Projet IA
Pathfinding

Groupe : Taleb Rida, Gabour Smail, Jday Achraf

Encadrant de TD : Parham Shams

Introduction

Dans le cadre du projet d’intelligence artificielle, nous disposons d’un ensemble d’agents qui doivent chacun atteindre une destination qui leurs est propre. On cherchera à trouver un ensemble de chemins, sans collision, qui permette à chaque agent d’atteindre sa destination.

L'objet de ce projet est d'étudier une version adversariale du cooperative path-finding : plusieurs équipes d'agents sont en compétition et cherchent à atteindre leurs destinations avant leurs adversaires. On se limitera à deux équipes dans ce projet. Afin d'éviter les situations de blocage, on fixera à l'avance une limite sur le nombre de tours du jeu. On constate qu’au niveau global, on peut donc voir ce problème comme un jeu à 2 joueurs dans lequel chaque équipe est un meta-joueur. En effet, les actions au sein de chaque équipe peuvent éventuellement être coordonnées.

Les déplacements des équipes obéissent à quelques règles précises :

- Les tours alternent entre équipes : les agents de l'équipe A peuvent tous bouger, puis ceux de l'équipe B, etc.
- Les agents se déplacent sur une case adjacente ou restent au même endroit.
- Il faut éviter toute collision : un agent peut se déplacer vers une case si celle-ci est libre ou libérée dans le même tour par un joueur de la même équipe. Une exception est faite pour les échanges de positions entre deux agents, qui n'est pas permise. (Cela supposerait que les agents se croisent).

Comme critère de victoire, on considère :

- L’équipe qui a réussi à placer le plus d'agents à leur destination, et en cas d'égalité,
- L’équipe dont la somme pour sur les agents de l'équipe des plus courts chemins à leur destination est la plus faible.

Nous allons tout d’abord présenter les différentes stratégies que nous avons mis en place, puis nous verrons les deux manières de gérer les collisions que nous avons tester durant ce projet. Et enfin, nous présenterons les résultats obtenus.

Nous avons implémenté 2 algorithmes de recherche en plus de celui qui nous a était donner (c.f astar) pour effectuer nos tests :

- L'algorithme de Greedy_Best_First sélectionne toujours le chemin qui semble le meilleur à ce moment-là. Il s'agit de la combinaison des algorithmes de recherche en profondeur et en largeur. Il utilise la fonction heuristique et la recherche. La recherche best-first nous permet de profiter des avantages des deux algorithmes. Avec l'aide de la recherche du meilleur premier, à chaque étape, nous pouvons choisir le nœud le plus prometteur. Dans l'algorithme de la meilleure première recherche, nous développons le nœud le plus proche du nœud cible et le coût le plus proche est estimé par la fonction heuristique.

- L'algorithme Random_First consiste à reprendre le même déroulement que l'algorithme Greedy_Best_First précédemment mentionné, sauf qu'au lieu de récupérer le nœud le plus proche du nœud cible, nous allons choisir un nœud de manière aléatoire.

Les 3 algorithmes ont été utilisé dans les trois stratégies que nous avons implémentées.

I. Stratégie préparation coopérative avec recalcule de chemins à chaque collision :

Objectif : Les objectifs sont répartis entre les deux équipes de manière aléatoire. Chaque équipe calculera pour chacun de joueurs la distance qu'il aura à parcourir pour chaque objectif, et ainsi assignées les objectifs à chaque joueur de tel sorte que la distance globale parcouru par l'équipe soit la plus courte.

Gestion collision : A chaque itération le joueur vérifie s'il n'y pas d'autre joueurs dans la prochaine case sur laquelle il passera. Dans le cas contraire, celui-ci recalculera un itinéraire en considérant momentanément les autres joueurs comme des murs.

II. Stratégie préparation coopérative avec recalcule de chemins à chaque itération :

Objectif : Les objectifs sont répartis entre les deux équipes de manière aléatoire. Chaque équipe calculera pour chacun de joueurs la distance qu'il aura à parcourir pour chaque objectif, et ainsi assignées les objectifs à chaque joueur de tel sorte que la distance globale parcouru par l'équipe soit la plus courte.

Gestion collision : A chaque itération le joueur recalcule son itinéraire en considérant, ici aussi, momentanément les autres joueurs comme des murs.

III. Stratégie sans préparation coopérative :

Objectif : Les objectifs sont répartis entre les deux équipes de manière aléatoire. Et chaque équipe attribue de manière aléatoire les objectifs à ses joueurs.

Gestion collision : A chaque fois qu'un joueur j rencontre un joueur k tel que (j != k), et qu'il ont la même case en commun à la prochaine itération, dans ce cas là le joueur j considère sa prochaine case momentanément comme un mur et recalcul son itinéraire. Dans le cas où le joueur k est immobile, le joueur j considère que l'emplacement du joueur k est un mur et ce jusqu'à la fin de la partie.

RESULTATS

Les deux premières stratégies seront testées sur un même seed.

Stratégie préparation coopérative avec recalcule de chemins à chaque collision :

Pour l'algorithme de recherche A* :

- ScoreOBJECTIF équipe 1 : 3
- ScoreOBJECTIF équipe 2 : 3
- ScoreCHEMIN équipe 1 : 53
- ScoreCHEMIN équipe 2 : 43

L'équipe 1 & 2 ont bien tous atteint leurs objectifs avant les n itérations.
L'équipe 2 est gagnante car l'équipe qui a le score de chemin le plus court (nombre de case parcouru au total par l'équipe).

Pour l'algorithme de recherche Greedy_Best_First :

- ScoreOBJECTIF équipe 1 : 3
- ScoreOBJECTIF équipe 2 : 3
- ScoreCHEMIN équipe 1 : 49
- ScoreCHEMIN équipe 2 : 55

L'équipe 1 & 2 ont bien tous atteint leurs objectifs avant les n itérations.
L'équipe 1 est gagnante car l'équipe qui a le score de chemin le plus court (nombre de case parcouru au total par l'équipe).

Remaque : Le total de case parcouru pour l'algorithme de recherche A* est inférieur à celui du Greedy_Best_First (96 <104). On peut en déduire que A* est plus efficace dans cette stratégie.

Stratégie préparation coopérative avec recalcule de chemins à chaque itération :

Pour l'algorithme de recherche A* :

- ScoreOBJECTIF équipe 1 : 3
- ScoreOBJECTIF équipe 2 : 3
- ScoreCHEMIN équipe 1 : 50
- ScoreCHEMIN équipe 2 : 44

L’équipe 1 & 2 ont bien tous atteint leurs objectifs avant les n itérations.
L’équipe 2 est gagnante car l’équipe qui a le score de chemin le plus court (nombre de case parcouru au total par l’équipe).

Pour l’algorithme de recherche Greedy_Best_First :

- ScoreOBJECTIF équipe 1 : 3
- ScoreOBJECTIF équipe 2 : 3
- ScoreCHEMIN équipe 1 : 44
- ScoreCHEMIN équipe 2 : 38

L’équipe 1 & 2 ont bien tous atteint leurs objectifs avant les n itérations.
L’équipe 2 est gagnante car l’équipe qui a le score de chemin le plus court (nombre de case parcouru au total par l’équipe).

Remaque : Le total de case parcouru pour l’algorithme de recherche A* est supérieur à celui du Greedy_Best_First (94 >82). On peut en déduire que Greedy_Best_First est plus efficace dans cette stratégie.

Stratégie sans préparation coopérative :

Dans cette stratégie nous allons comparer deux équipes avec des algorithmes de recherche différents sans seed. Nous allons réaliser une dizaine de confrontation entre chaque algorithme.

Pour 10 parties nous avons obtenue :

A* vs Greedy_Best_First :

A* : 7 victoires avec une moyenne de 57.5 déplacements par partie.

Greedy_Best_First : 3 victoires avec une moyenne de 58.9 déplacements par partie.

On remarque ici que A* en moyenne gagne plus souvent sur 10 parties.

A* vs Random_First :

A* : 4 victoires avec une moyenne de 61.5 déplacements sur 10 parties

Random_First : 6 victoires en moyenne avec une moyenne de 55.4 déplacements par partie.

On remarque ici que Random_First en moyenne gagne plus souvent sur 10 parties.

Greedy_Best_First vs Random_First :

> Greedy_Best_First : 7 victoires avec une moyenne de 59.6 déplacements par partie.
>
> Random_First : 3 victoires avec une moyenne de 63.5 déplacements par partie.

On remarque ici que Greedy_Best_First en moyenne gagne plus souvent sur 10 parties.

Conclusion

Nous avons à travers ces tests pu mettre en évidence certains points intéressant, notamment le fait que la seconde stratégie a certes un plus long temps d'exécution mais semble donner un parcours total des joueurs des équipes inférieur à la première stratégie. La dernière stratégie nous a permis de mettre en avant l'efficacité de chaque algorithme de recherche dans le cadre d'un « versus ». Nous avons pu remarquer que l'équipe utilisant A* gagnera plus souvent face à l'équipe utilisant Greedy_Best_First, tandis que l'équipe Greedy_Best_First gagnera plus souvent face à l'équipe Random_First, mais celle-ci gagnera plus souvent face à l'équipe utilisant A*.