fermat du 29 Juillet 1654, Pascal explique sa méthode sur l'exemple d'un jeu en trois parties, lorsque le premier joueur a un point et le second zéro, en commençant par des cas plus simples qui lui permettront de résoudre son exemple en procédant de proche en proche [2].

> « Voici à peu près comment je fais pour savoir la valeur de chacune des parties, quand deux joueurs jouent, par exemple, en trois parties, et chacun a mis 32 pistoles au jeu :
>
> Posons que le premier en ait deux et l'autre une ; ils jouent maintenant une partie, dont le sort est tel que, si le premier la gagne, il gagne tout l'argent qui est au jeu, savoir 64 pistoles ; si l'autre la gagne, ils sont deux parties à deux parties, et par conséquent, s'ils veulent se séparer, il faut qu'ils retirent chacun leur mise, savoir chacun 32 pistoles.
>
> Considérez donc, Monsieur, que, si le premier gagne, il lui appartient 64 ; s'il perd, il lui appartient 32. Donc, s'ils veulent se séparer sans la jouer, le premier doit dire : « Je suis sûr d'avoir 32 pistoles, car la perte même me les donne ; mais pour les 32 autres, peut-être je les aurai, peut-être vous les aurez, le hasard est égal. Partageons donc ces 32 pistoles par la moitié et me donnez, outre cela, les 32 qui me sont sûres ». Il aura donc 48 pistoles et l'autre 16.
>
> Posons maintenant que le premier ait deux parties et l'autre point, et ils commencent à jouer une partie. Le sort de cette partie est tel que, si le premier la gagne, il tire tout l'argent, 64 pistoles ; si l'autre la gagne, les voilà revenus au cas précédent, auquel le premier aura deux parties et l'autre une.
>
> Or, nous avons déjà montré qu'en ce cas il appartient, à celui qui a les deux parties, 48 pistoles : donc, s'ils veulent ne point jouer cette partie, il doit

1. Vous trouverez dans la rubrique « Dans nos classes » un problème donné en terminale scientifique. Vous trouverez dans le dossier sur l'histoire des probabilités en classe du groupe MATH, cité en bibliographie, un compte-rendu d'expérience en classe sur ce problème.

2. Fermat, *Œuvres*, publiées par P. Tannery et C. Henry, tome II, Gauthier-Villars, Paris, 1894, p. 290-291.

dire ainsi : « Si je la gagne, je gagnerai tout, qui est 64 ; si je la perds, il m'appartiendra légitimement 48 : donc donnez-moi les 48 qui me sont certaines, au cas même que je perde, et partageons les 16 autres par moitié, puisqu'il y a autant de hasard que vous les gagniez comme moi ». Ainsi, il aura 48 et 8, qui sont 56 pistoles.

Posons enfin que le premier n'ait qu'*une* partie et l'autre *point*. Vous voyez, Monsieur, que, s'ils commencent une partie nouvelle, le sort en est tel que, si le premier la gagne, il aura *deux* parties à *point*, et partant, par le cas précédent, il lui appartient 56 ; s'il la perd, ils sont partie à partie : donc il lui appartient 32 pistoles. Donc il doit dire : « Si vous voulez ne la pas jouer, donnez-moi 32 pistoles qui me sont sûres, et partageons le reste de 56 par la moitié. De 56 ôtez 32, reste 24 ; partagez donc 24 par la moitié, prenez-en 12 et moi 12, qui, avec 32, font 44. » »

Les principes utilisés par Pascal

Pascal utilise deux principes qu'il précisera dans son *Traité du Triangle Arithmétique*[3].

« USAGE DU TRIANGLE ARITHMÉTIQUE

POUR DÉTERMINER LES PARTIS QU'ON DOIT FAIRE ENTRE DEUX JOUEURS QUI JOUENT EN PLUSIEURS PARTIES

[...]Le premier principe qui fait connaître de quelle sorte on doit faire les partis est celui-ci. Si un des joueurs se trouve en telle condition que, quoiqu'il arrive, une certaine somme doive lui appartenir en cas de perte et de gain, sans que le hasard la lui puisse ôter, il n'en doit faire aucun parti, mais la prendre entière comme assurée parce que, le parti devant être proportionné au hasard, puisqu'il n'y a nul hasard de perdre, il doit tout retirer sans parti. »

C'est bien ce principe qu'a utilisé Pascal dans l'extrait ci-dessus lorsqu'il affirme : « Je suis sûr d'avoir 32 pistoles, car la perte même me les donne [...] donnez[-moi][...] les 32 qui me sont sûres ».

« Le second est celui-ci. Si deux joueurs se trouvent en cette situation que, si l'un gagne, il lui appartiendra une certaine somme, et, s'il perd, elle appartiendra à l'autre ; si le jeu est de pur hasard et qu'il y ait autant de hasards pour l'un que pour l'autre, et par conséquent non plus de raison de gagner pour l'un que pour l'autre, s'ils veulent se séparer sans jouer, et prendre ce qui leur appartient légitimement, le parti est qu'ils séparent la somme qui est au hasard par la moitié, et que chacun prenne la sienne. »

Ce second principe a également été utilisé : « mais pour les 32 autres, peut-être je les aurai, peut-être vous les aurez, le hasard est égal. Partageons donc ces 32 pistoles par la moitié ».

Ces deux principes répondent à l'exigence que les joueurs « ont reçu [...]le droit d'attendre ce que le hasard leur [...] peut donner. [...] le règlement de ce qui doit leur appartenir doit être tellement proportionné à ce qu'ils avaient droit d'espérer de la fortune que chacun d'eux trouve entièrement égal de prendre ce qu'on lui assigne ou de continuer l'aventure du jeu. »

Les deux principes énoncés permettent à Pascal de donner deux corollaires, le second équivalent au premier, qui permettent de faire le « parti ».

3. Pascal, *Œuvres complètes*, tome II, texte présenté, établi et annoté par Jean Mesnard, Desclée de Brouwer, 1970 (p. 1308-1311).

« *Corollaire premier*

Si deux joueurs jouent à un jeu de pur hasard, à condition que, si le premier gagne, il lui reviendra une certaine somme, et, s'il perd, il lui en reviendra une moindre ; et s'ils veulent se séparer sans jouer, et prendre chacun ce qui leur appartient, le parti est que le premier prenne ce qui lui revient en cas de perte, et de plus la moitié de l'excès dont ce qui lui reviendrait en cas de gain surpasse ce qui lui revient en cas de perte.

[...]*si, en cas de perte il lui appartient A, et en cas de gain $A + B$, le parti est qu'il prenne $A + \frac{1}{2}B$.*

Corollaire second

Si deux joueurs sont en la même condition qu'on vient de dire, je dis que le parti se peut faire de cette façon qui revient au même ; que l'on assemble les deux sommes de gain et de perte, et que le premier prenne la moitié de cette somme[4] ; [...] »

Pascal raisonne ici, comme dans sa lettre à Fermat, sur la somme que chaque joueur peut légitimement emporter si le jeu s'arrête, ce que nous nommons l'espérance de gain. Pascal remarque ensuite :

« qu'il ne faut proprement avoir égard qu'au nombre de parties qu'il reste à gagner à l'un et à l'autre, et non pas au nombre de celles qu'ils ont gagnées, puisque, comme nous avons dit, deux joueurs se trouvent en même état quand, jouant en deux parties, l'un en a une à point, que deux qui, jouant en douze parties, l'un en a onze à dix. Il faut donc proposer la question en cette sorte :

Étant proposés deux joueurs, à chacun desquels il manque un certain nombre de parties pour achever, faire le parti. »

Il n'est pas du tout intuitif qu'on ne doive tenir aucun compte des parties déjà gagnées, et ne considérer que les parties qui manquent à chaque joueur. Cette difficulté se retrouve aussi bien dans les solutions « historiques » envisagées avant le XVII^e siècle (Coumet, 1965), que dans les solutions proposées par les élèves lorsqu'on pose le problème de façon ouverte[5].

Un algorithme récursif

Pascal donne ensuite des exemples, très semblables à ceux de sa lettre, qui montrent bien qu'un cas se résout de proche en proche. Dans un jeu où il faut 3 points pour remporter la mise, pour savoir ce qui revient au premier joueur en cas d'arrêt du jeu quand le premier a 2 points et le second 0 point, il faut d'abord résoudre le cas où le premier a 3 et le second 0 (évident ici) et le cas où le premier a 2 points et le second a 1 point. Ce dernier cas nécessite de connaître le « parti » lorsque le premier a 3 points et le second 1 point (évident ici) et le cas où le premier et le second ont tous deux 2 points (partage par moitié selon le second principe).

Plus loin, il parle de la « portion [...] sur la somme qu'ils jouent, exprimée par une fraction », qui revient à un joueur. Cette fraction est, pour nous, la probabilité que le joueur gagne la partie, puisque « le parti [doit] être proportionné au hasard ». Il précise dans un lemme que, « si deux joueurs jouent à un jeu de pur hasard, à condition que, si

4. Pascal montre que cela est équivalent à la conclusion du corollaire premier car la somme considérée est $A + A + B$, dont la moitié est bien $A + \frac{1}{2}B$.

5. Vous trouverez un compte-rendu d'expérience dans le *Bulletin Vert* de l'APMEP, N° 514, p. 264-274. https://www.apmep.fr/IMG/pdf/AAA15029.pdf

le premier gagne, il lui appartiendra une portion quelconque sur la somme qu'ils jouent, et que, s'il perd, il lui appartiendra une moindre portion sur la même somme, exprimée par une autre fraction », on doit calculer la demi-somme de ces deux fractions et « cette fraction exprime la portion qui appartient au premier sur la somme qui est au jeu », en conformité avec ce qu'affirme le second corollaire. Nous pouvons traduire cela en termes de probabilités. Imaginons qu'il manque a parties au premier joueur et b parties au second, a et b non tous deux nuls ; appelons $P(a,b)$ la probabilité que le premier joueur gagne le jeu ; la portion de la mise que doit emporter le premier joueur est bien $P(a,b)$.

S'ils jouent maintenant une partie, si le premier la gagne, il manque alors au premier $a-1$ points et au second toujours b points. Donc, en cas de gain, il emporte la « portion » $P(a-1,b)$ de la mise ; si au contraire le premier perd, il lui manque alors toujours a points et au second $b-1$ points ; donc le premier joueur emporte la « portion » $P(a,b-1)$ de la mise. Le corollaire second affirme alors que, « s'ils veulent se séparer sans jouer », il lui revient la « portion » $\frac{1}{2}P(a-1,b)+\frac{1}{2}P(a,b-1)$ de la mise. On a donc :

$$P(a,b) = \frac{1}{2}P(a-1,b) + \frac{1}{2}P(a,b-1)$$

On voit bien ici une fonction récursive[6] $P(a,b)$ définie par :

---

a et b deux entiers naturels non tous deux nuls

Si $a=0,\ P(a,b)=1$

Sinon : **Si** $b=0,\ P(a,b)=0$

Sinon : **Si** $a=b,\ P(a,b)=\frac{1}{2}$

Sinon, $P(a,b)=\frac{1}{2}P(a-1,b)+\frac{1}{2}P(a,b-1)$

On peut implémenter cet algorithme[7] dans le langage Python :

```python

def probaparti(a,b):
    if a<0 or b<0 or (a==0 and b==0):
        print('Mauvaises entrées')
        #raise ValueError()
        return -1
    elif a==0:
        return 1
    elif b==0:
        return 0
    elif a==b:
        return 0.5
    else:
        return(0.5*probaparti(a-1,b)+0.5*probaparti(a,b-1))

a = input('Tapez le nombre a:')
b = input('Tapez le nombre b:')

a = int(a)
b = int(b)

proba = probaparti(a,b)
print(proba)

```

6. Vous trouverez dans la rubrique « Dans nos classes » un exercice d'introduction à la récursivité pour une classe de terminale.

7. Vous trouverez sur la page du groupe M. :A.T.H. les algorithmes de cet article en langage Python https://irem.u-paris.fr/utilisation-de-lhistoire-des-mathematiques-en-probabilites

L'algorithme permet d'obtenir la probabilité que le premier joueur, A, gagne le jeu, losqu'il manque a parties au premier joueur et b parties au second joueur, B. On peut faire quelques essais :

```
In [11]: probaparti(2,3)
Out[11]: 0.6875

In [12]: probaparti(9,10)
Out[12]: 0.5927352905273438
```

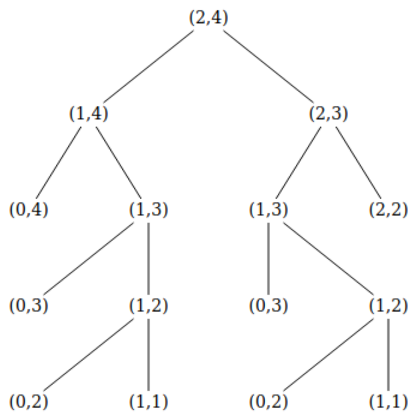
Voici ce qu'on obtient pour $(a, b) = (500, 501)$

```
RecursionError: maximum recursion depth exceeded in comparison
```

Que signifie ce message d'erreur ? Quels problèmes pose l'implémentation ? Ces problèmes sont-ils liés au langage, ou à la complexité de l'algorithme lui-même ?

Problèmes de l'implémentation de l'algorithme de Pascal

Examinons l'arbre de calcul suivant :



- **Pour** calculer P(2,4), il faut calculer P(1,4) et P(2,3).
  - **Pour** calculer P(1,4), il faut calculer P(0,4) et P(1,3) ; P(0,4)=1.
    - **Pour** calculer P(1,3), il faut calculer P(0,3) et P(1,2) ; P(0,3) = 1.
      - **Pour** calculer P(1,2), il faut calculer P(0,2), qui vaut 1, et P(1,1), qui vaut $\frac{1}{2}$.
  - **Pour** calculer P(2,3), il faut d'abord calculer P(1,3) et P(2,2) ; P(2,2) = 0.5.
    - **Pour** calculer P(1,3), il faut calculer P(0,3) et P(1,2) ; P(0,3) = 1.
      - **Pour** calculer P(1,2), il faut calculer P(0,2), qui vaut 1, et P(1,1), qui vaut $\frac{1}{2}$.

La « hauteur » de l'arbre est 4, car il y a quatre niveaux en plus de la « racine » (2,4). Pour chaque niveau, la somme $s = a + b$ est la même pour tous les couples du niveau : 6 à la racine, 5 au premier niveau, 4 au deuxième niveau, 3 au troisième niveau, 2 au quatrième niveau. En effet, le calcul de $P(a, b)$ nécessite celui de $P(a-1, b)$ et de $P(a, b-1)$ et $(a-1)+b = a+(b-1) = s-1$. La somme s est donc un « invariant » du niveau. On est sûr que le dernier niveau correspond à $s = 2$. En effet, inutile de continuer l'arbre après ce niveau car les seules décompositions de 2 en somme de deux entiers naturels sont : $2 = 0 + 2 = 1 + 1 = 2 + 0$ et on connaît $P(0, 2) = 1$, $P(1, 1) = \frac{1}{2}$, $P(2, 0) = 0$. Et on arrive toujours à ce niveau, car, partant de la racine (a, b) avec $s = a + b$, a et b non nuls, non égaux, et $a < b$, on doit passer par $(a - 1, b)$, $(a - 2, b)$ ..., $(1, b)$ (pour lesquels on a toujours $a - k < b$ et $a - k$ non nul, donc pas d'arrêt possible), puis par $(1, b)$, $(1, b - 1)$, ..., $(1, 2)$, $(1, 1)$ (arrêt à ce moment). Il y a donc toujours $s - 2$ niveaux après la racine.

Cet arbre va nous permettre de comprendre ce qui se passe lors de l'exécution du programme, ce que signifie le message d'erreur précédent, et les problèmes de complexité que nous pourrons rencontrer.

Lorsque l'exécution d'une fonction F rencontre un appel à une fonction G, l'exécution des instructions de la fonction F s'interrompt le temps d'exécuter les instructions de la fonction G (qui peut elle-même impliquer d'autres appels de fonction). Lorsque l'exécution de G s'achève, il faut retourner à son point d'appel dans F, et reprendre l'exécution de F à partir de ce point. Au moment du passage à la fonction G, les informations nécessaires au retour dans la fonction F sont stockées dans une « pile d'exécution ». Ces informations seront utilisées (et retirées de la pile) à la reprise de l'exécution de la fonction F. Dans le cas d'une fonction récursive, F et G sont la même fonction, mais le mécanisme est le même. Ainsi, pour calculer P(2,4), le programme fait un appel récursif à P car le calcul nécessite les valeurs P(1,4) et P(2,3). Au moment de l'appel récursif de la fonction P, on interrompt l'exécution des instructions de la « première » fonction P ; le premier niveau de la pile d'exécution contient toutes les informations nécessaires au retour au point d'appel. Pour calculer P(1,4), on fait de nouveau appel à P, car on a besoin de P(0,4) et P(1,3). On stocke les informations nécessaires dans le deuxième niveau de la pile. Puis on exécute le code de P pour connaître P(0,4). On a atteint la condition d'arrêt, donc on passe au calcul de P(1,3). Appel à P pour connaître P(0,3) et P(1,2) et stockage des informations dans le troisième niveau de la pile. Pour P(0,3), on a atteint la condition d'arrêt ; nouvel appel de P pour le calcul de P(1,2) et stockage d'informations dans le quatrième niveau de la pile. Cette fois-ci, la condition d'arrêt est remplie pour les deux nombres P(0,2) et P(1,1). On peut donc calculer P(1,2) et « dépiler » le quatrième niveau, puis calculer P(1,3) et dépiler le troisième niveau, puis calculer P(1,4) et dépiler le deuxième niveau. On est revenu à une pile avec un seul niveau.

On passe ensuite au calcul de P(2,3), qui va nécessiter de stocker de nouvelles informations, en reconstituant une pile à partir du premier niveau. On voit donc que la hauteur de la pile varie au cours de l'exécution, et que la hauteur maximale atteinte est la hauteur de l'arbre.

C'est de la hauteur de l'arbre que vient notre message d'erreur pour le couple (500,501) ; en effet, Python ne favorise pas l'écriture récursive [8] ; la taille (ou hauteur) de la pile est limitée par défaut à 1000, un choix dû à Guido van Rossum, le créateur du langage Python [9].

D'autres essais sont intéressants. Pour le couple (17,18), la réponse arrive en un peu plus d'une minute, et, pour le couple (15,25), toujours pas de réponse au bout de 5 minutes, sans pourtant avoir le message d'erreur ci-dessus, qui arrive très vite pour (500,501). La hauteur de l'arbre n'est pas en cause (38 niveaux seulement), mais, par contre, sa largeur (nombre de couples à calculer pour un niveau donné) augmente rapidement. Pour le couple

8. Voir : Ortiz P., *La récursivité*, article en ligne, mai 2021, http://pascal.ortiz.free.fr/contents/python/recursivite/recursivite.html

9. Voir : Becirspahic J.P., *Récursivité*, article en ligne, https://info-llg.fr/commun-mp/pdf/02.recursivite.pdf

(17,18) par exemple, la largeur est 2 au niveau 1, 2 au niveau 2 (car l'arbre s'arrête pour le couple (17,17)), 4 au niveau 3, 6 au niveau 4 et 11 au niveau 5. Au pire, on passe d'un niveau à l'autre en doublant le nombre de couples, mais cela n'arrive pas car il y a des arrêts pour $a = 0$, $b = 0$ ou $a = b$. Par exemple, à partir d'un couple (a, b) avec $a < b$, il y a au moins un arrêt après $b - a$ étapes lorsqu'on arrive à (a, a) après $(a, b - 1)$, $(a, b - 2)$, ... et également après a étapes lorsqu'on arrive à $(0, b)$.

On voit également sur l'exemple de P(2,4) que les valeurs pour certains couples sont calculées plusieurs fois, par exemple P(1,2) et P(1,3). Le nombre de calculs à effectuer devient vite très grand. Et, comme nous l'avons remarqué, on doit calculer plusieurs fois le même couple : par exemple, pour (17,18), (13,17) apparaît 4 fois au niveau 5, et cela entraîne aussi bien sûr des calculs « inutiles » aux niveaux suivants. Il faut donc optimiser l'algorithme en gardant en mémoire les couples déjà calculés. Avec le langage Python, on peut créer un « dictionnaire » indexé par les couples d'entiers avec l'instruction dict().

```
val_dict = dict()

def probaparti(a,b):

    if a<0 or b<0 or (a==0 and b==0):
        print('Mauvaises entrées')

        return -1

    elif a==0:
        return 1
    elif b==0:
        return 0
    elif a==b:
        return 0.5
    else:
        if (a,b) in val_dict.keys():
            return(val_dict[(a,b)])
        else:
            r = 0.5*probaparti(a-1,b)+0.5*probaparti(a,b-1)
            val_dict [(a,b)]=r

            return r

a = input('Tapez le nombre a:')
b = input('Tapez le nombre b:')

a = int(a)
b = int(b)

proba = probaparti(a,b)
print(proba)
```

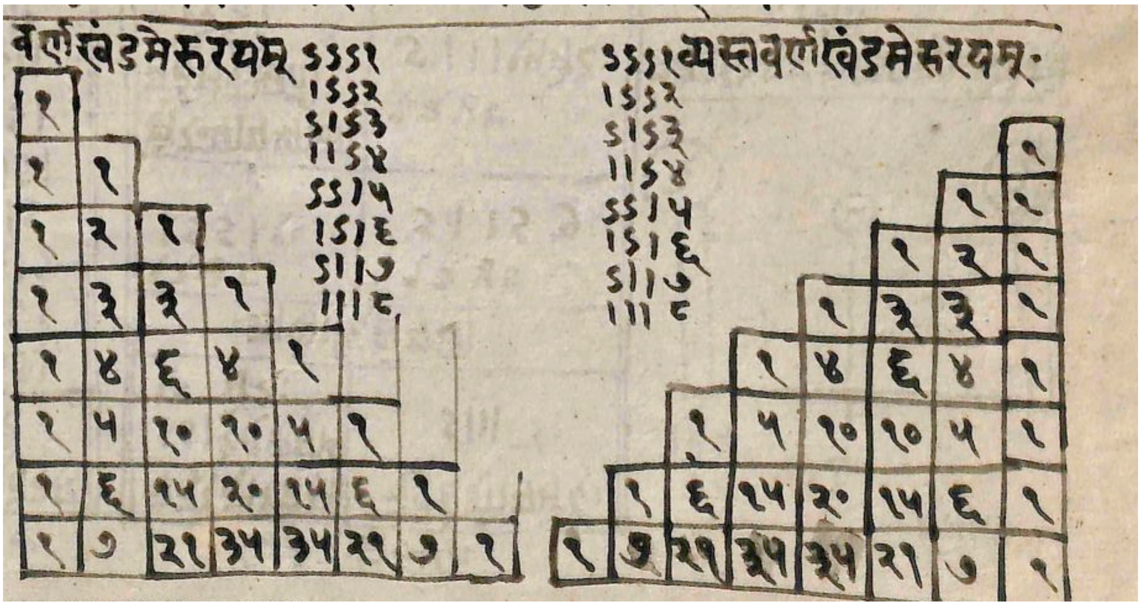
L'instruction val-dict = dict() établit un lien entre la « clé » (ici, le couple d'entiers (a, b)) et la valeur calculée pour ce couple (ici la valeur $probaparti(a, b)$). Avant l'appel récursif, on regarde si la valeur du couple est déjà dans le dictionnaire et, dans ce cas, on renvoie la valeur déjà calculée, sinon on effectue l'appel récursif, le processus continue, et, dès qu'une valeur est renvoyée, on la met dans le dictionnaire. En éliminant ainsi les calculs redondants, on gagne évidemment beaucoup en temps de calcul : pour le couple (15,25), le résultat est immédiat avec ce nouvel algorithme. C'est le cas également avec le couple (300,301).

Par contre, on utilise plus de mémoire ; la complexité en espace de mémoire est donc plus importante. On n'a par ailleurs rien gagné sur le problème de la hauteur de l'arbre, qui est un problème propre au langage, et nous aurons toujours des messages « RecursionError » pour le couple (500,501) ou tout couple pour lequel la hauteur de l'arbre dépasse

1000. Enfin, même si le gain en temps de calcul est amélioré, rechercher une valeur dans le dictionnaire coûte en temps de calcul. Pour améliorer ce dernier point, on peut construire l'arbre complet ligne à ligne ; comme on l'a déjà remarqué, pour chaque ligne de l'arbre, la somme $a+b$ est fixe. On peut construire cet arbre avec une fonction récursive, mais nous aurons alors toujours le même problème de « RecursionError » avec le langage Python. Or toute fonction récursive peut s'écrire avec une boucle. Nous donnons ci-dessous un algorithme permettant de calculer toutes les rangées de l'arbre, ligne par ligne, la variable *sum* étant la somme $a+b$, qui est un « invariant » de la ligne. L'algorithme calcule la probabilité de gagner du joueur A lorsqu'il manque a points à A et b points à B, en faisant varier la somme $a+b$ de 3 à $a+b$ et range dans un dictionnaire (*tree*) toutes les valeurs obtenues pour une somme donnée au cours du calcul (appelées *tree*$[(x,y)]$ dans l'algorithme ci-dessous).

```
 8
 9 def probaparti(a,b):
10
11     if a<0 or b<0 or (a==0 and b==0):
12         print('Mauvaises entrées')
13
14         return -1
15
16     elif a==0:
17         return 1
18     elif b==0:
19         return 0
20     elif a==b:
21         return 0.5
22     else:
23         tree = dict()
24         tree[(1,1)] = 0.5
25         for sum in range (3, a+b+1):
26             tree[(sum-1, 1)] = 0.5 * tree[(sum-2, 1)]
27             tree[(1, sum-1)] = 1.0 - tree[(sum-1, 1)]
28             for x in range (2, sum-1):
29                 y = sum - x
30                 r = 0.5 * tree[(x-1, y)] + 0.5 * tree[(x, y-1)]
31                 tree[(x, y)] = r
32         return tree[(a, b)]
33
34 a = input('Tapez le nombre a:')
35 b = input('Tapez le nombre b:')
36
37 a = int(a)
38 b = int(b)
39 proba = probaparti(a,b)
40 print(proba)
41
```

Regardons cet algorithme en détail : si $a=0$ ou $b=0$ ou $a=b$, l'algorithme renvoie directement les valeurs connues de $\mathrm{P}(a,\, b)$ (lignes 16 à 21). Dans les autres cas, on construit un dictionnaire appelé *tree*, dans lequel on va stocker les valeurs $\mathrm{P}(x,\, y)$ de toutes les lignes de l'arbre, pour les sommes $x+y$ allant de 3 à $a+b$ (il est inutile de calculer ces valeurs pour la somme 2, pour laquelle les seuls couples d'entiers sont (2,0), (1,1) et (0,2)). Rappelons que, dans le langage Python, l'instruction « for *sum* in range (3, $a+b+1$) fait prendre à la variable *sum* les valeurs entières n vérifiant $3 \leq n \leq a+b$. On stocke dans le dictionnaire la valeur $\mathrm{P}(1{,}1) = 0.5$ (ligne 24), puis on construit l'arbre des valeurs de $\mathrm{P}(x,\, y)$. Lorsque la variable *sum* prend la valeur n, on commence par ranger dans le dictionnaire les valeurs $\mathrm{P}(n-1{,}1)$ et $\mathrm{P}(1{,}n-1)$, puis on fait varier x de 2 à $n-2$, on calcule $\mathrm{P}(x, n-x)$ et on le range dans le dictionnaire : on a ainsi stocké toutes les valeurs $\mathrm{P}(x,\, y)$ telles que $x+y=n$, avec $1 \leq x \leq n-1$. Le calcul de $\mathrm{P}(x, n-x)$ pour $2 \leq x \leq n-2$ a nécessité les valeurs $\mathrm{P}(x-1, n-x)$ et $\mathrm{P}(x, n-1-x)$ où $1 \leq x-1 \leq n-3$, qui avaient été calculé pour l'étape précédente, lorsque *sum* était égale à $n-1$. On peut remarquer que, lorsque *sum*=3, on



triangle des coefficients binomiaux pingala

Manuscrit de la bibliothèque de Raghunath J&K ; 755 après J.-C.

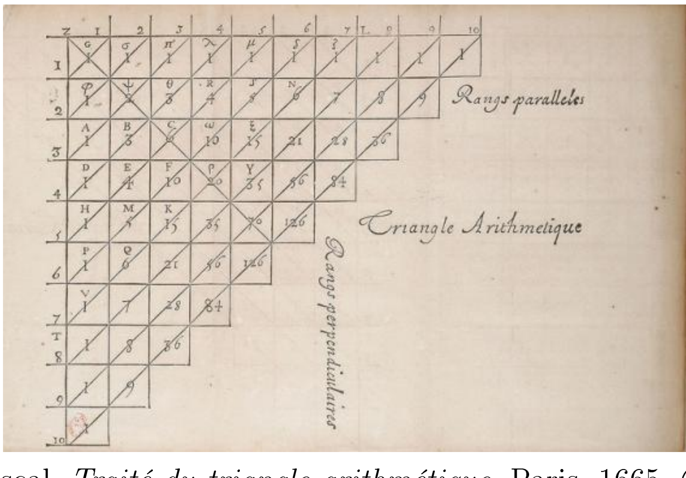
https://archive.org/details/PrakritPingalaPrastaraVarnaMatraPatakadiYantrani775GhaAlm4Shlf3DevanagariAlankarShastra/page/n10/mode/1up calcule et on stocke P(2,1) et P(1,2), mais on n'entre pas dans la boucle pour la variable x, car, pour $n = 3$, $n - 2 < 2$. Ainsi, on a calculé toutes les valeurs de P(x, y) telles que $x + y \leq a + b$, donc on a, en particulier, calculé P(a,b).

Cette fois-ci, on obtient immédiatement le résultat pour le couple (500,501) : 0.5126125090891802 (évidemment très proche de 0.5). Vous trouverez dans la rubrique « Dans nos classes » un problème qu'on peut proposer à des élèves de terminale générale ayant choisi la spécialité mathématique, qui permet d'aborder la récursivité et les problèmes de complexité en temps et en espace. Le programme suggère d'implémenter un algorithme permettant de construire le triangle arithmétique de Pascal, ce qui peut se faire de la même manière que pour construire l'arbre ci-dessus. Ce programme optimise le temps de calcul, mais, par contre, nécessite beaucoup d'espace mémoire et peut éventuellement poser des problèmes pour de très grandes valeurs des deux variables, si le programme utilisait toute la mémoire disponible de votre ordinateur ; l'OS peut décider de tuer un processus qui consomme trop de mémoire, mais cela ne fonctionne pas toujours. Nous vous conseillons donc la prudence avec ces programmes quant au choix des valeurs de (a, b). On peut cependant résoudre ce problème de mémoire. En effet, ce programme consomme beaucoup de mémoire car à la fin de son exécution, il a en mémoire (dans le dictionnaire *tree*) toutes les valeurs de l'arbre. Mais ce n'est pas nécessaire : pour calculer une nouvelle ligne de l'arbre, il n'a besoin que de la ligne précédente. Vous trouverez sur le site de l'IREM une version du programme qui résout le problème de consommation mémoire : au lieu d'utiliser un dictionnaire qui va contenir tout l'arbre, il utilise un nouveau dictionnaire pour chaque ligne. À chaque itération de la boucle principale sur *sum*, on ne garde en mémoire que deux dictionnaires, celui qui contient les valeurs de la ligne actuelle, et celui de la ligne précédente. Les dictionnaires construits par les autres itérations ne sont plus en mémoire. Cette version résout uniquement le problème de consommation mémoire, mais n'est pas plus rapide.

Le *Triangle Arithmétique*

La définition du Triangle

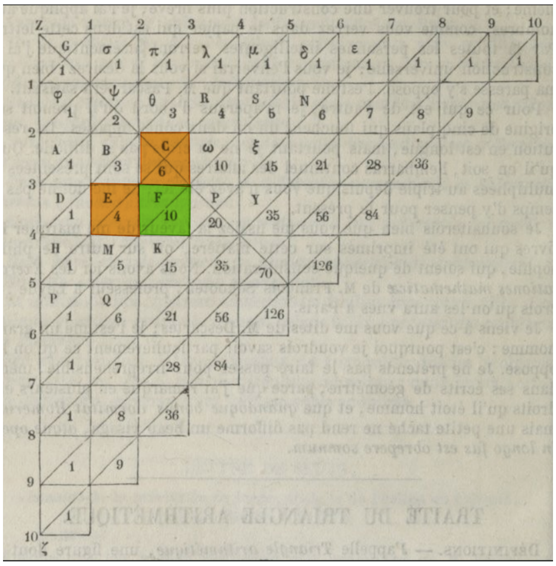
La figure provient du site de la BNF :



Blaise Pascal, *Traité du triangle arithmétique*, Paris, 1665. © B.N.F.
http://expositions.bnf.fr/pascal/grand/bla_070.htm

Les petits carrés sont des *cellules*, les *rangs parallèles* sont les lignes horizontales de cellules, les *rangs perpendiculaires* sont les lignes verticales, les nombres au-dessus du triangle s'appellent les *exposants* des rangs perpendiculaires et ceux à gauche du triangle les *exposants* des rangs parallèles, les « diagonales » sont les *bases* (comme par exemple les cellules contenant D, B, θ, λ, qui forment la quatrième base).

Le nombre de la première cellule est *a priori* arbitraire et s'appelle le *générateur* du triangle, mais, dans la pratique, les différents usages du triangle arithmétique utilisent le générateur 1, et les propriétés sont énoncées avec ce même générateur égal à l'unité, avec un commentaire sur ce que devient la propriété pour un générateur différent. Nous nous limiterons dans cet article aux propriétés et usages du triangle de générateur 1.



Génération du triangle

> « Le nombre de chaque cellule est égal à celui de la cellule qui la précède dans son rang perpendiculaire, plus à celui de la cellule qui la précède dans son rang parallèle. »

Exemple : F = E + C

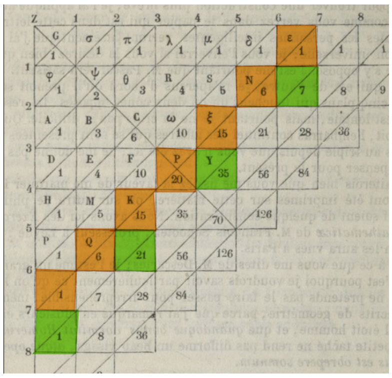
Quelques propriétés du triangle arithmétique

Pascal donne ensuite les conséquences de la génération du triangle, en appelant plus simplement cellule « le nombre de la cellule ». Par exemple, les cellules des premiers rangs parallèle et perpendiculaire sont toutes égales au générateur, Pascal considérant que, lorsqu'une cellule n'a pas de cellule qui la précède, on ajoute 0 au lieu de la celllule précédente. Chaque cellule est égale à sa réciproque (obtenue en échangeant les exposants des rangs parallèle et perpendiculaire). Les justifications sont données pour un exemple générique. Nous donnons ci-dessous les conséquences utilisées plus loin par Pascal, en utilisant un coloriage des cellules [10] pour faciliter la compréhension.

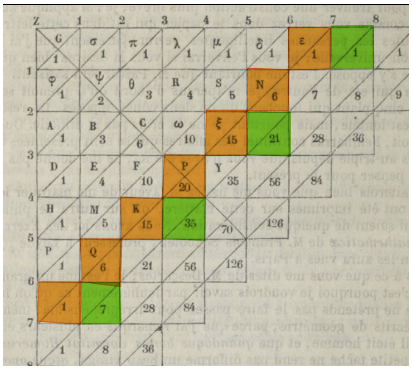
« *Conséquence septième*

[...] la somme des cellules de chaque base est double de celles de la base précédente »

10. Les coloriages ne sont pas dans le texte original.



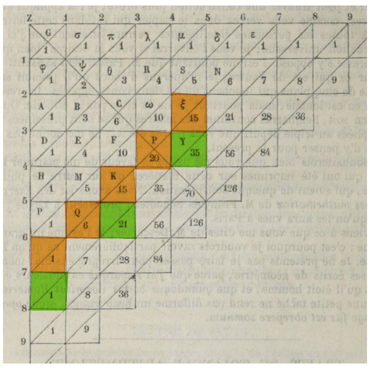
Somme des cellules oranges = somme des cellules vertes



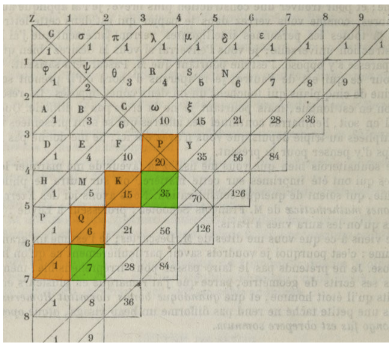
Somme des cellules oranges = somme des cellules vertes

Finalement, la somme des cellules de la base formée des cellules vertes est le double de la somme des cellules de la base formée des cellules oranges, c'est-à-dire de celles de la base précédente.

> « *Conséquence dixième*
>
> [...] la somme de tant de cellules continues qu'on voudra de sa base, à commencer par une extrémité, est égale à autant de cellules de la base précédente, plus encore à autant hormis une. »



Somme des cellules oranges = somme des cellules vertes



Somme des cellules oranges = somme des cellules vertes

Finalement la somme des b premières cellules de la $(n-1)^e$ base plus la somme des b–1 premières cellules de la $(n-1)^e$ base est égale à la somme des b premières cellules de la n^e base.

> « *Conséquence douzième*
>
> [...] deux cellules contiguës étant dans une même base, la supérieure est à l'inférieure comme la multitude des cellules depuis la supérieure jusques au haut de la base à la multitude de celles depuis l'inférieure jusques en bas inclusivement. »

Pascal donne l'exemple des cellules E et C, contiguës dans la 5^e^base : E est à C comme 2 à 3 (autrement dit $\frac{E}{C}=\frac{2}{3}$) , car il y a deux cellules depuis E, la cellule inférieure, jusqu'en bas de la base (cellules E et H) et trois cellules depuis C (la cellule supérieure) jusqu'en haut de la base (cellule C, R, μ). Pascal explique le principe de la démonstration :

> « Quoique cette proposition ait une infinité de cas, j'en donnerai une démonstration bien courte, en supposant deux lemmes.
>
> Le 1er, qui est évident de soi-même, que cette proportion se rencontre dans la seconde base ; car il est bien visible que φ est à σ comme 1 est à 1.
>
> Le 2e, que si cette proportion se trouve dans une base quelconque, elle se trouvera nécessairement dans la base suivante.
>
> D'où il se voit qu'elle est nécessairement dans toutes les bases : car elle est dans la seconde base par le premier lemme ; donc par le second elle est dans la troisième base, donc dans la quatrième, et à l'infini. »

On voit là très clairement posé le principe de la démonstration par récurrence. Le deuxième lemme est démontré sur un exemple générique.

> « Il faut donc seulement démontrer le second lemme, en cette sorte. Si cette proportion se rencontre en une base quelconque, comme en la quatrième Dλ, c'est-à-dire si D est à B comme 1 à 3, et B à θ comme 2 à 2, et θ à λ comme 3 à 1, etc. ;
>
> Je dis que la même proportion se trouvera dans la base suivante, Hμ, et que, par exemple, E est à C comme 2 est à 3. Car D est à B comme 1 à 3, par l'hypothèse.
>
> Donc $\underbrace{D+B}_{E}$ est à B comme $\underbrace{1+3}_{4}$ est à 3 [c'est-à-dire E est à B comme 4 à 3].
>
> De même B est à θ comme 2 à 2, par l'hypothèse.
>
> Donc $\underbrace{B+\theta}_{C}$ est à B comme $\underbrace{2+2}_{4}$ à 2 [c'est-à-dire C à B comme 4 à 2].
>
> Mais B à E comme 3 à 4, comme il est montré.
>
> Donc, par la proportion troublée, C est à E comme 3 à 2.

> Ce qu'il fallait démontrer.
>
> On le montrera de même dans tout le reste, puisque cette preuve n'est fondée que sur ce que cette proportion se trouve dans la base précédente, et que chaque cellule est égale à sa précédente, plus à sa supérieure, ce qui est vrai partout. » [11]

Il est facile, avec des notations modernes, de montrer ce deuxième lemme de manière tout à fait générale. Considérons deux cellules contiguës de la $(n+1)^e$ base, c_k et c_{k+1}, de rangs parallèles respectifs k et $k+1$. On a : $c_k = b_k + b_{k-1}$ et $c_{k+1} = b_{k+1} + b_k$ où, b_{k-1}, b_k, b_k sont trois cellules contiguës de la n^e base, de rangs parallèles respectifs, $k-1$, k et $k+1$. L'hypothèse de récurrence assure $\frac{b_k}{b_{k-1}} = \frac{n-k+1}{k-1}$ et $\frac{b_{k+1}}{b_k} = \frac{n-k}{k}$. Donc, $\frac{b_{k-1}}{b_k} = \frac{k-1}{n-k+1}$; on en déduit : $\frac{b_{k-1}+b_k}{b_k} = \frac{k-1+n-k+1}{n-k+1} = \frac{n}{n-k+1}$, c'est-à-dire $\frac{c_k}{b_k} = \frac{n}{n-k+1}$. De même $\frac{b_{k+1}+b_k}{b_k} = \frac{k+n-k}{k} = \frac{n}{k}$, c'est-à-dire $\frac{c_{k+1}}{b_k} = \frac{n}{k}$. On en tire alors $\frac{c_k}{c_{k+1}} = \frac{c_k}{b_k} \times \frac{b_k}{c_{k+1}} = \frac{k}{n-k+1} = \frac{k}{(n+1)-k}$. Ce qui assure l'hérédité.

Cette conséquence douzième permet à Pascal de résoudre le problème consistant à déterminer le nombre d'une cellule à l'aide des exposants k et l des rangs parallèle et perpendiculaire, sans utiliser le Triangle arithmétique. En effet, la cellule considérée c_k est alors la cellule de rang parallèle k de la n^e base, avec $n = k+l-1$. On a donc : $\frac{c_k}{1} = \frac{c_k}{c_n} = \frac{c_k}{c_{k+1}} \times \frac{c_{k+1}}{c_{k+2}} \times \ldots \times \frac{c_{k+l-2}}{c_n} = \frac{k}{n-k} \times \frac{k+1}{n-k-1} \times \frac{k+2}{n-k-2} \times \ldots \times \frac{k+l-2}{1} = \frac{k \times (k+1) \times \ldots \times (k+l-2)}{(l-1) \times (l-2) \times \ldots \times 1} = \frac{(n-1)!}{(k-1)! \times (l-1)!}$ avec $n-1 = (k-1)+(l-1)$.

Les combinaisons dans le *Traité du Triangle Arithmétique*

Pascal donne « divers usages du triangle arithmétique », le second usage s'appliquant aux combinaisons. Il donne une définition des combinaisons :

> « Lorsque de plusieurs choses, on donne le choix d'un certain nombre, toutes les manières d'en prendre autant qu'il est permis entre toutes celles qui sont présentées s'appellent ici *les différentes combinaisons*.
>
> Par exemple, si de quatre choses exprimées par ces quatre lettres, on permet d'en prendre, par exemple, deux quelconques, toutes les manières d'en prendre deux différentes dans les quatre qui sont proposées s'appellent *Combinaisons*.
>
> Ainsi on trouvera, par expérience, qu'il y a six manières différentes d'en choisir deux dans quatre ; car on peut prendre A et B, ou A et C, ou A et D, ou B et C, ou B et D, ou C et D.
>
> Je ne compte pas A et A pour une des manières d'en prendre deux ; car ce ne sont pas des choses différentes, ce n'en est qu'une répétée.
>
> Ainsi, je ne compte pas A et B puis B et A pour deux manières différentes ; car on ne prend en l'une et l'autre manière que les deux mêmes choses, mais d'un ordre différent seulement ; et je ne prends point garde à l'ordre ; de sorte que je pouvais m'expliquer en un mot, en disant simplement que je parle seulement des combinaisons qui se font sans changer l'ordre ». [12]

Puis il démontre sur un exemple générique la relation de récurrence sur ces combinaisons. Ce que Pascal nomme la multitude des combinaisons d'un nombre k dans un nombre n est ce que nous notons $\binom{n}{k}$.

11. Pascal B., *Œuvres complètes*, tome II, texte présenté, établi et annoté par Jean Mesnard, Desclée de Brouwer, 1970, Ré-édition Alençon, 1990, p. 1294-1295.

12. Pascal B., *Œuvres complètes*, tome II, texte présenté, établi et annoté par Jean Mesnard, Desclée de Brouwer, 1970, Ré-édition Alençon, 1990, p. 1302.

« *Lemme 4*

S'il y a quatre nombres quelconques, le premier tel qu'on voudra, le second plus grand de l'unité, le troisième tel qu'on voudra, pourvu qu'il ne soit pas moindre que le second, le quatrième plus grand de l'unité que le troisième : la multitude des combinaisons du premier dans le troisième, jointe à la multitude des combinaisons du second dans le troisième, égale la multitude des combinaisons du second dans le quatrième.

Soient quatre nombres tels que j'ai dit :

Le premier tel qu'on voudra, par exemple : 1.

Le second plus grand de l'unité, savoir 2.

Le troisième tel qu'on voudra, pourvu qu'il ne soit pas moindre que le second, par exemple, 3.

Le quatrième plus grand de l'unité, savoir, 4.

Je dis que la multitude des combinaisons de 1 dans 3, plus la multitude des combinaisons de 2 dans 3, égale la multitude des combinaisons de 2 dans 4. Soient trois lettres quelconques, B, C, D. Soient les mêmes trois lettres, et une de plus, A, B, C, D.

[...] Il faut démontrer que la multitude des combinaisons de 1 dans 3 et celles de 2 dans 3 égalent celles de 2 dans 4.

Cela est aisé, car les combinaisons de 2 dans 4 sont formées par les combinaisons de 1 dans 3, et par celles de 2 dans 3.

Pour le faire voir, il faut remarquer qu'entre les combinaisons de 2 dans 4, savoir, AB, AC, AD, BC, BD, CD, il y en a où la lettre A est employée, et d'autres où elle ne l'est pas. Celles où elle n'est pas employée sont BC, BD, CD, qui par conséquent sont formées de deux de ces trois lettres, B, C, D ; donc ce sont des combinaisons de 2 dans ces trois, B, C, D. Donc les combinaisons de 2 dans ces trois lettres B, C, D, font portion des combinaisons de 2 dans ces quatre lettres, A, B, C, D, puisqu'elles forment celles où A n'est pas employée.

Maintenant si des combinaisons de 2 dans 4 est employée, savoir AB, AC, AD, on ôte l'A, il restera une lettre seule dans les trois, B, C, D seulement de ces trois B, C, D, savoir, B, C, D, qui sont précisément les combinaisons d'une lettre dans les trois, B, C, D. Donc si aux combinaisons d'une lettre dans les trois, B, C, D, on ajoute à chacune la lettre A, et qu'ainsi on ait AB, AC, AD, on formera les combinaisons de 2 dans 4, où A est employée ; donc les combinaisons de 1 dans 3 font portion des combinaisons de 2 dans 4.

D'où il se voit que les combinaisons de 2 dans 4 sont formées par les combinaisons de 2 dans 3, et de 1 dans 3 ; et partant que la multitude des combinaisons de 2 dans 4 égale celle de 2 dans 3, et de 1 dans 3.

On montrera la même chose dans tous les autres exemples [...] » [13]

Pascal fait ensuite le lien avec le triangle arithmétique, avec deux propositions dont la deuxième affirme :

> « Le nombre de quelque cellule que ce soit égale la multitude des combinaisons d'un nombre moindre de l'unité que l'exposant de son rang parallèle, dans un nombre moindre de l'unité que l'exposant de sa base ».

Autrement dit, le nombre de la cellule de rang parallèle k dans la n^e base est le nombre de combinaisons de $k-1$ dans $n-1$, ce qui ne nous surprend pas après la formule démontrée plus haut pour ce nombre. Nous ne donnons pas la démonstration de cette propriété,

13. Pascal B., *Œuvres complètes*, tome II, texte présenté, établi et annoté par Jean Mesnard, Desclée de Brouwer, 1970, Ré-édition Alençon, 1990, P. 1304-1305.

que Pascal démontre comme conséquence d'une propriété sur la somme des cellules d'un même rang parallèle. On pourrait la montrer tout simplement en utilisant le mode de génération du triangle et la relation de récurrence sur les combinaisons. Mais l'utilisation pédagogique de cette justification paraît difficile ; certes, la formule pour les combinaisons est au programme de la spécialité de terminale générale, mais le programme demande de donner deux démonstrations de la relation de récurrence sur les combinaisons, l'une utilisant la définition des combinaisons, comme l'a fait Pascal sur un exemple générique, l'autre utilisant la formule de calcul des combinaisons : il ne serait donc guère cohérent de démontrer la formule en utilisant cette même relation de récurrence !

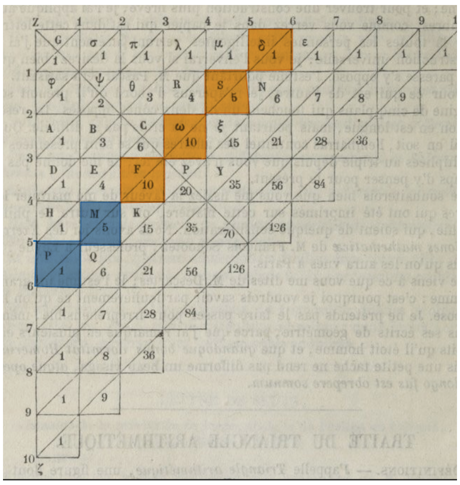
Problème des partis et triangle arithmétique

Rappel du principe de partage des mises

Rappelons le corollaire second sur le partage des mises lorsque les deux joueurs doivent arrêter le jeu lorsque'il manque un certain nombre de parties à chacun pour achever le jeu : « Si deux joueurs sont en la même condition qu'on vient de dire, je dis que le parti se peut faire de cette façon qui revient au même ; que l'on assemble les deux sommes de gain et de perte, et que le premier prenne la moitié de cette somme ; [...] ». Pascal explicite ce corollaire dans un lemme, en précisant que, s'il appartient à un joueur une certaine fraction (disons α) de la mise totale s'il gagne la partie, et une fraction moindre (β) s'il la perd, ce joueur doit emporter la fraction $\frac{\alpha+\beta}{2}$ de la mise totale. Pascal explique même comment prendre la demi-somme de deux fractions réduites au même dénominateur. Il est clair pour nous que ces fractions α et β sont les probabilités que le joueur gagne le jeu dans les cas où il gagne ou perd la partie suivante, si elle était jouée.

Lien avec le triangle arithmétique

Pascal donne la solution générale au problème, et donne des exemples pour faire comprendre ce calcul. Prenons le premier exemple de Pascal : il manque 2 parties à A et 4 parties à B. La mise totale est M. On considère la 6[e] base, car 6 est la somme des nombres de parties qui manquent à chacun des joueurs.

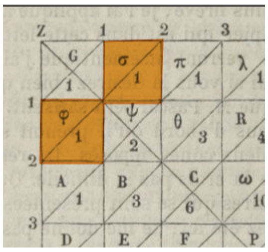


Règle de partage

A emporte la fraction suivante de la mise M : $\frac{F+\omega+S+\delta}{P+M+F+\omega+S+\delta}$, c'est-à-dire la fraction dont le numérateur est la somme des 4 cellules du « haut » de la base (4 étant le nombre de parties qui manquent au joueur A) et le dénominateur la somme de toutes les cellules de la base. Le joueur B emporte la fraction $\frac{P+M}{P+M+F+\omega+S+\delta}$.

D'une manière générale, s'il manque a points à A et b points à B, avec $n = a + b$, le « parti » est donné par la n^e base : la fraction de la mise totale qu'emporte A a pour numérateur la somme des b cellules « du haut » (ou par symétrie du triangle des b cellules du bas) et pour dénominateur la somme de toutes les cellules de la base. La démonstration se fait par récurrence sur n.

> « Quoique cette proposition ait une infinité de cas, je la démontrerai néanmoins en peu de mots par le moyen de deux lemmes.
>
> Le 1er, que la seconde base contient les partis des joueurs auxquels il manque deux parties en tout.
>
> Le 2e, que si une base quelconque contient les partis de ceux auxquels il manque autant de parties que de cellules, la base suivante sera de même, c'est-à-dire qu'elle contiendra aussi les partis des joueurs auxquels il manque autant de parties qu'elle a de cellules.
>
> D'où je conclus, en un mot, que toutes les bases du Triangle arithmétique ont cette propriété : car la seconde l'a par le premier lemme ; donc, par le second lemme, la troisième l'a aussi, et par conséquent la quatrième ; et à l'infini.
>
> Il faut donc seulement démontrer ces deux lemmes.
>
> Le 1er est évident de lui-même ; car s'il manque une partie à l'un et une à l'autre, il est évident que leurs conditions sont comme φ à σ, c'est-à-dire comme 1 à 1, et qu'il appartient à chacun cette fraction : $\frac{\sigma}{\varphi+\sigma}$ qui est $\frac{1}{2}$ ».



Cas où $a + b = 2$

Pascal ne considère qu'un cas pour la somme $a + b = 2$, car il ne considère pas les cas évidents où $a = 0$ ou $b = 0$, qu'il a donnés dans son premier exemple, après ses principes, exemple qu'il juge « peut-être mal à propos de toucher, parce qu'il est trop clair ; [...] Si à un des joueurs il ne manque aucune partie, et à l'autre quelques-unes, la somme entière appartient au premier ».

Le deuxième lemme est prouvé sur l'exemple générique du passage de la quatrième base à la cinquième, en montrant que « s'il manque, par exemple, deux parties au premier, et trois à l'autre, la portion qui appartient au premier sur la somme qui se joue, est exprimée par cette fraction : $\frac{H+E+C}{H+E+C+R+\mu}$ » avec l'« hypothèse de récurrence » : la quatrième base donne la portion appartenant au premier joueur quand la somme du nombre de parties manquantes est 4 ; en particulier, s'il manque 1 partie au premier joueur et 3 au second joueur, la portion de la mise totale revenant au premier est $\frac{D+B+\theta}{D+B+\theta+\lambda}$ et s'il manque 2 parties à l'un et 2 à l'autre, la fraction qui appartient au premier est $\frac{D+B}{D+B+\theta+\lambda}$ (autrement dit ici $\frac{1}{2}$).

Pour calculer la part qui revient au premier joueur (A) s'il manque 2 parties au premier et 3 au second, rappelons le principe vu plus haut : on doit faire la demi-somme des fractions qui lui appartiennent en cas de gain et en cas de perte. Or, en cas de gain de la partie suivante, il manquera 1 partie au premier et 3 au second, donc A emportera la fraction $\alpha = \frac{D+B+\theta}{D+B+\theta+\lambda}$ et, en cas de perte, il manquera 2 partie au premier et 2 au second, donc A emportera la fraction $\beta = \frac{D+B}{D+B+\theta+\lambda}$. Donc, la part que nous cherchons est $\frac{\alpha+\beta}{2}$, soit $\frac{D+B+\theta+D+B}{2(D+B+\theta+\lambda)}$. Or, le double de la somme des cellules de la quatrième base est la somme des cellules de la cinquième base, donc $2(D+B+\theta+\lambda) = H+E+C+R+\mu$ et $D+B+\theta+D+B = H+E+C$ par la conséquence dixième. Ce qui montre bien que la fraction de la mise totale que doit remporter A est $\frac{H+E+C}{H+E+C+R+\mu}$. Pascal fait remarquer que « le fondement de cette preuve est qu'une base est toujours double de sa précédente, par la 7e conséquence » combinée avec la 10e conséquence.

On peut écrire le raisonnement sur le cas général. Supposons qu'il manque a points à A et b points à B, avec $n = a + b$. Si A gagne la partie suivante, il lui manquera alors $a - 1$ parties, et toujours b à B. L'hypothèse de récurrence nous affirme que la fraction emportée par A se trouve dans la $(n-1)^e$ base, avec comme numérateur la somme des b premières cellules. Si au contraire A perd, il lui manquera toujours a parties, mais $b-1$ seulement à B. Cette fois, la fraction emportée par A se trouve dans la $(n-1)^e$ base, avec comme numérateur la somme des $b-1$ premières cellules. Et nous avons vu que la septième conséquence et la dixième conséquence permettent de justifier que la demi-somme de ces deux fractions est la fraction dont le numérateur est la somme des numérateurs, et le dénominateur le double du dénominateur commun. Le dénominateur est donc le double de la somme des cellules de la $(n-1)^e$ base, qui est la somme des cellules de la n^e base. Le numérateur est, quant à lui, la somme des b premières cellules de la $(n-1)^e$ base plus la somme des $b-1$ premières cellules de la $(n-1)^e$ base. On a vu que cette somme est égale à la somme des b premières cellules de la ne base. Ce qui termine la récurrence.

Ce raisonnement sur le triangle est sans doute moins utilisable en classe. Il paraît plus simple pour nous d'utiliser tout simplement les combinaisons pour obtenir des formules générales, en raisonnant, comme le fait Fermat dans sa correspondance avec Pascal, sur le nombre maximum de parties qu'il resterait à jouer pour que « le jeu soit décidé absolument ». En effet, s'il manque a points à A et b points à B, le jeu sera décidé en au plus $a+b-1$ parties. Le nombre total d'issues possibles de ces $a+b-1$parties est 2^{a+b-1}et le nombre d'issues favorables à A est le nombre de ces issues pour lesquelles A gagne au moins a parties sur ces $a+b-1$ parties, c'est-à-dire, avec nos notations modernes, $\begin{pmatrix} a \\ a+b-1 \end{pmatrix} + \begin{pmatrix} a+1 \\ a+b-1 \end{pmatrix} + \begin{pmatrix} a+2 \\ a+b-1 \end{pmatrix} + \ldots + \begin{pmatrix} a+b-1 \\ a+b-1 \end{pmatrix}$. Ces résultats généraux permettent à Pascal de s'intéresser à d'autres problèmes sur ces répartitions de mise (*Mnémosyne* n°6).

Conclusion

La lecture des textes de Pascal amène ainsi à des réflexions sur des problèmes divers, qui croisent des thèmes importants pour notre enseignement, qu'il s'agisse de probabilités par le biais du partage des mises, d'algorithmique par la méthode de Pascal, du principe de récurrence ou des problèmes de complexité en mémoire ou en temps en informatique. Il nous paraît intéressant d'aborder un problème aussi riche avec les élèves, d'autant plus que la plupart des exercices d'algorithmique en probabilités ne sont que des exercices de simulation, alors qu'ici, nous avons un algorithme récursif de calcul des probabilités, dont la version « naïve » est tout à fait accessible à nos élèves. Un travail bidisciplinaire entre collègues de mathématiques et d'informatique enrichirait les deux disciplines, du moins si la structure des groupes de spécialité permet un tel échange.

Remerciements

Nous remercions Antoine Meyer [14], qui nous a initiées à l'algorithmique dans le groupe « Algorithmique » de l'IREM de Paris-Diderot, et nous a le premier parlé des problèmes de complexité qui surgissent lors de l'implémentation de l'algorithme de Pascal, lors d'une séance de lecture des écrits de Pascal. Nous n'avions pas alors creusé ces questions, la connaissance d'aucun langage de programmation n'étant à ce moment exigible pour les élèves. Nous avons repris ce texte après les changements de programmes de 2019, demandant explicitement aux collègues de mathématiques de faire programmer les élèves en Python, et nous sommes penchées sur l'écriture des algorithmes en Python et les problèmes rencontrés lors de l'exécution des programmes. Nous remercions Pauline Brunet [15] pour son aide à l'écriture des programmes et David Bühler [16] pour avoir répondu à nos questions sur l'exécution de ces programmes.

Très brève bibliographie

Articles

Coumet, E. « Le problème des partis avant Pascal », In *Archives Internationales d'histoire des Sciences*, 18/73, 1965, pp245-272.
http://www.jehps.net/Juin2007/Coumet_partis.pdf

Guillemot M., « Le triangle arithmétique à travers les âges », *Bulletin Vert* n°416, Paris, 1998, p. 351-362.https://www.apmep.fr/IMG/pdf/AAA98031.pdf

Henry M. « Simulations d'expériences aléatoires en classe. Un enjeu didactique pour comprendre la notion de modèle probabiliste, un outil de résolution de problèmes. » In *Bulletin de l'APMEP*. N° 496. p. 536-550, APMEP, Paris, 2011.

Legrand P. « La récurrence au fil des siècles. », In *Bulletin de l'APMEP*. N°506. p. 600-610, APMEP, Paris, 2013.
http://numerisation.univ-irem.fr/AAA/AAA13070/AAA13070.pdf

Meunier N. « Le problème des partis avant Pacioli », In *Histoire de probabilités et de statistiques*, Commission Inter-IREM d'Épistémologie et d'Histoire des mathématiques, Ellipses, Paris, 2004.

Textes originaux

Fermat P. *Œuvres*, publiées par P. Tannery et C. Henry, tome II, Gauthier-Villars, Paris, 1894. Disponible sur gallica.bnf.fr

Pascal B., *Œuvres complètes*, tome II, texte présenté, établi et annoté par Jean Mesnard, Desclée de Brouwer, 1970.

14. Maître de conférences à l'Université Paris-Est, responsable du groupe « algorithmique » de l'IREM Paris-Diderot.
15. Chercheuse au Laboratoire d'Intégration des Systèmes et des Technologies du C.E.A.
16. Chercheur au Laboratoire d'Intégration des Systèmes et des Technologies du C.E.A.

Pascal B., *Œuvres complètes*, Tome III, Physique et mathématique, Hachette, Paris, 1869-1872. Disponible sur gallica.bnf.fr.

Documents pour la classe du groupe M. : A.T.H.

La rubrique « Dans nos classes » de ce numéro présente des activités pour la classe autour de la solution de Pascal au problème des partis, et vous trouverez également des compte-rendus de séances en classe sur les probabilités dans les références ci-dessous.

Brochures (disponibles en ligne sur la page du groupe, site de l'IREM de Paris Diderot) : les pages citées ci-dessous comportent des extraits de textes et les énoncés donnés aux élèves.

Brochure M. : A.T.H. n° 61 de l'IREM Paris : Extrait du *Triangle arithmétique* de Pascal et extrait d'un texte de Montmort (pp 83-94) ; le problème des partis de Pacioli à Pascal et Fermat (pp 99-147).

Brochure M. : A.T.H. n° 79 de l'IREM Paris : Extrait du *Triangle arithmétique* de Pascal (pp 99-106).

Revue *Mnémosyne* du groupe M. : A.T.H. n°6 : une généralisation du problème des partis avec un texte de Pascal (pp. 15-24).

Dossier « Utilisation de l'histoire des mathématiques en probabilités » du groupe M. : A.T.H.
https://irem.u-paris.fr/utilisation-de-lhistoire-des-mathematiques-en-probabilites