Bordeaux INP
ENSC

Projet Prolog : Création d'un jeu de réflexion

-

Rush Hour

2017-2018

Rapport de projet

Développeurs	Encadrants
Sylvain Duhau-Marmon	Baptiste Pesquet
Samuel Jollois	Pierre-Alexandre Favier
Valentin Mancier	

I. Organisation du code

Le code est constitué de 5 fichiers Prolog :

- '**Duhau-Marmon_Jollois_Mancier_affichage.pl**' qui contient toutes les fonctions permettant l'affichage et les interactions avec le board sous forme graphique.
  NB : L'affichage est optimisé pour un MacBook Air.
- '**Duhau-Marmon_Jollois_Mancier_bfs.pl**' qui contient l'affichage en console du jeu, ainsi que les fonctions permettant la résolution du jeu par l'algorithme du parcours d'arbre en largeur.
- '**Duhau-Marmon_Jollois_Mancier_astar.pl**' qui contient l'affichage en console du jeu, ainsi que les fonctions permettant la résolution du jeu par l'algorithme A*.
- Un dossier '**old**' qui contient l'ancienne représentation abstraite du Rush Hour et l'ancienne version de l'affichage graphique qui utilisait cette représentation.

II. Justification des choix

L'affichage sous forme graphique est rendu possible grâce à la librairie XPCE. Elle améliore grandement l'expérience utilisateur du joueur en permettant une meilleure visualisation du board et en rendant les commandes de jeu plus intuitives.

Les prédicats suivants sont définis comme dynamiques car ils sont amenés à être altérés durant le déroulement du jeu à l'aide les fonctions natives "assert" et "retract" :

- board
- parent
- possible
- score
- vu
- partieEnCours
- scoreMini

Le board est défini comme étant une liste de 36 entiers en ligne représentant chacun le contenu d'une case. Soit il s'agit de 0, dans ce cas il n'y a pas de véhicule sur cette case. Soit il s'agit de 1, dans ce cas il s'agit du véhicule que l'on doit sortir de la grille. Sinon l'entier correspond au numéro du véhicule sur cette case.

En ce qui concerne les véhicules :

- un véhicule est défini par un entier car cela permet d'ajouter autant de véhicules que l'on veut à notre niveau, mais aussi de s'affranchir des contraintes liées aux couleurs.
- l'orientation est horizontale si un véhicule apparaît 2 fois à la suite dans la liste du board, sinon l'orientation est verticale.
- la position du véhicule est la première case occupée par la voiture sur le board en partant du coin en haut à gauche de la grille.

- la longueur d'un véhicule est déterminé en comptant combien de fois l'entier désignant ce véhicule est présent sur le board.

III. Prédicats

A. mouvement

Paramètres :

- **X** : coordonnée X d'une case de la grille
- **Y** : coordonnée Y d'une case de la grille
- **Direction** : direction dans laquelle se dirige le véhicule
- **Board** : board actuellement affiché
- **NvBoard** : board créé après le déplacement du véhicule

Déplacer un véhicule implique que la case vers laquelle il se déplace ne soit pas occupée et que le mouvement soit initié par un véhicule. Il faut de plus déterminer si le véhicule possède la bonne orientation et si c'est un camion ou une voiture, pour cela on utilise le prédicat longueur.
On remplace enfin la case sur laquelle le véhicule se déplace par l'entier correspondant au véhicule et la case qu'il quitte par 0 sur board courant pour obtenir le nouveau board.

B. possible/2

Paramètres :

- **X** : coordonnée X d'une case de la grille
- **Y** : coordonnée Y d'une case de la grille

Il parcourt toutes les cases du board. Si le coup est possible, il crée un prédicat possible/4 qui est ajouté à la liste des coups faisables puis continue le parcours à partir d'une position qui n'a pas été explorée.

Les autres prédicats, plus explicites, sont commentés dans les fichiers source.

IV. Intelligence Artificielle

Au cours de ce projet, nous avons pu implémenter 3 algorithmes de parcours d'arbre qui permettent de trouver une liste de coups gagnants.

L'algorithme de parcours d'arbre en largeur explore tout l'espace des positions et trouve obligatoirement la solution optimale si celle-ci existe. Afin d'utiliser cet algorithme, il nous a fallu alléger le coût en mémoire d'une position pour rendre l'exploration faisable et rapide. Le nombre de prédicats associés à une position était trop important dans la représentation contenue dans le dossier old.

Trouver les coups possibles à partir d'une position donnée est la tâche la plus cruciale dans l'exécution de cet algorithme, cependant il y a de nombreuses subtilités en jeu, comme pour ne pas explorer une position déjà rencontrée, ou encore garder une trace des coups joués.

Pour prioriser certaines positions, il nous a fallu mettre en place une distance heuristique à l'objectif. Celle-ci est la somme de la distance de la voiture rouge à la case finale et du nombre de véhicules bloquant le chemin de cette voiture.

L'algorithme Greedy best search renvoie des solutions non-optimales mais avec significativement moins de noeuds ouverts. Il tient seulement compte de la distance heuristique. Son exécution nous permet de juger si la distance heuristique est pertinente.

L'algorithme A* renvoie des solutions optimales avec, selon les positions, jusqu'à 60% moins de noeuds explorés qu'en largeur. Il tient compte de la distance heuristique, mais également du nombre de coups joués pour atteindre une position donnée.
Notons que le coût - temporel et en mémoire - du tri des positions est supérieur à l'amélioration apportée en terme d'exploration.

En conséquence, avons choisi d'utiliser l'algorithme de parcours en largeur pour déterminer le chemin optimal à partir de la solution courante.

A. Parcours d'arbre en largeur

Niveau 1 : 16 coups, 600 ms, 1072 noeuds
Niveau 2 : 22 coups, 210 ms, 435 noeuds
Niveau 3 : 18 coups, 1370 ms, 2332 noeuds
Niveau 4 : 21 coups,4200 ms, 5212 noeuds
Niveau 5 : 50 coups, 140 ms, 257 noeuds
Niveau 6 : 42 coups, 5200 ms, 6483 noeuds
Niveau 7 : 32 coups, 12850 ms, 10756 noeuds
Niveau 8 : 34 coups, 22300 ms, 15129 noeuds
Niveau 9 : 71 coups, 2355 ms, 4168 noeuds
Niveau 10 : 93 coups, 35000 ms, 19480 noeuds

B. Greedy best search

Niveau 1 : 16 coups, 407 ms, 452 noeuds
Niveau 2 : 22 coups, 78 ms, 162 noeuds
Niveau 3 : 18 coups, 3500 ms, 1896 noeuds
Niveau 4 : 24 coups, 9370 ms, 1928 noeuds
Niveau 5 : 54 coups, 63 ms, 226 noeuds
Niveau 6 : [out of local stack]
Niveau 7 : [out of local stack]
Niveau 8 : [out of local stack]
Niveau 9 : [out of local stack]

Niveau 10 : [out of local stack]

C. A*

Niveau 1 : 16 coups, 718 ms, 1043 noeuds
Niveau 2 : 22 coups, 110 ms, 378 noeuds
Niveau 3 : 18 coups, 750 ms, 922 noeuds
Niveau 4 : 21 coups, 3828 ms, 2207 noeuds
Niveau 5 : 50 coups, 94 ms, 246 noeuds
Niveau 6 : [out of local stack]
Niveau 7 : [out of local stack]
Niveau 8 : [out of local stack]
Niveau 9 : [out of local stack]
Niveau 10 : [out of local stack]