

Etude mathématique-algorithmique

Par Mohamed CHERIF, Bruno GABRIEL, Xavier MARTIN

Table des matières

Etude intuitive	2
Recherche sur un plateau 3x3	2
Déterminer une méthode à privilégier	4
Recherche du meilleur rapport efficacité-temps	4
Retour sur la méthode la plus sur.....	4
Recherche sur le temps pour cette méthode.....	4
Faut-il faire des concessions sur l'efficacité pour gagner en temps ?	4
Améliorations à apporter	5

I) Etude intuitive

a) Recherche sur un plateau 3x3

Le tableau 3x3 est celui qui est particulier pour plusieurs raisons. Tout d'abord, il s'agit du plus petit tableau possible (et qui présente un minimum d'intérêt).

Ensuite, il est très simple de le résoudre car il ne contient qu'une seule pièce au milieu. Ainsi, en plaçant dans un premier temps les coins, puis les bords, il ne reste plus qu'à trouver la bonne rotation de la pièce centrale.

Revenons à cette résolution mais étape par étape.

En effet, lors du positionnement des 4 coins, il se peut que l'on ne puisse pas placer totalement les bords (autres que les coins). Il faut donc placer un coin, puis associer les côtés correspondants pour avoir le coin suivant qui correspond. Enfin, il suffit comme dit précédemment de placer le centre.



On obtient ainsi une méthode sur et rapide qui permet de résoudre un plateau 3x3.

Cependant, cette résolution marche uniquement pour ce modèle de plateau. On peut tout de même essayer de voir si une résolution du contour puis du centre du plateau ne serait pas la plus adaptée.

a) Recherche sur des plateaux de diverses tailles (carré ou rectangle)

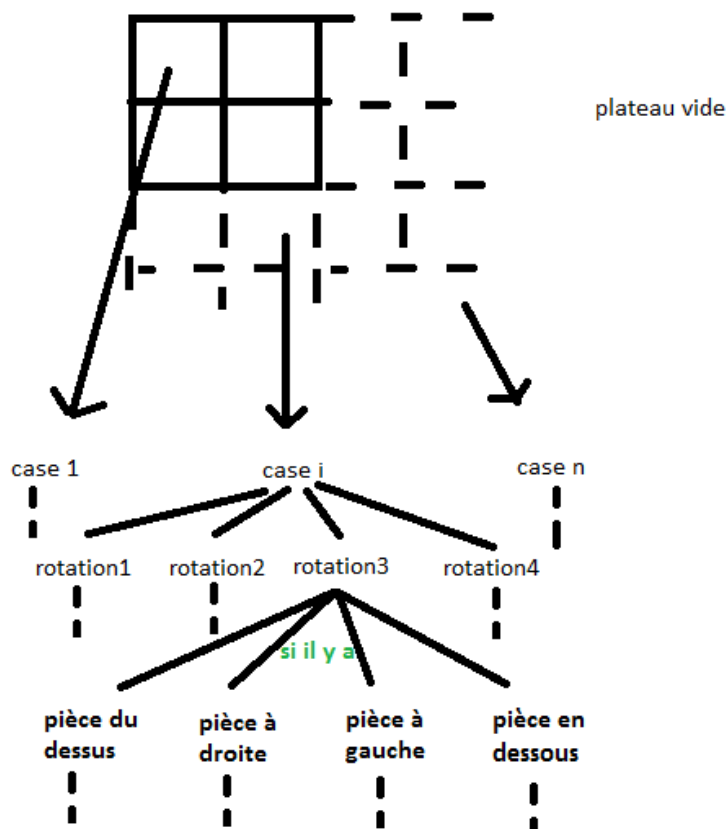
Nous allons tout d'abord voir ce que le modèle précédent donne sur un plateau 4x4.



Ainsi, nous obtenons un chemin relativement facile à réaliser qui semble résoudre de manière fiable le plateau. Cependant, si nous prenons un plateau réellement plus grand, un problème arrive très vite : si l'anneau précédent est erroné, tout ce qui a été réalisé par la suite est faux.

Dès lors, il semble bon de voir avec d'autres modèles de résolutions. Le constat est sans équivoque : quel que soit le parcours utilisé, la moindre erreur à un instant t rendra faux les éléments suivants ($t+x$).

On remarque donc (sans même réaliser de calculs) qu'il faut plusieurs années (pour pas dire millions voir milliard) pour parcourir cet arbre (représenté ci-dessous)



b) Déterminer une méthode à privilégier

Après cette brève étude intuitive, nous remarquons que la tâche ne semble pas aussi aisée que cela. Malgré un bon moyen de résoudre le plateau le plus petit, il ne semble pas que cette méthode soit plus pertinente qu'une autre pour ce qui est des plateaux de dimensions supérieurs.

II) Recherche du meilleur rapport efficacité-temps

a) Retour sur la méthode la plus sûre

La méthode la plus sûre est celle dont les probabilités de placer une bonne pièce à un instant est le plus élevé. Afin de voir selon les chemins parcourus lequel est la plus sûre, nous avons réalisé une succession de parcours avec des plateaux de différentes tailles.

On remarque qu'à l'exception de certains parcours, le « poids » des plateaux complets sont strictement les mêmes. D'où l'intérêt de choisir les poids les plus forts au début du parcours. Ainsi, on réduit les retours en arrière. Si l'on reprend ce que l'on a trouvé précédemment, ça revient à réduire la variable x (dans $t+x$).

b) Recherche sur le temps pour cette méthode

Le problème des méthodes possédant une forte fiabilité dès le début est qu'elles demandent un grand nombre de tests. Ce sont tous ces tests qui vont allonger le temps d'exécution du programme. Il faut donc trouver un moyen pour permettre de privilégier les tests qui ont le plus de chances d'aboutir à la réalisation du plateau.

c) Faut-il faire des concessions sur l'efficacité pour gagner en temps ?

L'aspect temps est important, mais « perdre » du temps afin de s'assurer que l'on évolue sur des bases solides réduit de manière considérable des branches à parcourir par la suite. Il y a donc malgré tout gain de temps.

III) Améliorations à apporter

Enfin, afin de permettre d'être plus rapide, nous devons absolument réduire au maximum le parcours de l'ensemble de l'arbre (qui représente toutes les possibilités).

En premier, il faut connaître le nombre de chaque « couleur » (dans le sens où une pièce est composée de 4 couleurs) des pièces pas encore jouées. Ainsi, si l'on place une pièce et que l'on déduit du nombre de couleur restante le nombre de cette même couleur présente sur la pièce, on peut savoir quand il en reste qu'une et dans quel cas il faut assembler ces 2 pièces. On gagne ainsi à la fois sur le nombre de pièces à tester et sur les rotations.

Cependant, rien ne nous garantit qu'il ne reste pas 2 autres pièces avec la même couleur déjà posées sur le plateau et sans être assemblé avec une paire. Il faut donc constamment vérifier avec un modulo 2 si une couleur a déjà une pièce en attente de placement sur le jeu ou mieux encore, compter le nombre de pièces d'une couleur déjà placées et en attente d'une paire. Il reste plus par la suite qu'à vérifier si le nombre de couleur dans les pièces non jouées est supérieur ou égale au nombre de couleurs en attente sur le plateau. Ceci permet d'identifier rapidement les pièces que l'on peut tenter de placer ou celles qui devront pas être placées (si le nombre de couleur hors plateau égale le nombre de couleur en attente sur plateau, on devra placer uniquement les dites couleurs avec les couleurs en attente, et les réservées à cet emploi. C'est-à-dire qu'il sera impossible de les placées ailleurs).

Ensuite, l'idée consiste à exploiter les données qui ont été créées afin de déterminer les priorités pour placer les pièces sur le plateau. On peut ainsi prendre les couleurs qui ont le moins grand chiffre (de pièces restantes avec la dite couleur). On aura donc le moins de choix de pièce hors-jeu à associer avec une pièce sur le plateau. Et ainsi, diminuer les probabilités d'erreurs.