

TP : Introduction à l'Intelligence Artificielle

2020/2021

Objectifs

L'objectif de ces travaux pratiques est de générer et de résoudre de manière optimale tous les taquins 3x3 (et 4x4!).

Résoudre un taquin (dit initial) revient à trouver le nombre de déplacements *minimal*, de la case vide, nécessaire pour passer du taquin initial vers un taquin but (décrit ci-dessous). Chaque déplacement, de la case vide, coûte une unité.

Le but de ces travaux pratique est aussi de comparer les performances de plusieurs algorithmes et tester expérimentalement l'admissibilité de fonctions heuristiques.

La réalisation de ces premier travaux pratiques est une étape nécessaire pour résoudre efficacement et pleinement le problème du taquin 4x4 en utilisant l'algorithme IDA* avec des heuristiques avancées.

Langages de programmation

Il est préférable d'utiliser le langage C (le langage Python est accepté).

Taquin but

Nous utiliserons le taquin but suivant :

Table 1: Exemple d'un taquin 3x3 but

	1	2
3	4	5
6	7	8

Pour ce TP, une implémentation des algorithmes de recherches à base d'heuristiques est demandée

D'abord, il est important d'implémenter les fonctions de pré-traitements (certaines déjà données dans les TPs précédents) :

1. Ecrire une fonction qui vérifie si un taquin 3x3 est un taquin but ou non.
2. Ecrire une fonction qui vérifie si un taquin 3x3 est solvable ou non.
3. Ecrire une fonction qui prend un taquin initial 3x3 en paramètre et calcule ses taquins successeurs. Un taquin successeur est obtenu depuis le taquin initial en déplaçant (quand cela est possible) la case vide vers la gauche, le haut, la droite, le bas respectivement.

4. Ecrire une fonction, appelée `h_malplacees` (on la notera dans la suite h_1), qui prend un taquin initial 3x3 en paramètre et calcule le nombre de cases mal placées (qui ne sont pas à la bonne position dans le taquin final).
5. Ecrire une fonction, appelée `h_manhattan` (on la notera dans la suite h_2), qui prend un taquin initial 3x3 en paramètre et calcule l'heuristique basée sur la distance de Manhattan.

Pour les algorithmes de recherche informés, il est demandé dans ce TP la programmation de l'algorithme de recherche A^* (et IDA^* si le temps le permet). On suppose qu'un déplacement de la case "blanc" coûte une seule unité.

On utilisera les deux heuristiques suivantes :

- h_1 : Nombre de cases qui ne sont pas dans leurs positions initiales.
- h_2 : La distance de Manhattan.

Pour l'évaluation de vos algorithmes, générer un échantillon de taquins initiaux. La sortie de chaque algorithme est un fichier où chaque ligne est composée de cinq champs (une ligne = résultats obtenus sur un taquin initial de l'échantillon généré) :

- le premier champs contient le taquin initial (les 9 chiffres du taquins, ligne par ligne. Le vide sera codé par le chiffre 0)
- le deuxième champs contient le nombre de déplacements nécessaires pour la résolution du taquin, et
- le troisième champs contient une chaîne de caractères, composée de l'un des quatre caractères : "G" (Gauche), "D" (Droite), "H" (Haut) et "B" (Bas). Cette chaîne de caractères représente la séquence des déplacements de la case vide de l'état initial vers l'état but. Cette séquence permet de vérifier que votre solution est correcte (voir TP précédent).

Remarques :

- Il est important de réfléchir à une bonne structure de données et à l'algorithmique avant de se lancer dans la programmation.
- Les travaux pratiques doivent-être réalisés par vous-même (il n'est pas autorisé de reprendre un code existant; peu importe sa provenance).