Quand des agneaux rencontrent un loup...

Pas	Code	Libellé
00	86 6	2nd Lbl 6
01	32 0	STO 0
02	33 6	RCL 6
03	86 2	2nd Lbl 2
04	34 1	SUM 1
05	61 0	SBR 0
06	-34 1	Inv SUM 1
07	34 2	SUM 2
08	61 0	SBR 0
09	-34 2	Inv SUM 2
10	34 3	SUM 3
11	61 0	SBR 0
12	-34 3	Inv SUM 3
13	34 4	SUM 4
14	86 0	2nd Lbl 0
15	56	2nd Dsz
16	-61	Inv SBR
17	81	R/S
18	86 4	2nd Lbl 4
19	32 0	STO 0
20	33 5	RCL 5
21	51 2	GTO 2
22	86 7	2nd Lbl 7
23	00	0
24	65	-
25	86 3	2nd Lbl 3
26	33 5	RCL 5
27	51 5	GTO 5
28	86 1	2nd Lbl 1
29	00	0
30	65	-
31	86 9	2nd Lbl 9
32	33 6	RCL 6
33	86 5	2nd Lbl 5
34	85	=
35	34 7	SUM 7
36	33 1	RCL 1
37	66	2nd x = t
38	51 8	GTO 8
39	33 2	RCL 2
40	66	2nd x = t
41	51 8	GTO 8
42	33 3	RCL 3
43	66	2nd x = t
44	51 8	GTO 8
45	33 4	RCL 4
46	66	2nd x = t
47	51 8	GTO 8
48	33 7	RCL 7
49	81	R/S

Le loup et les agneaux
pour TI-57
Auteur Jacques Deconchat

Comment s'y prendre pour programmer un jeu sur un ordinateur de poche ? Voici quelques premières idées que vous pourrez mettre en pratique. Une TI 57 vous suffira.

■ Vous venez d'acheter votre calculatrice de poche. Vous avez lu les documents d'accompagnement, vous l'avez mise en route, vous avez essayé quelques-uns des programmes proposés, vous vous êtes familiarisé avec les instructions élémentaires et avec le langage, LMS (Langage Machine Spécialisé) ou Basic plus ou moins évolué.

Vous êtes donc maintenant dans la disposition d'esprit de la plupart des amateurs de ces petites merveilles de notre époque, et vous avez envie de réaliser, de créer votre propre programme. Bien sûr, si vous êtes « matheux », vous allez immédiatement concocter quelques programmes bien « faits » pour calculer un PGCD, un produit de matrices, ou une intégrale par la méthode de Simpson. Nous allons pourtant supposer que, comme beaucoup de monde, vous êtes légèrement réfractaire aux mathématiques (sans aller jusqu'à l'allergie complète, bien sûr !) et que vous aimeriez imaginer un programme plus distrayant, moins technique ; en bref, vous voulez créer un jeu.

Ne vous lancez pas tout de suite sur la première idée qui vous vient à l'esprit, aussi originale soit-elle : il se trouve que la création d'un jeu n'est pas, et de loin, l'exercice de programmation le plus facile. Il est préférable de faire une approche prudente en commençant par des jeux qui se prêtent assez aisément à une transposition sur calculatrice.

Recherchons donc un jeu connu, qui ne nécessite pas de matériel particulier, et dans lequel les coups à jouer sont faciles à traduire dans une notation compréhensible par la machine. Dans cette catégorie, nous pouvons citer pêle-mêle tous les jeux dont les résultats font intervenir des nombres (jeux de dés, jeu de loto, jeu de roulette, etc.) ou peuvent être commodément assimilés à des nombres (par exemple, les jeux de cartes, en utilisant un codage numérique des cartes). De même, les jeux où la position des pièces pourra être représentée par un couple de nombres (jeux à damiers, comme la bataille navale, les dames, les échecs, Othello, etc.).

Comme il fallait bien se décider, nous avons choisi d'adapter à une calculatrice élémentaire un jeu très connu, appelé « le loup et les agneaux ». Le terrain de jeu est un damier, soit de 8 cases sur 8 cases, (voir fig. 1) soit de 10 x 10, et les règles sont très simples, la stratégie n'étant toutefois pas évidente pour un débutant.

Plaçons-nous dans le cas d'un échiquier avec 1 loup et 4 agneaux. Le loup est placé au départ en (4,h) ou en (6,h). Les agneaux se

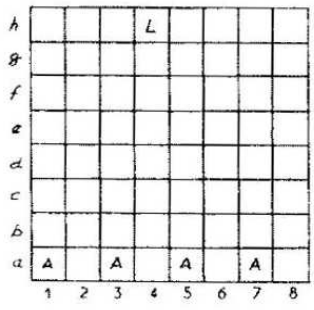


Fig. 1
Répartition de départ sur un échiquier 8x8

trouvent en (1,a), (3,a), (5,a) et (7,a). Loup et agneaux se déplacent uniquement en diagonale (comme les fous aux échecs), mais d'une seule case à chaque coup. Une seule pièce bouge à chaque fois et l'on ne peut pas passer son tour. Les agneaux ne peuvent qu'avancer, mais le loup peut avancer ou reculer. Enfin, il n'y a pas de prise, ni de superposition : une pièce ne doit pas venir sur une case occupée par une autre pièce.

Le loup gagne s'il arrive à passer derrière les agneaux pour se poser sur l'une des cases de la ligne a. Les agneaux seront considérés comme gagnants s'ils arrivent à bloquer le loup (celui-ci ne peut plus jouer) ou s'ils parviennent tous sur la ligne h avant que le loup ne soit sur la ligne a. Nous n'envisagerons pas ici le problème que pose la programmation des stratégies respectives du loup ou des agneaux. Selon la capacité de votre machine et votre connaissance du jeu, vous pourrez imaginer un programme grâce auquel l'ordinateur s'occupe du déplacement des agneaux (il existe alors une stratégie gagnante) ou du déplacement du loup (qui devra alors réussir à gagner en face d'un joueur peu expérimenté).

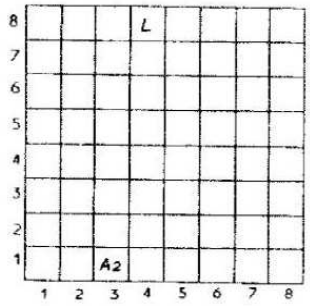
Notre propos est simplement de transformer le jeu de telle sorte qu'il puisse être pratiqué par deux personnes disposant d'une machine programmable de poche. L'étude est faite plus particulièrement pour la TI-57, mais la généralité des solutions proposées les rend valables bien sûr sur tout autre appareil.

Repérer la position des pièces

Le petit nombre de mémoires disponibles ne permet pas de noter facilement la situation d'occupation de chaque case de l'échiquier. Une mémoire particulière sera utilisée pour repérer la position du loup (mémoire M7 dans notre exemple) et une mémoire pour la position de chacun des agneaux (M1 pour le premier, M2 pour le second...).

Chaque mémoire peut contenir un nombre de 8 chiffres en virgule flottante : chaque individu pourra donc être noté sous forme d'un couple (colonne, ligne). Ainsi, le couple (4,8) en mémoire M7 signifie que le loup se trouve dans la

Fig. 2
Le loup est en (4,8) et l'agneau 2 est en (3,1)



colonne 4, ligne 7, soit (4,h) avec la notation citée au début. De même, (3,1) en mémoire M2 signifie que l'agneau n° 2 se trouve en colonne 3, ligne 1 (voir fig. 2).

Si la machine utilisée dispose d'une capacité plus importante, il peut être préférable d'affecter deux mémoires au repérage de la position d'une pièce, le repérage étant alors fait par deux entiers, ce qui rend les contrôles ultérieurs plus simples et plus rapides.

Comment passer d'une case à l'autre ?

Le loup dispose de 4 possibilités de déplacement (il peut avancer ou reculer en diagonale) ; les agneaux ne disposent chacun que de deux possibilités (avancer en diagonale). D'où le schéma de la figure 3. Comment réaliser concrètement ce déplacement, dans le cas le plus général, celui du loup ? Observons la situation sur une case quelconque de l'échiquier (x, y) telle qu'elle est représentée à la figure 4.

Cette codification, la plus usuelle, n'est bien entendu pas la seule possible. Souvenons-nous maintenant que nous avons décidé de repérer chaque position par un décimal *x, y ;* ce qui signifie que le repérage colonne est fait par un entier x, mais

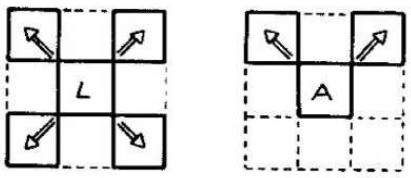


Fig. 3
Le loup a plus de liberté que les agneaux dans ses mouvements.

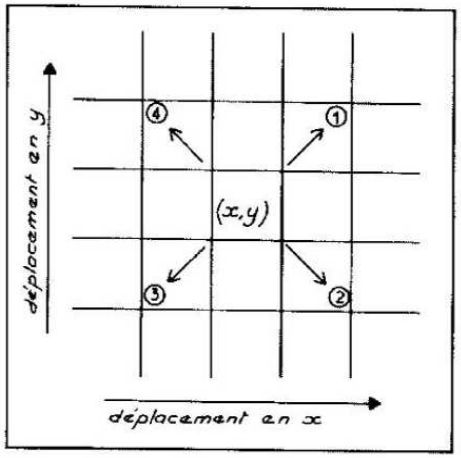


Fig. 4
Les quatre cases où le loup peut se rendre sont :
x + 1, y + 1 : déplacement 1
x + 1, y - 1 : déplacement 2
x - 1, y + 1 : déplacement 3
x - 1, y - 1 : déplacement 4

que le repérage ligne est en fait réalisé par un décimal *0, y* et c'est la somme de ces deux nombres qui définit notre position : *x, y = x + 0, y.*
Par exemple : 3,5 = 3 + 0,5

Pour ajouter 1 à x et 1 à y, il faut donc en réalité ajouter 1 et 0,1 soit ajouter 1,1. Pour ajouter 1 à x et enlever 1 à y, il faut ajouter 1 et enlever 0,1 ce qui revient à ajouter 0,9 (en effet : 1 - 0,1 = 0,9). Pour enlever 1 à x et enlever 1 à y, il faut enlever 1 et enlever 0,1 ce qui revient à enlever 1,1 (en effet -1 -0,1 = -1,1). Pour enlever 1 à x et ajouter 1 à y, il faut enlever 1 et ajouter 0,1 ce qui revient à enlever 0,9. (En effet, -1 + 0,1 = -0,9). On remarque que deux valeurs absolues seulement interviennent dans les résultats : les valeurs 1,1 et 0,9, d'ailleurs liées par la relation 1,1 + 0,9 = 2.

Comme il nous reste suffisamment de mémoires disponibles, nous utiliserons les mémoires M5 et M6 pour stocker ces deux nombres (en réalité, nous mettrons 1,1 dans M5 et -0,9 dans M6).

Nous savons comment réaliser concrètement le déplacement ; mais il nous faut communiquer à la machine la référence de la pièce à déplacer et la direction du déplacement. Une première possibilité, sur laquelle nous aurons l'occasion de revenir, consiste à associer un code à chacune des directions de déplacement possibles, par exemple, 1, 2, 3 et 4. Les agneaux devront alors se servir

seulement des déplacements 1 et 4 (avec un contrôle éventuel). Cette méthode oblige à prévoir dans le programme un décodeur qui associe à 1 le déplacement +1,1 ; à 2 le déplacement +0,9, etc. La solution que nous avons retenue est plus facile à mettre en œuvre, elle consiste à affecter un sous-programme particulier à chacune des directions de déplacement, ce sous-programme devant être appelé chaque fois que l'on désire effectuer un déplacement dans la direction correspondante.

Les noms des sous-programmes doivent être choisis de façon aussi parlante que possible. Nous avons retenu ici une convention grâce à laquelle les noms des déplacements sont en accord avec la disposition des touches du clavier de la calculatrice : le loup étant supposé se trouver à l'emplacement de la touche 5, il peut se déplacer dans les directions : 7, 9, 1 et 3 (voir fig. 5).

Pour les agneaux, le choix d'un déplacement devra être précédé du numéro de l'agneau déplacé ; on a défini deux autres sous-programmes : l'agneau étant supposé se trouver sur la touche du 2, il peut prendre la direction du 4 ou 6. Ainsi, 3 SBR 4 déplacera l'agneau n° 3 d'une case dans la direction SBR 4.

Initialisation du programme

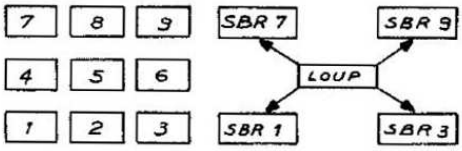
Taper : 4.8 STO 7 position initiale du loup
1.1 STO 1 position initiale du 1er agneau
3.1 STO 2 deuxième agneau
5.1 STO 3 troisième agneau
7.1 STO 4 quatrième agneau
1.1 STO 6] déplacemts
0. 9 +/-STO 5] les plus fréquents

Faire : 2nd Fix 1

Exemple de début de partie :

SBR 1	Le loup avance 3.7 nouvelle position
1 SBR 6	Le 1er agneau avance 1.1 (déplacement)
RCL 1	2.2 Le 1er agneau est passé de 1.1 en 2.2
SBR 3	Le loup avance 2.8 nouvelle position
2 SBR 6	Le 2e agneau avance 1.1 (déplacement)
RCL 2	4.2 Le 2e agneau est passé de 3.1 en 4.2
.	
.	
.	
etc	

Fig. 5
Les numéros des sous-programmes de déplacement ont été choisis parce qu'ils rappellent la disposition des touches numériques de la calculatrice.



On peut résumer les déplacements à réaliser sous la forme d'un tableau :

Numéro du SBR	1	3	7	9	4	6
Action réalisée	-1,1	+0,9	-0,9	+1,1	-0,9	+1,1
	Loup				Agneau	

Les actions les plus fréquentes sont +1,1 et -0,9, et c'est la raison pour laquelle on range +1,1 en M5 et -0,9 en M6. Nous venons de voir que chaque déplacement d'un agneau (SBR 4 ou SBR 6) est précédé du numéro de l'agneau déplacé. En fonction de ce numéro, le déplacement devra être affecté soit à la mémoire M1, soit à M2, M3 ou M4.

Le procédé que nous utiliserons exploite la possibilité de décrémentation automatique de la TI-57 (décrémenter signifie ici : enlever 1 dans la mémoire de décrémentation). Nous allons en effet ajouter le déplacement dans M1, puis tester le numéro de l'agneau après décrémentation grâce à un sous-programme particulier ; si ce n'est pas le bon, on enlève le déplacement de M1 et on l'ajoute dans M2 ; on continue ainsi jusqu'à M4.

Du côté des améliorations

Si la machine dont vous disposez a une capacité suffisante, il sera intéressant de prévoir un programme de contrôle de non-dépassement des limites du terrain de jeu. Ce contrôle devra être effectué avant l'éxécution du coup et refuser le coup s'il n'est pas autorisé. De même, il serait bon de prévoir un contrôle s'assurant de la validité des coups joués, c'est-à-dire vérifiant que le loup n'arrive pas sur une case occupée par un agneau ou que les agneaux n'arrivent pas sur une case occupée par un autre agneau ou par le loup. Un dernier contrôle, enfin, devrait permettre de dire si le loup est arrivé dans une position gagnante ou si au contraire ce sont les agneaux qui ont gagné. Pour la calculatrice minimale considérée ici (TI-57), le seul contrôle consiste à vérifier au moyen d'une batterie de tests que le loup n'arrive pas sur une case occupée par l'adversaire, ce cas donnant lieu à un affichage clignotant, avec non-exécution du coup.

L'affichage du damier n'est pas possible sur la TI-57 ; seule la position du loup est affichée après chacun de ses déplacements. En ce qui concerne les agneaux, leurs déplacements sont affichés, mais pour connaître leur position, on doit regarder le contenu des mémoires M1, M2, M3, M4. Cela étant, si l'on dispose de davantage de place, un simple afficheur sur 8 chiffres, avec 2 chiffres supplémentaires pour l'exposant, permet de suivre complètement le déroulement du jeu sur la calculatrice (voir fig. 6).

1 1	3 1	5 1	7 1	48
1er agneau	2ème agneau	3ème agneau	4ème agneau	loup

Fig. 6
Une des façons de codifier la position de cinq pièces sur l'afficheur de la TI 57.

Pour commencer une partie, après avoir correctement initialisé les mémoires 1 à 7, on tire à pile ou face lequel, du loup ou d'un agneau, se déplacera en premier ; le loup avance ou recule en faisant SBR n (avec n=1, ou 3, ou 7, ou 9) ; les agneaux se déplacent avec k SBR p,(k=1, ou 2, ou 3, ou 4, autrement dit le numéro de l'agneau, et p=4 ou 6, la direction du déplacement).

La partie se termine si le loup arrive sur la ligne 1 (le loup a gagné) ou si les agneaux réussissent soit à le bloquer, soit à gagner tous la ligne 8 : ils l'ont alors échappé belle !

☐ Jacques Deconchat