LE JEU DU TAQUIN

Pierre-Alain Cherix

Section de mathématiques de l'Université de Genève

CE QUI NOUS A MOTIVÉS.

Le jeu comme outil d'apprentissage, l'idée n'est pas nouvelle et nous avons pu le constater en maintes occasions. Dans cet article nous allons décrire une expérience faite sur une durée d'une année avec des lycéens de première et de terminale (16-18 ans), dans le cadre du programme "MATH.en.JEANS" dans un lycée de Haute-Savoie auquel mon collègue, Martin Gander, et moi-même participons en tant que chercheurs depuis quelques années déjà. Parmi les questions mathématiques diverses que nous avons été amenés à poser, nous avons réalisé que les mathématiques discrètes, en particulier les mathématiques énumératives[1] avaient un certain succès, surtout s'il était possible, pour les élèves, de conjecturer certains résultats en utilisant des approches informatiques qui permettent de simuler certaines situations.

Dans le cas d'un puzzle, la recherche d'une solution via un programme permet de mettre en œuvre des stratégies plus ou moins efficaces. Cet article contient quelques réflexions sur un puzzle très connu, le taquin.

UN PEU D'HISTOIRE.

Dans notre enfance nous avons certainement tous eu un jour dans les mains l'objet suivant sur lequel nous avons passé plus ou moins de temps suivant notre patience.

Fig. 1 : Le taquin de notre enfance

Le taquin consiste en un cadre dont le côté est un peu plus grand que 3 unités (respectivement 4) dans lequel coulissent huit pavés carrés de côté un numérotés de 1 à 8 (respectivement 15 carrés numérotés de 1 à 15). Le dernier espace est vide et permet de faire glisser les pièces sur le jeu. L'image précédente montre un taquin quatre fois quatre, mais pour ce travail, nous nous concentrerons sur un taquin trois fois trois. À partir de maintenant nous considérerons donc des taquins trois fois trois. Un taquin se présente donc comme cela :

[1] Face à une situation ne comportant qu'un nombre fini, éventuellement très grand, de cas, il est possible de faire une recherche exhaustive, parmi tous ces cas, de ceux comportant certaines caractéristiques. Les mathématiques énumératives développent des méthodes permettant de trouver de manière plus systématique tous les cas pouvant apparaître dans une certaine situation ou tout au moins de compter ou d'estimer le nombre de tels cas. L'archétype de ce type de mathématiques est l'analyse combinatoire, mais les maths discrètes avec des techniques comme les fonctions génératrices ou la théorie des graphes peuvent aussi entrer dans cette dénomination.

3	5	
4	1	2
8	7	6

Fig. 2 : Exemple de taquin 3 fois 3

En faisant coulisser les pièces, mais sans les sortir du plateau, il s'agit de ranger le taquin, en réordonnant les nombres. Le taquin rangé se présente donc comme suit.

1	2	3
4	5	6
7	8	

Fig. 3 : Le taquin rangé

L'inventeur de ce jeu, à la fin du XIX siècle, Sam Loyd fut l'un des inventeurs de jeux et de casse-têtes mathématiques et logiques les plus prolifiques. On lui doit autant des puzzles logiques que des illusions d'optiques tel le fameux "Get of the earth" (Loyd, 2019), des questions calculatoires ou des problèmes énumératifs comme le taquin.

En bon commerçant, pour encourager les clients à acheter son puzzle, Sam Loyd offrait même un dollar aux personnes qui, après avoir acheté son casse-tête 30 cents arrangé comme ci-dessous, auraient réussi à le ramener sur le taquin rangé.

1	2	3
4	5	6
8	7	

Position initiale

1	2	3
4	5	6
7	8	

Position finale

Fig. 4 : Positions initiale et finale du taquin

Sam Loyd n'eut jamais à débourser son dollar. Il savait certainement qu'il n'aurait pas à le faire. Mais les acheteurs pouvaient changer la configuration initiale du taquin. En effet, contrairement au modèle de notre enfance où les pièces ne pouvaient être sorties sans l'utilisation d'un levier, les premiers taquins avaient des carrés numérotés posés librement sur un plateau qui pouvaient être retirés sans autre du plateau.

Imaginons donc la situation suivante : vous renversez votre taquin, les pièces tombent et vous remettez les pièces au hasard. Quelle chance avez-vous, partant de cette configuration, de parvenir à un taquin rangé en utilisant uniquement les mouvements autorisés dans le jeu ? Pour répondre de manière empirique à cette question, une approche informatique est possible. Si on arrive à résoudre efficacement un taquin par ordinateur, il est possible d'estimer quelle est la chance d'obtenir un taquin rangé si on tire au hasard la configuration initiale.

Il existe une réponse théorique à cette question, mais le but de travail fait dans le cadre de MATHS.en.JEANS est de développer une technique rapide de résolution permettant de faire des essais statistiques. Dans l'approche algorithmique choisie, nous allons utiliser le langage de la théorie des graphes. Nous allons donc rapidement définir les notions de graphes (orientés ou non) et d'arbres de manière informelle de telle sorte que les explications qui suivent puissent se comprendre aisément.

UN PEU DE TERMINOLOGIE DE GRAPHES.

Un graphe est un ensemble de points reliés par des traits, appelées des arêtes (dans notre cas nous n'accepterons qu'une arête entre deux sommets donnés et les boucles, c'est-à-dire une arête reliant un sommet à lui-même, ne seront pas considérées). Dans le cas d'un graphe orienté, les arêtes ne sont pas de simples segments, mais des flèches possédant chacune un point de départ et un point d'arrivée. Nous n'allons pas définir plus précisément ces notions, mais les exemples de figures suivantes devraient permettre d'en comprendre l'idée.

Graphe non orienté Graphe orienté

Fig. 5 : Graphe non orienté et orienté

Un chemin est une suite d'arêtes (orientées ou non suivant le contexte) partant d'un sommet et tel que deux arêtes successives dans la suite se rejoignent en un sommet.

Fig. 6 : Chemin et circuit

Un circuit est un chemin dont le sommet initial est le même que le sommet final. Par définition, un arbre est graphe sans circuit. Un arbre peut être un graphe simple ou un graphe orienté. Un arbre enraciné est un arbre orienté et satisfaisant les conditions suivantes :

- Il existe un seul point dont les arêtes qui le touchent partent de celui-ci, ce sommet s'appelle la racine de l'arbre.
- Tout autre sommet S, autre que la racine n'a qu'un prédécesseur P, c'est-à-dire qu'il n'existe qu'un seul sommet P et une seule arête orientée dont le sommet final est S et le sommet initial est P. S s'appelle le successeur de P.

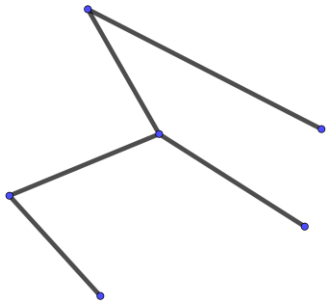
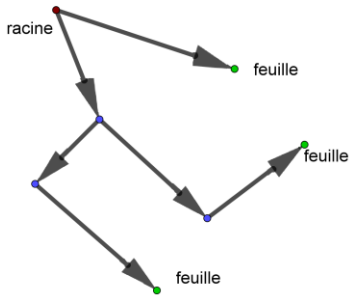


Fig. 7 : Arbre



Arbre enraciné

Dans un arbre enraciné, un sommet n'ayant pas de successeur s'appelle une feuille.

Avec ces outils, nous allons essayer d'expliquer brièvement comment un jeu n'ayant qu'un nombre fini de positions peut être résolu. Nous allons l'expliquer directement sur le taquin, mais cette démarche est possible pour la plupart des jeux.

RÉSOUDRE UN JEU AVEC UN GRAPHE.

Une manière théorique de modéliser un puzzle ou un jeu ayant un nombre d'états fini consiste à donner la liste de tous les états possibles, à relier les paires d'états différant d'un coup de jeu, à repérer parmi eux d'une part ceux correspondant à un début de partie, de l'autre ceux correspondant à une fin de partie. L'objet obtenu est un arbre.

Le taquin n'a qu'un nombre fini de positions initiales, en fait 9! = 9*8*7*6*5*4*3*2 = 362880. Il semble donc facile de résoudre un tel problème informatiquement, vu sa relative petite taille. Mais la difficulté réside dans le fait que les mouvements de base pour changer de configuration sont peu nombreux. Il n'est donc pas évident de construire une stratégie. En effet pour permuter les nombres de certaines cases, il faut déplacer un nombre souvent bien plus important de pièces.

Pour construire une résolution d'un puzzle ou pour étudier une stratégie gagnante pour un jeu, il faut construire, comme expliqué précédemment l'arbre de possibilités de celui-ci en reliant un état initial par une arête orientée à tous les états qu'on peut obtenir par un coup unique et répéter cette construction à partir des nouveaux états obtenus. Pour une explication plus complète de la modélisation d'un jeu ou d'un puzzle, se référer à Webb (2007).

Dans le cas du taquin, cette stratégie consiste, partant de la configuration T_0, à construire l'arbre orienté des possibles de la manière suivante. T_0 est appelé la racine et on construit récursivement l'arbre en créant toutes les configurations obtenues (T_1, T_2, ...) par un seul mouvement de la place libre et en les reliant à la racine.

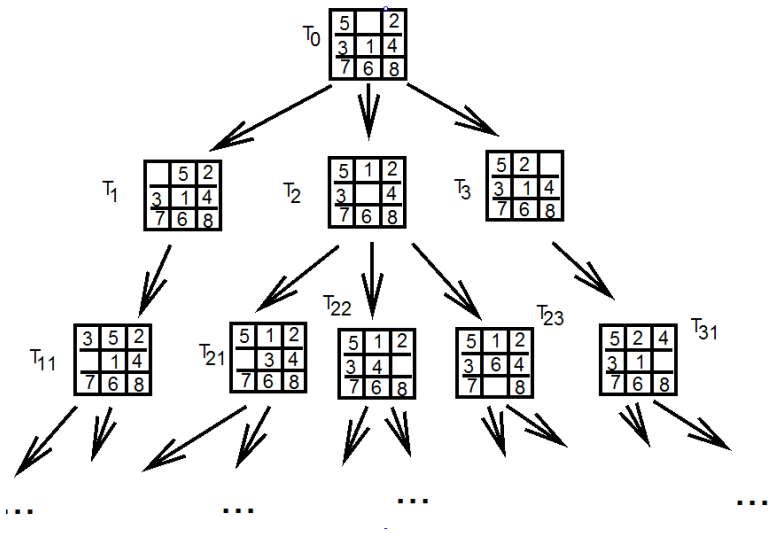


Fig. 8 : Début de l'arbre des états du taquin partant d'une configuration initiale

Et en continuant ainsi de suite pour tous les T_i, on construit les niveaux successifs : en dessous de T_1, on trouve T_{11}, T_{12}, ..., en dessous de T_2, on trouve T_{21}, T_{22}, ..., en évitant de mettre plusieurs fois la même configuration.

Chaque T_i, à l'exception de T_0, a un prédécesseur et éventuellement, mais pas nécessairement, des successeurs. Les configurations sans successeur sont appelées les feuilles.

Une fois l'arbre terminé, il suffit de rechercher s'il existe une feuille correspondant à un taquin rangé. Si c'est le cas, la résolution consiste, partant de cette feuille de remonter successivement vers l'unique prédécesseur possible jusqu'à revenir à la configuration de laquelle on était parti.

En pratique, dans les problèmes et jeux intéressants, l'arbre complet est en général bien trop imposant pour être construit complètement. Il faut alors ruser. Le but est de construire un arbre plus petit en ne construisant que certaines branches plus « prometteuses » que d'autres. Comment faire cela ? Il existe bien des manières d'attaquer un tel problème, mais nous allons décrire une démarche assez générale pour être utilisée dans un nombre important de situations, à savoir l'algorithme "Branch and Bound".

L'ALGORITHME "BRANCH AND BOUND"

Comme nous venons de le dire, construire l'arbre contenant toutes les configurations partant de T_0 est trop long, la stratégie proposée consiste à ne construire qu'un sous-arbre. Celui-ci dépend d'une fonction "Score" permettant d'évaluer la qualité, la valeur d'une configuration. Nous décrivons plus loin dans le texte la fonction que nous avons choisie et la manière dont elle est calculée, mais celle-ci n'est pas unique. Cette démarche est nommée l'algorithme "Branch and Bound". La configuration voulue, dans notre cas le taquin rangé, aura la valeur maximale. Ainsi, si on cherche les directions dans lesquelles la valeur de la fonction Score augmente, on espère que l'on atteindra rapidement le maximum. Pratiquement, partant d'une configuration T_0, on évalue tout d'abord Score(T_0). On construit ensuite l'ensemble des descendants directs (T_1, T_2, ...) de T_0 et on calcule pour chacun d'eux la valeur de la fonction Score, etc. (nommés score(T_i) dans Fig. 9 et Fig. 10).

On choisit alors la feuille ayant le meilleur Score. Ce n'est que pour cette feuille que l'on va créer les descendants. On reprend alors toutes les feuilles, et pas seulement les dernières construites et on choisit à nouveau la meilleure feuille relativement à la fonction Score, pour en prendre ses descendants, puis on réitère le processus. Attention, selon la terminologie définie précédemment une feuille dont on a créé les descendants n'est plus une feuille.

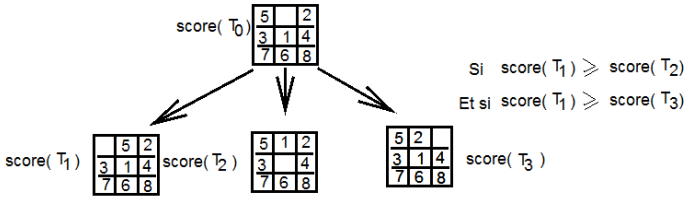


Fig. 9 : Premier pas de l'algorithme Branch and Bound

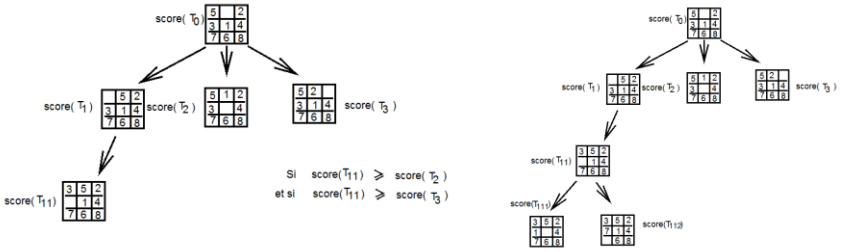


Fig. 10 : Deuxième et troisième pas de l'algorithme Branch and Bound

Ainsi on ne construit l'arbre enraciné que dans les directions où l'on détecte le meilleur Score. En pratique on finit par arriver à des extrema locaux. Si le principe de l'algorithme est simple, le codage est plus délicat. En effet, il s'agit de stocker d'une manière ou d'une autre une telle structure d'arbre enraciné et étant capable de dire si tel ou tel sommet est une feuille ou non. De plus dans la construction des étapes successives du graphe, la feuille ayant le meilleur Score perd son statut de feuille puisqu'on ajoute à l'arbre les successeurs de celle-ci. Si conceptuellement, cette idée n'est pas difficile, les détails techniques peuvent s'avérer pénibles, nous ne nous y attarderons donc pas trop. Les personnes intéressées peuvent aller voir les codes sources écrits en Scilab sur le site de RMé. Faisons néanmoins quelques remarques concernant ce programme, celles-ci ont chacune un intérêt plus général sur ce qui est important en informatique.

QUELQUES REMARQUES À PROPOS DU PROGRAMME

La première s'intéresse à la manière de coder un taquin. L'informatique est la science du codage et de la manipulation automatique d'informations. La manière dont sont stockées les données a donc une importance capitale. Un point important pour la performance de cet algorithme est de pouvoir évaluer le Score d'une configuration assez rapidement, ce qui dépend de la manière dont l'information est stockée. Un taquin peut être codé de plusieurs manières. Regardons-en trois. Notons au lieu du vide la valeur 9. Ce choix de 9, plutôt que 0, est utile pour la construction de la fonction Score et sera expliqué plus loin. Il est clair qu'il n'y pas qu'une seule manière de coder un tel type d'information, mais il faut absolument être cohérent avec ses propres choix.

4	9	3
6	1	7
2	8	5

Fig. 11 : Exemple de taquin

Le taquin ci-dessus ressemble à une matrice 3 fois 3, il peut donc se coder par une matrice, mais peut tout aussi bien se coder comme le vecteur ligne à 9 composantes (4;9;3;6;1;7;2;8;5), ou plus simplement encore par le nombre 493617285. Bien que les deux dernières écritures semblent être identiques, la manipulation informatique d'un nombre ou d'une suite de nombres n'est pas identique. Ces trois

méthodes (matricielle, vectorielle ou numérique) ont chacune des avantages et des inconvénients. Il est donc bon d'utiliser l'une ou l'autre suivant ses besoins. En outre, il est possible de construire de manière performante des fonctions NtoV, VtoM, MtoV et VtoN passant d'une forme à l'autre, dont les noms explicitent le rôle. On peut donc choisir de changer de codage en fonction de la tâche à effectuer sans que cela ne coûte trop en termes de temps de calcul. Bien sûr, choisir comme codage de base celui dans lequel les opérations sont les plus nombreuses optimalise le code en termes de performance de calcul et permet ainsi de gagner un peu de temps. La composition de ces fonctions permet ensuite de passer de n'importe quelle forme de codage d'un taquin à n'importe quelle autre. L'avantage du codage par le nombre est la facilité et la rapidité de comparaison et de manipulation. Le codage vectoriel est utile pour construire la fonction Score décrite plus loin dans le texte. Le codage matriciel est celui qui permet le mieux une visualisation proche du taquin.

On pourrait penser que compter le nombre de pièces en bonne position est une bonne fonction Score. Cependant, la configuration 123459786 est beaucoup plus désirable que 213456789, car on ne sait pas si cette dernière peut ou non être réordonnée, alors qu'il est évident que la configuration 123459786 peut l'être.

1	2	3
4	5	
7	8	6

Fig. 12 : 123459786

2	1	3
4	5	6
7	8	

213456789

Or ces deux configurations ont le même nombre de nombres bien placés, si on considère le vide comme la case à placer en bas à droite. Il semble donc que la configuration avec les nombres bien placés en haut à gauche du tableau soit plus désirable que celle en bas à droite. On choisit donc une fonction

Score: $(v_1; v_2; v_3; \dots; v_9) \rightarrow \sum_{i=1}^{9} \delta(i, v_i) 10^{9-i}$ où $(i, v_i) = \begin{cases} 1 \; si \; i = \; v_i \\ 0 \; sinon \end{cases}$.

L'intérêt de cette fonction Score est qu'elle pondère davantage les nombres bien placés dans le haut gauche du taquin que ceux bien placés en bas à droite. La valeur de la fonction Score correspondant à un taquin rangé est donc 111111111, mais si on choisit cette valeur comme condition d'arrêt, on se rend compte que parfois le programme ne semble pas pouvoir l'atteindre. Par contre si on choisit une condition d'arrêt égale à 111111000, on constate empiriquement que l'algorithme se termine toujours. Ceci donne une bonne condition d'arrêt laissant juste les nombres 7 et 8 et la case vide sur la dernière ligne. Et on remarque que parmi les 6 configurations correspondantes 789, 798, 879, 897, 978 et 987, comme le nombre 9 correspond à la case vide, il est toujours possible de le ramener à la fin de la dernière ligne, laissant deux seules possibilités pour cette dernière qui sont

7	8	9

ou

8	7	9

La première correspond au taquin rangé et il a été montré empiriquement et théoriquement, que la seconde ne peut pas aboutir à un taquin rangé. C'est sans doute ce qu'avait constaté Sam Loyd, qui lui permettait d'offrir un dollar à la personne qui aurait su ranger un taquin aboutissant à cette configuration.

Il est clair que le choix de la fonction n'est pas unique et peut avoir un impact sur l'efficacité de l'algorithme, elle doit donc être choisie avec soin. En effet, dans ce genre de démarche, c'est la construction d'une fonction Score efficace qui est le cœur de l'algorithme, mais en est aussi la partie difficile. Coder un jeu ou la résolution d'un puzzle comme le taquin via un graphe amène à devoir

trouver un chemin permettant de passer d'un sommet à un autre. Dans un graphe, ceci n'est pas toujours simple, mais dans un arbre orienté ayant une racine c'est plus facile. Il suffit de garder la trace de l'unique prédécesseur (le père) de chaque configuration.

On peut donc essayer de répondre à notre interrogation initiale : si on tire au hasard les nombres dans un taquin, est-il toujours possible de les ramener sur le taquin rempli correctement, c'est-à-dire d'arriver sur la configuration d'un taquin rangé ? C'est la question à laquelle se sont attaqués les lycéens.

QU'ONT FAIT LES LYCÉENS ?

Un travail MATH.en.JEANS se déroule en général comme suit : un chercheur propose à des lycéens une question ouverte à leur niveau. En petit groupe et sous la supervision d'enseignants de leur établissement, ces derniers mettent alors librement en place des stratégies pour résoudre la question posée. Ce travail peut durer un peu moins d'une année scolaire, les lycéens se retrouvent de manière hebdomadaire avec leurs enseignants et rencontrent le chercheur entre deux et trois fois dans l'année. Ils doivent ensuite rédiger un article qui sera présenté à un congrès MATH.en.JEANS annuel en face d'autres groupes de lycéens. C'est dans ce cadre que nous avons proposé cette question.

Dans notre rôle de chercheurs, nous avons rencontré pour la première fois les lycéens dans leur établissement en automne 2016, c'est à ce moment que nous leur avons proposé des sujets de recherche. Les lycéens travaillaient par groupe de deux ou trois sur divers sujets, le taquin avait été choisi par deux des groupes. Les enseignants du lycée avaient un contact hebdomadaire avec les élèves, l'interaction entre les chercheurs et les lycéens s'est plutôt construite via des échanges email. Les chercheurs ont néanmoins rencontré le groupe à deux autres reprises et l'un des chercheurs s'est rendu au congrès MATHS.en.JEANS au printemps suivant. Ce projet a donc duré 6 mois. Durant les deux premiers mois, les lycéens ont mis au point des stratégies de résolutions du taquin, mais celles-ci étaient trop lentes au niveau de l'exécution pour permettre des résultats statistiques, nous avons donc rediscuté avec eux d'un algorithme de résolution optimisé du type Branch and Bound qu'ils ont implémenté suivant nos conseils. Ils ont ainsi pu faire un grand nombre de résolutions de taquins.

En tirant au hasard une configuration initiale du taquin et en essayant de la résoudre, les lycéens ont remarqué que la résolution était possible environ une fois sur deux et semblait impossible dans le reste des cas. Ils en ont déduit qu'il devait être possible de séparer les configurations initiales en deux parties de même taille, l'une contenant les configurations dont la résolution est possible, l'autre celles amenant à une impossibilité. À ce stade, ceci était une conjecture et non une preuve. En se renseignant sur le sujet, les élèves ont réalisé qu'il était possible de démontrer cette conjecture par des moyens théoriques. Toutefois faire développer une telle preuve par les élèves n'était pas le but initial du travail proposé, mais cela aurait pu être une démarche dans un deuxième temps. Malheureusement le temps manquait. Notre but dans ce cadre était de donner aux lycéens des outils de prospectives statistiques pour se faire une idée de la réponse à la question posée par Sam Loyd, à savoir s'il est possible de passer de la configuration 123456879 à la configuration 123456789. La réponse semblait être négative et les lycéens en ont eu confirmation en trouvant des arguments théoriques expliquant cette impossibilité.

Pour les personnes intéressées à voir comment ce résultat se démontre, voir Coste (2011).

Les dates indiquées montrent que ce travail a été entrepris il y a quelques années et n'était pas conçu en vue d'une publication. Ainsi nous n'avons récolté que peu d'informations concernant la pratique des élèves et nous n'avons pas consigné les interactions entre les groupes de lycéens, ni entre les groupes de lycéens et les chercheurs. Cela explique pourquoi la description du travail en classe est si succincte. Nous n'avons simplement pas récolté de données à ce niveau, car cela n'était pas notre propos. L'idée de cet article est venue plus tard quand nous avons appris la parution de ce numéro spécial de RMé s'intéressant aux jeux.

CONCLUSION

Ainsi le taquin n'est pas toujours résoluble. La preuve classique utilisant la théorie des groupes fut pendant longtemps l'idée-même de ce que voulait dire "faire des mathématiques", mais la puissance calculatoire toujours plus grande des ordinateurs a un peu changé la donne, alors qu'il y a encore quelques décennies, une preuve d'existence abstraite de solutions se suffisait souvent à elle-même. Actuellement des résultats constructifs sont recherchés bien plus activement, puisque qu'avec une puissance de calcul de plus en plus importante, une méthode constructive, même gourmande en calcul et inapplicable manuellement devient efficace pour obtenir, ou au moins approximer, la solution cherchée. Mais revenons au taquin : Sam Loyd pouvait sans risque proposer un dollar, somme importante à son époque, pour toute personne réussissant son défi, puisque ce dernier était impossible. Mais de telles questions piquent notre curiosité et celle-ci nous emmène souvent dans des mondes insoupçonnés. En ce sens, Sam Loyd nous a offert, avec ses puzzles, des heures de défis qui valent bien sa renommée.

S'il est vrai que les jeux mathématiques sont un plaisir en soi, nous sommes convaincus que ce plaisir peut se trouver à plusieurs niveaux. Bien évidemment, la joie de réussir le défi est le premier de ces niveaux, mais réussir à construire un programme informatique permettant de le faire directement, apporte une énorme satisfaction aux élèves. La joie de la réussite est d'autant plus grande que l'algorithmique, et surtout le codage, peut être parfois frustrante pour l'élève qui débute. Offrir à des lycéens la possibilité de chercher, de conjecturer et de parfois démontrer la véracité d'un résultat donne à ceux-ci le goût pour ce type d'activité.

BIBLIOGRAPHIE

De Vos, M. &. Kent, D. A. (2016). Game Theory, a playful introduction. *Student Mathematical Library Vol 20*, AMS.

Webb, J. N. (2007). *Game Theory, Decisions, Interaction and evolution*. Springer Undegraduate Mathematics Series, Springer.

Coste, M. (2011). *Le jeu de taquin, du côté de chez Galois*. 4 novembre 2011. http://images.math.cnrs.fr/Le-jeu-de-taquin-du-cote-de-chez-Galois.html.

Loyd, S. (1898). *Get of the earth.* http://www.murderousmaths.co.uk/games/getofftheearth.htm. (consulté janvier 2019).