ÉTUDE MATHÉMATIQUE
MATHÉMATIQUES ET INFORMATIQUE

LES PROMENADES D'ELTON LE KANGOUROU

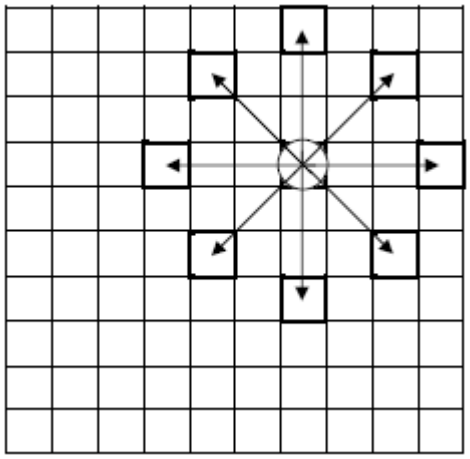
La première partie de cet article permet de nous remémorer les avatars d'un problème qui a occupé quelques numéros du Petit Vert de 1989 à 1999. Dans la seconde partie, Marie Duflot-Kremer (maitre de conférences à l'Université de Lorraine) nous propose une solution informatique à cet intéressant problème.

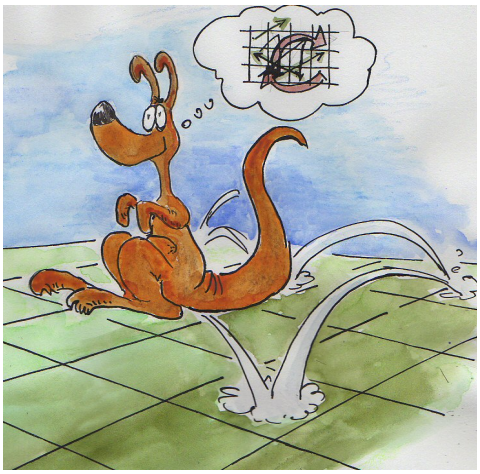
L'énoncé du problème dans le Petit Vert de décembre 1989

Au zoo de Raon-l'Étape vit l'ami ELTON[2], un kangourou solitaire enfermé dans un enclos de 10 m sur 10 m (soit 100 m²). Pour tromper son ennui, il saute de case en case,

- soit de 3 m vers le nord, l'est, le sud ou l'ouest,
- soit de $2\sqrt{2}$ m vers le nord-est, le sud-est, le sud-ouest ou le nord-ouest.

Inlassablement, en 100 sauts, il revient à la case de départ en ayant piétiné toutes les cases.





Première partie : HISTORIQUE

François Drouin raconte :

« J'ai découvert ce kangourou lors d'un stage Jeux animé en particulier par Claude PAGANO, Pierre DORIDANT, Marie-José BALIVIERA et Odile BACKSCHEIDER dans les années 1980. Je l'ai introduit au Club maths du collège, puis en classe de sixième, puis « en famille ».
Nous nous étions limités à la recherche d'un parcours passant par toutes les cases sans nécessairement revenir au point de départ, ce qui aurait représenté une difficulté supplémentaire. Plutôt que regarder sauter l'« ami Elton », nous utilisions simplement le nombre de carreaux dans le carré : « Euh ! l'aire ! ».

De nombreux trajets n'arrivaient qu'à des cases de numéro inférieur à 24. Comme nous l'avait expliqué Claude Pagano, nous regardions si on pouvait faire « reculer le kangourou » pour obtenir un trajet passant par un nombre supérieur de cases. Les cases sont nomérotées de 0 (case de départ) à 24. L'exemple ci-contre montre comment gagner une case :

	17		11	18
4	1	8	5	2
15	12	19	15	13
9	6	3	10	7
20	0	14	21	

	17		11	18
4	1	8	5	2
15	12	19	15	13
9	6	3	10	7
20	0	14	21	-1

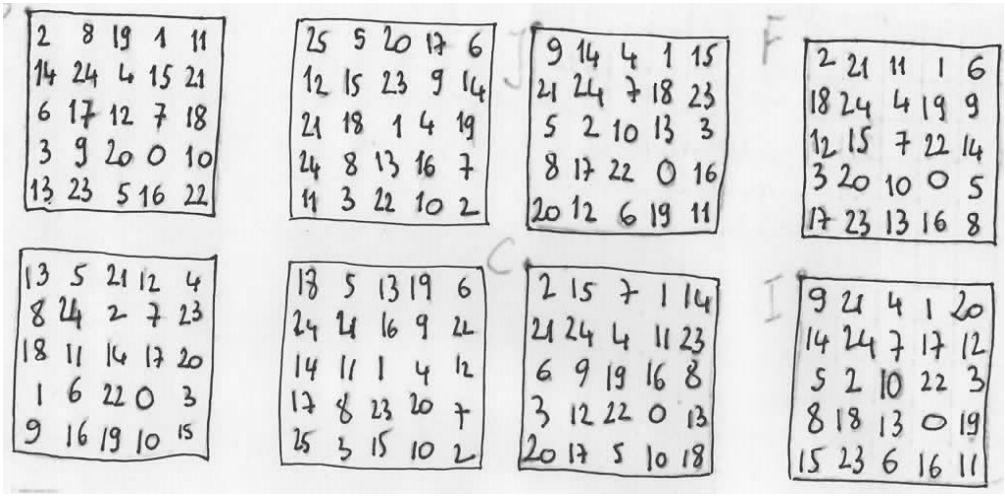
	18		12	19
5	2	9	6	3
16	13	20	16	14
10	7	4	11	8
21	1	15	22	0

[2] Il s'agit bien entendu d'une allusion aux chemins hamiltoniens pris par ce nouvel ami Elton !

retour au sommaire

Finalement, nous avions récolté un certain nombre de grilles dans lesquelles toutes les cases avaient été parcourues par le kangourou ».

Voici quelques exemples de ces premiers essais, proposés par Claude PAGANO :



Première solution informatique (Petit Vert 21 de mars 1990)

Nous avons reçu une solution (partielle) de François Munier (de Lusse, Vosges). Il faisait partir l'ami Elton du coin nord-ouest de son enclos. Pas la peine de se fatiguer... c'est l'ordinateur qui fait le travail. Le programme était écrit en Pascal.

		e_4			S_3				
									e_3
S_4									
									S_2
e_1									
				S_1			e_2		

92	97	76	91	98	75	58	68	74	57
78	89	94	79	88	66	61	71	54	62
84	81	99	85	82	69	52	64	59	51
93	96	77	90	95	72	55	67	73	56
100	86	83	80	87	65	60	70	53	63
23	13	7	20	12	29	47	44	41	50
16	4	10	15	5	34	39	27	33	36
1	19	24	2	8	45	42	30	46	43
22	14	6	21	11	28	48	35	40	49
17	3	9	18	25	31	38	26	32	37

Il n'y a aucune solution pour un carré 4x4 (réponse immédiate de l'ordinateur).
Pour un carré 5x5, l'ordinateur donne toutes les solutions en quelques secondes.
Pour un carré 6x6, la première réponse arrive après plusieurs heures de travail (on est en 1999, il y a eu des progrès depuis !) … inutile, par conséquent, d'essayer avec le carré 10x10 !

Il faut donc aider la machine : par exemple en partageant le carré initial en quatre carrés 5x5 avec des points d'entrée et de sortie bien choisis, comme montré ci-dessus à gauche. Ci-dessus à droite, une des solutions obtenues avec cette méthode.

Un kangourou qui n'a pas fini de sauter

Bien qu'il ait désormais une solution, le problème continue à intéresser les élèves (qui, eux, n'ont à leur disposition que du papier, un crayon et une gomme). Aussi, près de 8 ans plus tard, dans le n°55 de septembre 1988 du Petit Vert, on pouvait lire ce qui suit.

retour au sommaire

Voici la solution trouvée par un élève, Mickaël Misler[3], du collège de Hombourg-Haut (Moselle), en utilisant deux symétries orthogonales (la case de départ porte le numéro 0) : 25 est le symétrique de 24 ; 26 est le symétrique de 23 ; 27 est le symétrique de 22 ; etc. Puis 50 est le symétrique de 49 ; 51 est le symétrique de 48 ; 52 est le symétrique de 47 ; etc. Et enfin 75 est le symétrique de 74 ; 76 est le symétrique de 73 ; 77 est le symétrique de 72 ; etc.

Remarque : il n'est pas aisé qu'Elton puisse sauter sur toutes les cases d'un enclos de 5 m sur 5 m. Le retour possible à la case départ (ici de 99 à 0) est un « plus » que cet élève a réussi pour l'enclos de 10 m sur 10 m.

	51		74			75			
	50							99	
					13	8	5	19	11
27	49				3	21	15	0	22
					6	18	12	7	17
					14	9	4	20	10
26	48		25		2	24	16	1	23

Notre kangourou Elton est parti également faire quelques bonds à Saint-Mihiel (Meuse) : le pauvre n'a droit, au départ, qu'à un enclos de 5 m sur 5 m, qui s'agrandit peu à peu, devenant un enclos de 6 m sur 6 m, puis de 7 m sur 7 m... pour espérer lui rendre finalement son espace vital de 10 m sur 10 m.

À partir de solutions pour les enclos de 5 m sur 5 m et 6 m sur 6 m, les élèves meusiens ont construit des solutions pour des enclos de 10 m sur 10 m, de 12 m sur 12 m, de 18 m sur 18 m, etc.
Pour les enclos de 7 m sur 7 m ou de 8 m sur 8 m, il ne leur a pas été facile de trouver des réponses qui permettaient le retour à la case départ. Et pour l'enclos de 9 m sur 9 m, personne ne réussissait à trouver de réponse.

Un autre idée est venue aux élèves : Elton pouvait-il se déplacer dans un enclos rectangulaire non carré ? La solution déjà trouvée pour 12 m sur 18 m était une première réponse à cette question. Et l'on s'est intéressé aux enclos de 4 m sur 5 m, de 5 m sur 6 m, etc.

Mais on ne savait toujours pas si un enclos de 9 m sur 9 m pouvait accueillir notre kangourou !

Cet "angoissant" problème a donc été présenté (à Maxéville) aux professeurs stagiaires de deuxième année de l'I.U.F.M. Et voici la solution que l'un d'eux a trouvée. Elle utilise des rotations de 90° à partir d'une des solutions proposées pour l'enclos de 4 m sur 5 m. Malheureusement, une fois arrivé à la case 80 (la 81ème et dernière de son périple), notre pauvre Elton ne pouvait plus retourner à son point de départ.

				20	3	6	9	2
				14	11	18	15	12
		21		5	8	1	4	7
				19	16	13	10	17
40				0				80
		41			61			
			60					

Cependant, il pouvait retourner à la case 1 et continuer inlassablement son circuit, ignorant la case centrale "0".

Parmi les lecteurs du Petit Vert, amis des bêtes (et des jeux mathématiques), s'en trouverait-il un qui réussirait à faire revenir le kangourou à son point de départ dans un enclos de 9 sur 9 m ???

Le zoo de Raon-l'Étape pourrait alors lui fournir un enclos de remplacement !

[3] Élève de Martine Dechoux.

retour au sommaire

Le retour de l'ami Elton… dans le Petit Vert n°57 (mars-avril 1999)

Une fois bien reposé, il additionne les numéros de chaque ligne, de chaque colonne et de chaque diagonale, et il trouve (quelle chance !) toujours le même résultat. Saurez-vous retrouver le circuit (hamiltonien) de l'ami Elton ? Y a-t-il d'autres circuits ayant cette propriété ?

La réponse aux questions précédentes a été trouvée par un certain Jean-Marie (dont nous ignorons le nom). La voici, ci-contre à droite.

5	14	29	4	13	30	3	12	31
61	41	26	62	40	25	63	39	24
28	71	48	45	72	69	46	73	2
6	15	60	78	66	57	79	11	32
49	42	27	70	47	74	64	38	23
19	77	67	44	53	68	35	56	1
7	16	59	75	65	58	80	10	33
50	43	20	51	36	21	52	37	22
18	76	8	17	54	9	34	55	0

8	76	50	7	81	51	24	71	52	25
36	29	10	37	28	11	38	27	12	39
49	6	80	75	60	88	82	61	23	70
9	77	92	97	78	85	94	72	53	26
35	30	65	87	90	74	59	89	13	40
48	5	79	84	93	98	83	62	22	69
18	45	91	96	66	86	95	73	54	1
34	31	64	57	32	63	58	99	14	41
47	4	19	46	3	20	67	2	21	68
17	44	33	16	43	56	15	42	54	0

Ce même Jean-Marie proposait également une solution correspondant à l'enclos de 10 m sur 10 m qui n'utilisait pas la décomposition de l'enclos en quatre carrés de 5x5 (voir ci-contre à gauche).

Il y avait alors une remarque : *à suivre dans un prochain numéro*...
Était-ce un poisson d'avril ?

Depuis cette date, on croyait avoir perdu la trace d'Elton… Il n'était plus dans son enclos lorrain : il avait fait un gigantesque bond pour se retrouver dans la brochure « Jeux 6 » éditée par l'APMEP !

Quelques années plus tard, en 2017, Elton fait son « come back » avec la solution du problème, grâce aux progrès de l'informatique.

Seconde partie : RÉSOLUTION INFORMATIQUE DU PROBLÈME

Le backtracking ou retour sur trace

Même si le terme "retour sur trace" est connu par Wikipedia, la méthode utilisée pour résoudre le problème du kangourou est connue des informaticiens sous ne nom de "backtracking". Cette méthode consiste à essayer de construire une solution progressivement (ici choisir les sauts consécutifs du kangourou) en respectant quelques règles de base :
— faire un des 8 sauts possibles,
— ne pas retourner sur une case déjà visitée,
— ne pas sortir du cadre,
— dire qu'on a gagné si toutes les cases sont visitées et on peut revenir en un saut au départ.

Seulement, quand on enchaine des sauts choisis arbitrairement, on a toutes les chances de se retrouver dans une situation où on ne peut pas continuer (toutes les cases n'ont pas été visitées mais il n'y a pas moyen de faire un saut de plus dans une case non visitée) alors on annule le dernier saut et on en essaie un autre. Et si aucun saut ne convient, on revient une étape en arrière de plus, et ainsi de suite. L'inconvénient d'une telle méthode est que, dans le pire des cas, on peut visiter tous les chemins possibles, et ce nombre de chemins est très grand. On peut bien sûr essayer de l'optimiser. Par exemple si toutes les cases qui permettent de retourner au départ sont visitées alors qu'on n'a pas tout parcouru, on ne pourra plus revenir au départ et c'est perdu, pas besoin de continuer. De plus, pour des raisons de symétrie on peut éliminer certains chemins.

retour au sommaire

Quelle que soit la méthode utilisée, on se doute bien que, si on augmente la taille du terrain du kangourou, le temps pris par un algorithme de recherche de chemin avec retour sur trace va exploser.
Même si on supposait, ce qui est super restrictif, qu'on n'a que deux choix pour continuer à chaque étape (au lieu des 8 maximum théoriquement possibles) et que tous les chemins (sauf le bon) s'arrêtent car il est impossible de continuer après avoir rempli un tiers de la grille, on a tout de même $2^{n2/3}$ possibilités, ce qui pour un carré de 10 sur 10 fait 2^{33} soit quelques milliards. Mais si on passe à un carré de taille 20 cela donne déjà 2^{133} soit de l'ordre de 10^{30} soit plus ou moins 10^{13} années en testant un milliard de chemins par seconde.
Ce qui fait qu'on s'en sort c'est que, s'il existe une solution, justement on ne va pas tester tous ces chemins. Dès qu'on trouve la bonne solution on va s'arrêter. Mais par contre s'il n'y a pas de solution ou si on explore les chemins en terminant par le/les bons, on peut avoir un algorithme qui prend énormément de temps. On se retrouve donc à lancer un algorithme sans aucune certitude et sachant qu'il peut tourner pendant des heures/jours ou plus.

Algorithmes directs

Quelques années après les premières tentatives d'aider Elton, les ordinateurs ont évolué et peuvent plus rapidement donner une solution. Mais comme on a vu dans la section précédente, même une division par mille du temps de calcul d'un ordinateur ne nous permet pas d'aller beaucoup plus loin dans la taille du champ. La complexité du problème joue contre nous, et c'est plus la chance (ou l'existence de nombreuses solutions) qui peut nous aider.
Pour tester les petites instances du problème, nous avons tout d'abord réalisé un programme naïf, en Python, implantant un algorithme simple de backtracking.

L'algorithme, présenté ci-dessous, est assez simple. Il garde en mémoire trois choses : le chemin qu'on a suivi jusqu'à présent (qui au départ contient juste la case du haut à gauche, souvent numérotée 0,0 en informatique), les cases déjà visitées (au départ juste celle du haut à gauche) et le nombre de cases encore non visitées (reste).
Puis on écrit une fonction informatique **cherche (chemin, reste)**. Cette fonction va, étant donnés un début de chemin (le paramètre "chemin") et un nombre de cases non encore visitées (le paramètre "reste") essayer de compléter le chemin existant pour en faire une solution. Cette fonction utilise une autre fonction toute simple : **fin(chemin)** qui renvoie la dernière étape d'un chemin (là où il s'arrête pour le moment).

```
Fonction cherche (chemin, reste)
    Si reste=0 et je peux revenir en un saut de fin(chemin) à départ Alors
        Arrêter tout, on a gagné!!!
    Sinon
        Pour chacun des 8 sauts possibles Faire
            Si en partant de fin(chemin) et en faisant ce saut on arrive sur une case c non visitée
            (et dans le cadre) Alors
                noter que la case c est visitée
                ajouter la case c à la fin de chemin
                appeler cherche(chemin, reste-1)
                # les deux lignes suivantes ne sont faites que si la case c n'a pas
                  donné un parcours correct. C'est donc le retour sur trace.
                enlever la case c de chemin
                noter que la case c est non visitée
            Finsi
        Finpour
    Finsi
Finfonction
```

Avec cette fonction on va bien tester les chemins possibles, en ajoutant une étape au chemin puis en regardant si on peut continuer et finir, et si vraiment cette étape ne mène à rien on l'enlève et on en teste une autre.

retour au sommaire

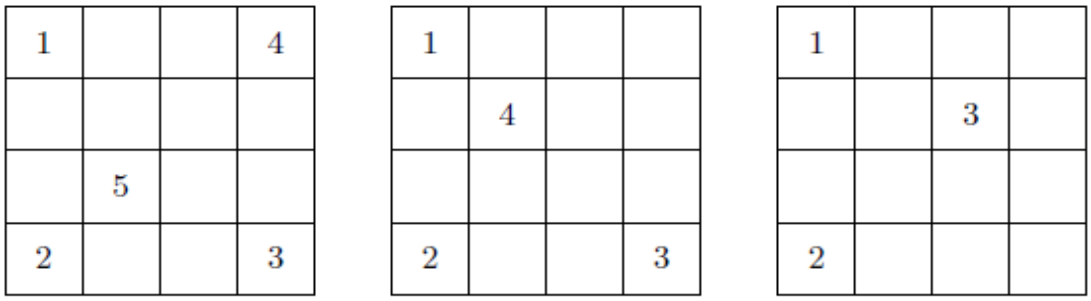


Figure 1 – Trois étapes de la recherche (infructueuse) de parcours dans un champ de 4 par 4.

Pour l'exemple d'un carré de 4 sur 4 (pour lequel il n'y a pas de solution) on va à partir du point de départ, tester le saut en bas (ce qui nous amène dans la case (0,3) non visitée. De là on teste le saut en bas (hors du cadre), puis en haut (déjà visité) puis à droite (ça marche) et on est dans la case (3,3).
De là on teste en bas (hors du cadre), puis en haut ce qui nous amène dans la case (3,0). de cette case seul le saut en bas à gauche est possible ce qui nous donne le premier dessin de la Figure 1. Seulement maintenant on va tester tous les sauts à partir de cette case (numérotée 5 sur la figure) mais ils sont tous soit hors du cadre soit déjà visités. On va donc enlever cette étape, voir si à partir de 4 on peut faire autre chose, voir que non, effacer le 4 de notre grille et voir ce qu'on peut faire à partir du numéro 3 ce qui nous donne notre deuxième dessin, voir qu'on est encore bloqué, revenir encore d'une étape en arrière et essayer (troisième dessin) et ainsi de suite.
Avec notre programme et en tournant sur un ordinateur portable moderne (et donc pas une grosse station de travail ou un super calculateur) on trouve les solutions pour un carré de 5 x 5 en une demi-seconde, de 6 x 6 en 7 secondes, de 7 x 7 en 6 secondes (comme quoi même s'il y a plus de chemins la chance peut jouer), de 8 x 8 en un quart de seconde, mais pour celui de 9 x 9 l'ordinateur ne donne pas de réponse avant une interruption du programme au bout de plusieurs heures.

Quelques améliorations

Afin d'augmenter nos chances de résoudre directement de plus grandes instances du problème, il fallait utiliser des techniques/outils moins rudimentaires qu'un programme avec backtracking sans trop d'optimisation. Pour cela nous avons utilisé un outil appelé "solveur de contraintes", qui va prendre en entrée un ensemble d'(in)équations représentant le problème, les simplifier au maximum avant de tenter de trouver une solution qui satisfasse toutes ces (in)équations.
Pour décrire le problème du parcours du Kangourou sous la forme acceptée par l'outil, nous avons d'abord énoncé qu'il fallait construire une liste de positions sur le carré, que cette liste de positions devait être de longueur n x n et être un circuit (donc ne passant pas deux fois par la même case, et permettant de revenir au départ).
Il a ensuite fallu dire quelles contraintes reliaient une position (X0,Y0) à la suivante (X,Y) dans le circuit. Les contraintes ressemblent à :
— X = X0 + DX
— Y = Y0 + DY
— DX et DY appartiennent à l'ensemble {-3 ; -2 ; 0 ; 2 ; 3}
— abs(DX) + abs(DY) ≥ 3
— abs(DX) + abs(DY) ≤ 4
Les deux dernières contraintes portant sur les valeurs absolues permettent de ne laisser que les huit sauts autorisés. Le solveur de contraintes a ensuite été lancé sur cette description du problème.
Les techniques de simplification de systèmes de contraintes permettent d'éviter de tester bon nombre de chemins qu'un simple algorithme de backtracking mettrait des heures, des jours, voire des années à parcourir.
Avec cette méthode, nous avons réussi à trouver des solutions directes pour les carrés de côté 7 à 11, ainsi que pour certaines valeurs (mais pas toutes) jusqu'au carré 20x20. Comme pour le backtracking, la méthode utilisant la résolution de contraintes finit par tester des valeurs encore possibles après simplification et est du coup très sensible à la "chance" : on peut pour une plus grande instance tomber plus rapidement sur une solution, par hasard, que pour une petite instance où on finit par se lasser et abandonner.

retour au sommaire

Utiliser les petites instances pour résoudre les grandes

Comme on sait que, même avec des méthodes optimisées et adaptées, résoudre directement des grands champs pour le kangourou n'est pas gérable, on va essayer de ruser.
Une première technique qui avait été envisagée et déjà publiée dans le Petit Vert était pour la première instance qui ne tourne pas sur ordinateur, de décomposer le carré de 9 sur 9 en 4 rectangles de 4 sur 5 plus une case au milieu. L'idée étant, en partant de la case centrale, de sauter dans un des rectangles, puis de le visiter complètement en terminant sur une case près d'un bord à partir de laquelle on pourrait sauter au rectangle suivant, que l'on visiterait, etc. pour finir après avoir visité les 4 rectangles par ressauter dans la case centrale. Sans ordinateur, un lecteur avait trouvé une solution mais qui ne pouvait pas retourner dans la case centrale. Il y avait un cycle qui parcourait toutes les cases sauf la centrale.
Avec cette idée et un ordinateur, il a été possible (en modifiant légèrement le programme de base pour ne plus terminer dans la case de départ mais dans une case choisie) de trouver un chemin qui visite tout le carré de 9 sur 9, visible dans la figure 2 ci-dessous.

32	22	37	27	21	4	7	10	3
39	25	34	40	15	12	19	16	13
36	28	31	23	6	9	2	5	8
33	41	38	26	20	17	14	11	18
30	24	35	29	1	66	76	81	71
51	54	57	60	53	79	73	64	78
45	42	49	46	43	62	70	67	75
56	59	52	55	58	65	77	80	72
50	47	44	61	48	68	74	63	69

Figure 2 : Parcours d'un carré de 9 cases sur 9 en le découpant en 4 dalles de 4x5 cases.

Pour des instances plus grandes (donc à partir de 10), nous avons eu l'idée de réutiliser cette technique de découper un grand champ en plus petits, parcourant un petit puis sautant au suivant. Le résultat mathématique élémentaire que nous avons utilisé est que toute valeur v supérieure où égale à 10 (donc celles qu'on ne sait pas traiter) se décompose en $v=a$x5+bx6+cx7. Sachant cela, tout carré de v sur v va pouvoir être découpé, verticalement et horizontalement en bandes de 5, 6 ou 7 cases de large, formant des dalles rectangulaires de 5 à 7 cases de longueur/largeur. A partir de cela, il "suffit" de

— trouver un parcours entre les différentes dalles de notre carré,

— trouver un parcours du kangourou au sein de chaque dalle en choisissant un point d'entrée et un point de sortie qui permettent de lier cette dalle à la précédente et la suivante dans notre parcours.

Pour le parcours entre les dalles, la solution retenue dépend du nombre de bandes découpées en hauteur/largeur. Si ce nombre de bandes est pair, un cheminement tout simple nous permet de passer d'une dalle à une autre qui partage un côté avec elle comme le montre le premier dessin de la figure 3.

Si le nombre de bandes est impair, on doit ruser un peu. On ne zigzague plus seulement de gauche à droite en remontant les dalles, mais sur les deux lignes du haut on zigzague

retour au sommaire

également et on finit par un saut en diagonale, comme visible dans le deuxième dessin de la figure 3 ci-dessous.

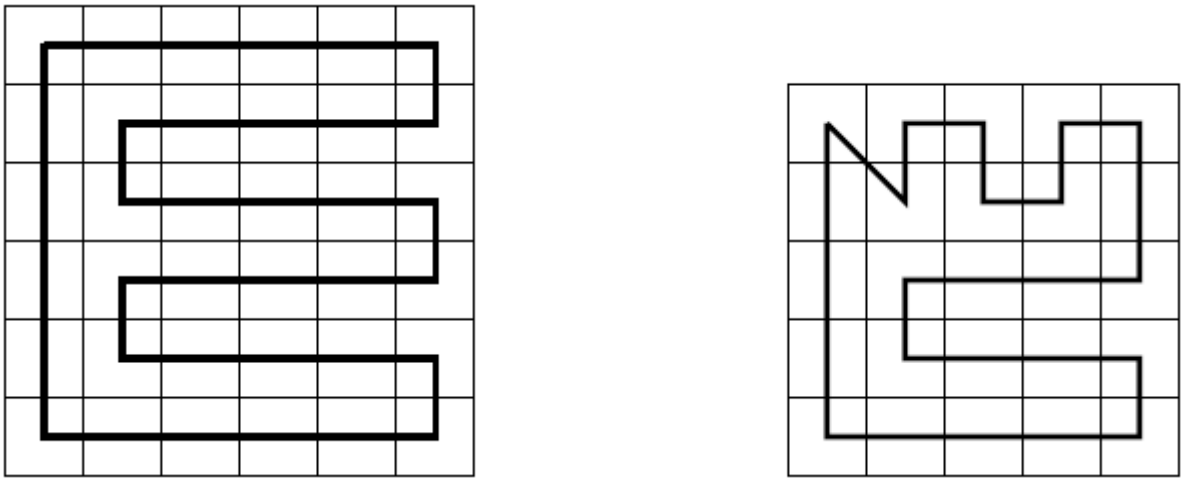


Figure 3 : Parcours entre les dalles d'un champ.
A gauche avec un nombre de rangées pair et à droite quand il est impair.

Vus ces déplacements, on peut déduire quels sont les différents parcours dans les dalles qui nous intéressent. Il y a 5 cas différents, illustrés dans la Figure 4 selon la position de la dalle en cours par rapport à la précédente et la suivante. Elles peuvent être alignées, réaliser un quart de tour à droite, à gauche, ou correspondre au saut final (arriver sur la tuile en diagonale ou aller de la tuile en diagonale à la tuile de départ.

Il suffit donc, pour les différentes tailles possibles de rectangles, de voir si pour chacun des 5 points d'entrée possibles on peut tout visiter en rejoignant à la fin le point de sortie. Nous avons donc fait tourner notre programme naïf pour des rectangles de taille 5 x 5, 5 x 6, 6 x 5, 6 x 6, 6 x 7, 7 x 6 et 7 x 7.

La taille 5 x 7 est inutile, car, comme on travaille sur un carré, on fait le même découpage verticalement et horizontalement. Un rectangle de 5 x 7 signifierait qu'on a une bande de taille 5 et une de taille 7, que l'on peut remplacer par deux bandes de taille 6. Comme il était naïf, notre programme a eu vraiment du mal pour le carré de 7 x 7, pour lequel un des points d'entrée (le premier de la figure 4) a pris énormément de temps. Au total pour les 5 cas il a mis 8 heures et 21 minutes !!! C'est très long, surtout que pendant ce temps là on ne sait pas s'il va donner un résultat dans l'année ou pas.

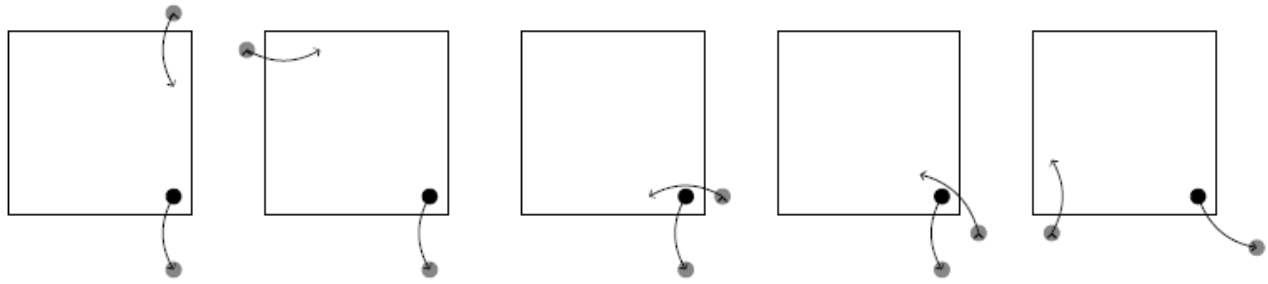


Figure 4. 5 cas de points d'entrée et de sortie dans les dalles en fonction de sa position par rapport aux dalles d'avant/après dans le parcours. De gauche à droite : 1) quand les trois dalles sont alignées 2) quand on fait un quart de tour à droite 3) quand on fait un quart de tour à gauche 4) la dalle de départ quand on y arrive depuis la diagonale 5) la dernière dalle quand on doit rejoindre le départ en diagonale donné dans la Figure 5. Le remplissage a été fait avec un programme dédié permettant, étant donné les remplissages des petites dalles, de remplir tout le carré.

A titre d'exemple, nous avons appliqué cette méthode à un carré de taille 19 x 19 et obtenu le parcours donné dans la figure 5. Le remplissage a été fait avec un programme dédié permettant, étant donné les remplissages des petites dalles, de remplir tout le carré.

retour au sommaire

25	18	8	24	19	7	300	321	311	301	322	310	249	270	285	250	269	286	253
15	3	27	16	4	34	318	306	296	319	307	297	262	277	282	261	276	281	260
29	12	22	32	9	23	292	315	303	293	312	302	284	257	264	279	254	265	274
26	17	5	35	20	6	299	320	308	298	323	309	248	271	288	251	268	287	252
14	2	28	13	1	33	317	305	295	316	304	294	263	278	283	258	275	280	259
30	11	21	31	10	36	291	314	324	290	313	325	289	256	267	272	255	266	273
52	42	58	53	41	65	361	331	336	360	328	337	247	226	215	210	227	216	209
60	70	50	63	69	49	342	347	352	339	344	353	219	231	236	224	230	235	223
46	54	38	66	57	37	335	359	327	332	356	326	238	211	246	233	214	245	228
51	43	59	71	40	64	351	330	343	348	329	338	206	225	218	207	242	217	208
61	67	47	62	68	48	341	346	355	340	345	354	220	232	237	221	229	234	222
45	55	39	44	56	72	334	358	350	333	357	349	239	212	243	240	213	244	241
95	82	87	94	77	114	148	129	115	147	128	156	205	192	157	175	191	158	174
109	92	101	108	89	100	121	141	134	120	140	135	165	183	199	166	182	198	167
86	106	76	83	103	73	152	146	127	155	145	124	186	170	204	185	169	203	190
96	81	88	93	78	113	149	130	116	142	133	119	177	193	160	176	200	159	173
110	91	102	107	90	99	122	138	151	123	139	136	164	184	196	163	181	197	168
85	105	75	84	104	74	153	143	126	154	144	125	187	171	201	188	172	202	189
97	80	111	98	79	112	150	131	117	137	132	118	178	194	161	179	195	162	180

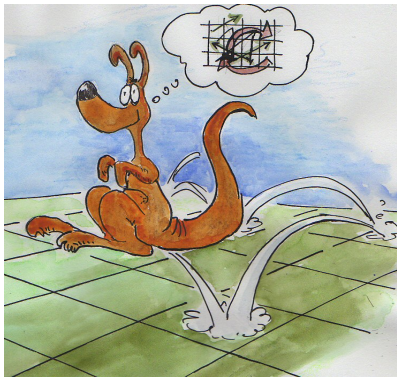
Figure 5 – Parcours d'un carré de 19 cases sur 19 en le découpant en 9 dalles

Conclusion

La morale de cette histoire est que, sur de telles questions, la force brute de l'ordinateur trouve rapidement ses limites quand la taille du problème augmente. Le backtracking est donc une bonne solution pour de "petites" instances (la signification de "petites" dépendant grandement du problème et des optimisations apportées à l'algorithme), mais si on veut en résoudre de plus grandes voire même montrer qu'il existe une solution pour toute instance possible, on a souvent besoin d'utiliser d'autres méthodes de raisonnement, à base de papier, de crayon et de neurones qui chauffent, à combiner avec une solution automatique sur de petites instances.

Avec ce raisonnement en deux étapes : découper le carré en dalles puis recoller les parcours dans chaque dalle, on peut donc rassurer Elton (et lui donner une solution) : oui, il peut visiter toutes les cases sans repasser deux fois par la même, puis revenir à son point de départ, et ce pour tout carré d'au moins 5 cases de côté.

Bonne route à lui !



υ **TÉLÉCHARGER LE PROGRAMME PYTHON**

retour au sommaire