Le solitaire

# Présentation

Jeu de plateau à un joueur

Le plateau est constitué d’un ensemble de cases carrées

Il existe principalement deux configurations :

|  |  |
| --- | --- |
| Le plateau classique (33 cases) | Le plateau français (37 cases) |
| |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | | |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | |

Au départ, chaque case est occupée par une pierre, sauf au moins une case.

Dans les jeux classiques et français, seule la case centrale est vide.

Les cases contigües sont celles qui partagent un même bord. Il n’y a pas de mouvements en diagonale.

Un mouvement consiste à choisir une pierre A et à la faire sauter par-dessus une pierre B contigüe pour autant que A atterrisse dans une case vide du plateau. Alors B est retiré. Cela revient à déterminer trois cases contiguës alignées, les deux premières contenant une pierre, la troisième à une des extrémités est vide, et d’inverser la présence des pierres dans ces trois cases.

Si on note x la présence d’une pierre, o une case sans pierre, les 4 mouvements de base sont :

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| x x o | → | o o x |  | o x x | → | x o o |  | x x o | → | o o x |  | o x x | → | x o o |

Chaque mouvement fait donc décroitre de exactement 1 unité le nombre de pierres sur le plateau.

L’objectif est d’arriver à ne conserver qu’une seule pierre sur le plateau.

La position de la dernière pierre n’est pas déterminante.

S’il y a N pierres sur le plateau initial, il faut exactement N-1 mouvements pour atteindre la solution (si elle existe).

# Volume

Calcul grossier du nombre de situations différentes :

Sur un plateau de N cases, chaque case peut contenir ou non une pierre. Le nombre théorique de situations différentes est donc

Pour 33 cases, cela représente environ 9 milliards de cas.

Pour 37 cases, cela représente environ 140 milliards de cas.

On peut réduire ces cas d’un facteur 8 en considérant les symétries possibles.

Le nombre de cas dans les premières étapes est lui aussi réduit. Mais on peut penser que plus de mouvements sont effectués, et plus de situations sont possibles.

# Fiche de jeu

C’est un fichier texte contenant la situation initiale, et éventuellement une solution lorsqu’elle a été trouvée.

La situation initiale est décrite en premier. Si une solution est enregistrée, elle est précédée d’une ligne dont le 1er caractère est un tiret ‘-‘.

La situation initiale est une suite de lignes non vides constituées d’espaces ‘ ‘, de lettres ‘x’ ou ‘o’.

Le ‘x’ désigne une pierre, le ‘o’ une case vide.

L’algorithme calcule automatiquement la largeur du plateau L.

Une solution est une suite de mouvements à effectuer pour résoudre le jeu.

Un mouvement est un couple d’indices (D,A). D est l’indice de la pierre déplacée, A est l’indice de la case vide recevant la pierre initialement en D.

Le plateau est inclus dans un rectangle de largeur L, les indices décrivent les cases de ce rectangle de gauche à droite en partant du haut.

# Recherche

## Algorithme en profondeur

A chaque mouvement, le nombre de pièces diminue d’une unité, donc on ne peut établir de boucle dans une suite de mouvements. Ce qui autorise la recherche récursive arborescente en profondeur.

Dans les deux cas envisagés, il y a 33 cases ou 37 cases, initialement 32 ou 36 pierres.

Le nombre de mouvements d’une éventuelle solution (la profondeur récursive) est donc de 31 ou 35.

Ici, le multi threading aurait pu être envisagé, aucun accès concurrentiel n’étant à effectuer (hormis un flag indiquant qu’une solution a été trouvée) une fois les structures dupliquées.

Dans le cas du jeu classique, très peu de temps a été nécessaire sur un Core i7 cadencé à 3.5 Ghz pour trouver la solution. Il se trouve que les choix effectués ont été favorables pour sélectionner les « bons » mouvements.

Dans le cas du jeu français, 17 heures de calcul n’ont pas abouti.

## Pistes d’amélioration

Les deux jeux présentent une symétrie carrée, de nombreuses situations équivalentes par l’une des 4 rotations ou 4 symétries axiales et diagonales peuvent être abandonnées. Mais cela oblige alors à constituer un stock des situations rencontrées pour comparaisons.

On doit alors faire un choix parmi ces deux stratégies :

* Stocker une situation et ses symétries pour identifier rapidement si une nouvelle situation a déjà été rencontrée, ou est équivalente à une de ses symétries
* Stocker un seul exemplaire, mais alors il faut générer pour toute nouvelle situation ses symétries et tester leur présence éventuelle dans le stock.

De nombreux embranchements peuvent ensuite aboutir à des solutions déjà rencontrées, et la constitution d’un stock de situations rencontrées permet la aussi d’abandonner des situations déjà rencontrées. En particulier, certains mouvements possibles d’une situation donnée peuvent être effectués dans un ordre quelconque et aboutir alors à des situations identiques.

Par exemple : si on peut effectuer un mouvement M1 impliquant les cases A, B et C, et un autre mouvement M2 impliquant les cases D, E et F, et si ces deux groupes de cases sont disjoints, alors effectuer M1 puis M2, ou M2 puis M1 aboutit au même résultat.

Mais l’effort à faire pour détecter ces situations semble consommateur de temps et peu économe en place mémoire.

* On effectue M1 depuis la situation S, on inscrit le résultat S1 dans le stock, puis on effectue M2 sur S1, et on stocke le résultat S12.
* Ou on effectue M2 depuis la situation S, on inscrit le résultat S2, différent de S1, dans le stock. Puis on effectue M1 sur S2, et on découvre alors que le résultat S21 est égal au résultat S12 précédemment stocké.

A l’inverse, détecter que M2 peut permuter avec M1 sur S, et donc ne pas l’effectuer n’empêche pas de retrouver S12, égal au S21 qu’on aurait sinon obtenu.

Plus le nombre de mouvements permutables deux à deux est important, plus il semble pertinent de détecter ces groupes pour n’effectuer qu’un mouvement du groupe.

L’accès continuel en lecture et écriture à ce stock, et donc nécessairement exclusif, par contre, fait perdre beaucoup d’intérêt au multi-threading.

Dans certaines situations, une pierre isolée ne pourra plus être éliminée. Mais il est difficile d’établir des critères permettant de repérer avec certitude ces situations, surtout s’il reste beaucoup de pierres sur le plateau, et donc de potentialités pour aller « repécher » une pierre apparemment isolée.

On peut éventuellement, à partir d’une situation donnée, envisager toutes ses descendantes et les classer selon un critère comme la compacité de la disposition des pierres pour éviter d’en laisser trop éloignées.

Il semble préférable, si on gère un stock pour déterminer des positions égales ou équivalentes à éliminer, de répartir ces situations dans des stocks différents selon le nombre de pierres de la situation.

On peut alors envisager une recherche en largeur et non plus en profondeur, mais si cela permet de ne gérer que deux stocks simultanément, et donc réduit l’occupation mémoire, cela ralentit la découverte d’une éventuelle solution opportuniste.

## Les symétries

Le plateau est composé de cases, les centres de ces cases ont des coordonnées entières dans le repère orthonormé choisi.

Les coordonnées de ces cases sont situées dans deux intervalles identifiés : [xMin, xMax], [yMin, yMax]

Une « symétrie » est une rotation ou une symétrie axiale qui conserve globalement toutes les cases du plateau. On envisage les 3 rotations de 90°, 180°, 270°, dont le centre reste à déterminer, et les symétries axiales selon des axes parallèles aux axes du repère ou à ses deux diagonales principales.

Pour que le plateau soit conservé, il est nécessaire, mais pas suffisant, que son rectangle englobant le soit. Cela est le cas pour la rotation 180° et les deux symétries principales. Cela n’est le cas pour les autres symétries que si le rectangle est un carré.

Le centre du rectangle englobant joue un rôle essentiel dans la recherche de symétries. Mais dans le cas général, ses coordonnées peuvent ne pas être entières. Soient

ses coordonnées. Si le rectangle est un carré , les coordonnées de son centre sont soit toutes deux entières, soit toutes deux demi-entières. Leur somme ou leur différence est donc entière.

Si

Rotation 90° autour du centre du carré englobant

Rotation 180° autour du centre du rectangle englobant

Rotation 270° autour du centre du carré englobant

Symétrie du rectangle selon un axe vertical

Symétrie du rectangle selon un axe horizontal

Symétrie du carré selon une parallèle à la 1ère diagonale

Symétrie du carré selon une parallèle à la 2ième diagonale

# Résolution incrémentale

## Présentation

Il apparait, c’était attendu, que la recherche de solution du jeu français, naïve, ou même en tenant compte des symétries, ne peut aboutir faute de place mémoire.

Par contre, il est très rapide de trouver une solution au jeu classique (moins de 0,2 seconde).

Un jeu classique sur un plateau français est lui aussi résolu en peu de temps (moins de 120 secondes).

On voit que l’ordre dans lequel l’algorithme teste les mouvements a son importance.

Et quelques essais ont montré que le fait de partir de situations sur le plateau français où quelques pièces ont été retirées permet de résoudre ces situations, sans pour autant que j’aie trouvé de situation initiale qui ait une solution.

Le challenge est maintenant de savoir s’il existe une solution sur plateau classique ou français lorsqu’initialement une pierre exactement est retirée, quel que soit son emplacement.

## Partir de moins loin

L’idée est alors de constituer une liste des situations initiales de départ en jouant quelques coups depuis la situation originelle, Puis d’étudier chacune des situations obtenues séparément. Cela veut dire par contre qu’on va recalculer les situations intermédiaires à chaque nouvelle situation de départ, mais je n’envisage pas d’écrire sur disque des giga octets de données persistantes. D’autant plus qu’avoir rencontré une situation ne renseigne pas sur le fait qu’elle soit gagnante ou non, et la rencontrer à nouveau depuis une situation initiale différente n’apporte alors aucune information.

Première étape : évaluer combien de mouvements on peut raisonnablement effectuer à partir d’une situation initiale donnée pour constituer les situations intermédiaires qu’il faudra évaluer individuellement.

En tenant compte des symétries, il existe exactement 8 situations initiales originales sur un plateau français (7 sur un plateau classique) selon la position de la case vide initiale.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Les 8 situations initiales originales, une seule pierre retirée, avec mise en évidence des symétries. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

J’ai préparé 8 fichiers texte décrivant chacun une de ces situations initiales. Ils portent les noms « Français [n].txt ». [n] = 0 pour la situation classique, puis les autres situations sont décrites dans les fichiers de n° 1 à 7.

Dans une première version, j’ai fait générer les stocks pour chacune de ces 8 situations jusqu’à une profondeur de 10 mouvements. Puis j’ai tracé les quantités de situations différentes inscrites dans ces stocks réparties par nombre de mouvements effectués.

On remarque que les situations 0 et 2 génèrent exactement le même nombre de situations à chaque niveau jusqu’au niveau 10.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Français 0 : 50 secondes | | | | | | | | | | |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 1 | 1 | 3 | 15 | 70 | 341 | 1 604 | 6 950 | 27 948 | 102 261 | 335 839 |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Français 1 : 150 secondes | | | | | | | | | | |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 1 | 2 | 6 | 32 | 173 | 908 | 4 628 | 21 895 | 94 892 | 374 264 | 1 327 244 |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Français 2 : 50 secondes | | | | | | | | | | |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 1 | 1 | 3 | 15 | 70 | 341 | 1 604 | 6 950 | 27 948 | 102 261 | 335 839 |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Français 3 : 80 secondes | | | | | | | | | | |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 1 | 1 | 3 | 18 | 98 | 525 | 2 627 | 12 347 | 53 537 | 210 705 | 744 605 |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Français 4 : 180 secondes | | | | | | | | | | |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 1 | 2 | 9 | 52 | 274 | 1 380 | 6 618 | 29 447 | 120 646 | 449 833 | 1 509 235 |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Français 5 : 125 secondes | | | | | | | | | | |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 1 | 2 | 7 | 31 | 176 | 891 | 4 340 | 19 782 | 82 820 | 315 494 | 1 081 799 |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Français 6 : 180 secondes | | | | | | | | | | |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 1 | 2 | 9 | 47 | 252 | 1 282 | 6 245 | 28 161 | 117 201 | 445 706 | 1 529 920 |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Français 7 : 160 secondes | | | | | | | | | | |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 1 | 3 | 10 | 47 | 251 | 1 248 | 5 915 | 26 178 | 106 417 | 394 935 | 1 320 808 |

Bonne nouvelle, j’ai trouvé une solution à partir de « Français 1.txt » auquel j’ai fait subir 10 mouvements pour tester la situation suivante :

Liste des 10 mouvements à effectuer pour arriver à la situation intermédiaire ci-dessous :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| (1;-3)← | (0;-1)↑ | ( -1;-3)→ | (-2;-2)→ | (-2;-1)→ |
| (1;-1)← | (1;-3)↓ | (1;0)↑ | (3;-1)← | (1;-2)← |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | -3 | -2 | -1 | 0 | 1 | 2 | 3 |
| -3 |  |  |  |  |  |  |  |
| -2 |  |  | x |  |  | x |  |
| -1 | x |  | x |  | x |  |  |
| 0 | x | x | x | x |  | x | x |
| 1 | x | x | x | x | x | x | x |
| 2 |  | x | x | x | x | x |  |
| 3 |  |  | x | x | x |  |  |

Liste des 25 mouvements à effectuer depuis la position intermédiaire pour résoudre le jeu :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| (-1;0)→ | (-3;0)→ | (-2;2)↑ | (0;1)← | (-2;1)↑ |
| (0;3)↑ | (2;2)← | (1;0)↓ | (2;1)↑ | (-1;3)↑ |
| (3;1)↑ | (0;1)← | (-3;1)→ | (-1;0)↓ | (-1;-2)↓ |
| (1;3)↑ | (-1;2)→ | (1;2)↑ | (1;0)↑ | (2;-2)← |
| (3;-1)← | (-3;-1)→ | (-1;0)↑ | (-1;-2)→ | (1;-1)↑ |

Cette recherche a pris 7 minutes et 8 secondes.

Il y a environ 8 185 000 situations différentes après 10 mouvements effectués sur l’une des 8 situations françaises. A raison de 7 minutes par situation, cela représente environ 107 années de calcul ininterrompu.

Et encore : les 7 minutes ont été obtenues par découverte d’une solution. Combien de temps aurait été nécessaire pour tester une situation intermédiaire sans solution ?

## Par les deux bouts

### Présentation

Partant d’un plateau de NC cases donné,

* On calcule toutes les situations possibles où une pierre est retirée exactement. Appelons cet ensemble EI. Le plateau français comporte 8 telles situations aux symétries près.
* On calcule toutes les situations obtenues à partir des situations où une pierre quelconque est enlevée, et en jouant ND coups. On les sauvegarde pour pouvoir reprendre la recherche après une interruption. Appelons cet ensemble ED. Il contient les situations à NC – 1 – ND pierres.
* On calcule toutes les situations où une pierre exactement reste sur le plateau. Ce sont les situations gagnantes. Appelons cet ensemble EG. Il y en a 8 dans le cas du plateau français.
* On calcule toutes les situations qui aboutissent à une solution de EG après NF mouvements. On les sauvegarde aussi. Appelons cet ensemble EF. Il contient des situations à NF + 1 pierres.
* Puis on teste chaque situation SD de ED pour trouver des mouvements qui aboutissent à une situation SF de EF. Si on en trouve une, on élimine toutes les situations de EI qui permettent d’obtenir cette situation SD gagnante.
* Si EI est vide, on a démontré qu’il existe une solution à chacune des situations initiales de EI.
* Si on veut interrompre le processus, on enregistre la situation de ED actuellement testée, ou du moins son index dans ED, afin de pouvoir repartir de celui-ci au redémarrage.

EI 🡺 ED 🡺 EF 🡺 EG

Le calcul de EF n’est pas très différent du calcul de ED, il suffit de partir de toutes les situations où il n’y a qu’une pierre en place et d’effectuer NF mouvements opposés à ceux permettant de supprimer une pierre.

Il faut aussi pouvoir retrouver les mouvements qui mènent d’une situation de départ à la situation SD de ED pour laquelle a été trouvée une solution. Et aussi ceux qui mènent de la situation de EF à la situation gagnante de EG. En joignant les 3 séries de mouvements, on obtient la solution.

On ne conserve pas les chemins qui relient EI à ED, ni ceux qui relient EF à EG. On les recalcule quand on en a besoin.

De la même manière, on ne sauvegarde pas les situations entre ED et EF lors de la suspension de la recherche. Cela représente trop de volume de données. Et si on interrompt la recherche, c’est en partie parce que la mémoire sature, et qu’on veut la libérer en repartant d’un stock vide.

Il reste à choisir ND et NF pour limiter les temps de calcul et l’occupation mémoire.

### Fonctionnel

Choix d’un fichier xml « chemin/nom.xml » qui décrit le plateau initial, et fournit des valeurs aux paramètres ND et NF.

Nous stockons les données dans un répertoire « chemin/nom »

Dans ce répertoire,

* un fichier de pilotage nom.xml, initialement égal à celui fourni. Et qui contiendra les éventuelles solutions trouvées et les informations de reprise en cas de suspension de la recherche.
* deux fichiers binaires ED.dat et EF.dat calculés lors de l’initialisation, et qui ne sont plus modifiés par la suite.

L’outil permet de choisir le fichier originel « chemin/nom.xml » contenant une description du plateau et des valeurs pour les deux paramètres ND et NF

Si le répertoire n’existe pas, il le crée et l’initialise

S’il existe mais que la situation initiale du fichier diffère de celle sauvée dans le fichier de pilotage, ou l’un des paramètres ND ou NF, il efface le contenu du répertoire et le réinitialise (après confirmation).

Initialisation du répertoire :

* Création du fichier de pilotage nom.xml
* Création des deux fichiers de données initiaux.

Fichier de démarrage, exemple :

<?xml version="1.0" encoding="utf-8"?>

<solitaire>

<parametres nd="9" nf="12" />

<plateau>

xxx

xxxxx

xxxxxxx

xxxxxxx

xxxxxxx

xxxxx

xxx

</plateau>

</solitaire>

Fichier de pilotage, exemple :

<?xml version="1.0" encoding="utf-8"?>

<solitaire>

<parametres nd="9" nf="12" />

<plateau>

xxx

xxxxx

xxxxxxx

xxxxxxx

xxxxxxx

xxxxx

xxx

</plateau>

<reprise idx="999999999" />

<solution>

<plateau>

oxx

xxxxx

xxxxxxx

xxxxxxx

xxxxxxx

xxxxx

xxx

</plateau>

<mouvement x="1" y="-3" dir="o" />

<mouvement x="0" y="-1" dir="n" />

</solution>

</solitaire>

Les valeurs de c sont les indices des cases du plateau. Le plateau est contenu dans un rectangle minimal dont les cases sont numérotées à partir de 0 de gauche à droite en partant du haut.

Les directions sont o, n, e, s pour ouest, nord, est et sud.

La balise <plateau> contient la description fournie dans le fichier original, et sert de base pour une comparaison lors d’une reprise, avec parametres/@nd et parametres/@nf.

Si la situation décrite dans le fichier fourni n’a aucune case inoccupée, alors l’outil recherche une solution pour chaque situation issue de ce plateau complet, duquel il retire une pierre (aux symétries près). Sinon il recherche une solution pour la situation fournie uniquement.

La valeur de reprise/@idx est initialisée à 0, et mise à jour avec l’indice de la situation en cours de test, et avec laquelle la recherche repart. Cet indice est l’index dans le fichier ED.dat de la situation à prendre en compte.

Si le plateau initial est complet et qu’il a N cases (et donc N pierres), alors ED.dat contient les situations accessibles après ND mouvements appliqués à des situations initiales ayant N-1 pierres : le nombre de pierres dans les situations de ED.dat est N – 1 – ND .

Si le plateau initial est incomplet, qu’il a NC cases et NP pierres ( NP < NC ), alors ED.dat contient les situations accessibles après ND mouvements appliqués à cette situation initiale ayant NP pierres : le nombre de pierres dans les situations de ED.dat est NP – ND .

Le fichier EF.dat contient les situations obtenues après NF mouvements inverses sur les situations à 1 pierre, ces situations ont donc NF – 1 pierres.

Ces deux fichiers contiennent des tableaux d’octets qui donnent les indices des pierres des situations calculées initialement.

# Résolution manuelle

BPS : bouton principal de la souris

## Objectif

* Pouvoir déplacer (drap & drop) une pierre vers une destination valide, la lâcher et mettre à jour le plateau (pas d’animation).
* Pouvoir sélectionner une pierre pour laquelle un seul mouvement est possible et déclencher une animation de déplacement de la pierre
* Pouvoir sélectionner une pierre pour laquelle au moins deux mouvements sont possibles, puis sélectionner une de ses destinations valides, et déclencher une animation de déplacement de la pierre
* L’animation est courte, mais tenir compte des autres interactions (en particulier retour en arrière) pour l’interrompre (fin anticipée).

## Les actions

* Ep : Enfoncer le BPS sur une pierre pour laquelle au moins un mouvement est possible
* Ev : Enfoncer le BPS sur une case vide accessible à la pierre actuellement sélectionnée
* Rp : Relâcher le BPS au-dessus de la pierre actuellement sélectionnée
* Rv : Relâcher le BPS au-dessus de la case vide actuellement sélectionnée
* D : Déplacer le curseur alors qu’une pierre est sélectionnée
* S : sélectionner la pierre pointée

## Les états

* L : Libre
* S : Une pierre est sélectionnée
* Drag : Une pierre est S et déplacée (drag)
* Drop : Une pierre est S et lâchée
* V : Une pierre est S et une case vide est sélectionnée
* A : Une animation (déplacement visuel de pierre) est en cours de réalisation

Si Ep

Si L

S

Si S

S