Rapport de Mini-Projet 3I025

Recherche de chemins coopératifs entre agents

Castellon Clément Zhang Noé

25 mars 2019

Première partie

introduction

1 Projet

Nous considérons plusieurs agents en compétition qui cherchent chacun à atteindre leur propre objectif, nous souhaitons éviter qu'il y ait des collisions entre eux. Nous cherchons à maximiser la coopération entre les agents et ainsi minimiser le temps total nécessaire à ce que chacun atteigne son objectif. Pour cela nous étudierons trois approches dont nous détaillerons les spécificités dans ce compte-rendu.

2 Notions de base

2.1 Agents Intelligents

Un agent intelligent est une entité autonome qui peut-être caractérisée par plusieurs caractéristiques, celui-ci doit être capable de s'adapter à son environnement, aussi il doit pouvoir être configurable afin de répondre aux besoins précis de l'utilisateur. De plus, dans un cadre plus avancé, il doit savoir profiter de ses expériences passées afin de mieux comprendre les souhaits de l'utilisateur. Lors du déploiement de plusieurs agents la communication et l'interaction entre eux est primordiale afin de maximiser la coopération.

2.2 Collisions

Dans notre problème de cheminements, il existe 2 types de collisions différentes entre les agents, la première a lieu lorsque deux agents différents veulent accéder à la même case au même instant, la seconde à lieu lorsque ceux-ci sont face à face et vont se croiser, la prochaine position des deux agents est la position courante de ceux-cis.

2.3 Solutions

Afin d'empêcher les différentes collisions qui pourrait arriver, différentes solutions peuvent être implémentées afin que les agents puissent réagir de manière autonome. Trois manières différentes vont être présentées.

Deuxième partie
Stratégies

3 Path-Splicing

3.1 Idée globale

Cette première implémentation consiste à modifier le chemin emprunté par un agent seulement lorsque celui-ci détecte une collision c'est-à-dire à l'instant précédent la collision, en effet lorsqu'un agent détecte une collision avec un autre, il va considérer le lieu de collision comme un obstacle et va calculer en conséquence un nouveau chemin qui lui permettra d'atteindre un état postérieur de son chemin en tenant compte du lieu de collision.

3.2 Implémentation

Dans un premier temps chaque agent calcule indépendamment des autres un chemin lui permettant d'atteindre son objectif, pour cela nous avons utilisé l'algorithme A* qui permet de calculer le plus court chemin entre deux points d'un espace dimensionné. Une fois les chemin calculés, nous les gardons en mémoire et nous associons un compteur à chaque agent qui correspond à l'étape où se trouve celui-ci dans son chemin associé. Lorsqu'une collision est détecté, l'agent qui détecte la collision va recalculer un chemin qui l'amènera à la position suivant la collision tout en considérant, le lieux de collision comme un obstacle, si état final alors random move puis recalcule.

3.3 Résultat

Graphique à mettre

4 Coopérative Basique

4.1 Idée globale

Pour cette stratégie, les agents se déplacent par groupes formés à partir des chemins qui ne produisent pas de collisions. Les groupes sont alors executés en différés. Différentes stratégies peuvent être implémentées afin de définir l'ordre de passage.

4.2 Implémentation

Pour chaque agent, un chemin est calculé à partir de l'algorithme A étoile sans tenir compte des autres agents. On forme ensuite des groupes d'agents en fonction des chemins qui ne forment pas de collisions s'ils sont exécutés en même temps. L'ordre de passage des groupes est alors organisé grâce à une stratégie choisie, la stratégie de base consistant à prendre le premier groupe formé et ainsi de suite. Lorsque tout les agents d'un groupe ont atteint leur objectif, on recalcule un chemin vers leur nouvel objectif et on leur associe un groupe à partir de la fin de la file. Enfin, avant d'envoyer le prochain groupe on va recalculer leur chemin en tenant compte de la position courante des agents des autres groupes. Si deux nouveaux chemins forme produisent une collision on en enlève un des deux et on envoie le groupe de chemin.

Ces choix d'implémentation ne sont pas optimaux du point de vue de la performance, car si deux chemins ne se croisent que sur une case, sans garantie qu'ils ne se croisent à un instant donné, par précaution on va quand même ne pas les faire exécuter leurs chemins respectifs simultanément, ce qui rend la résolution plus longue que dans les autres implémentations. Cependant on a pas à gérer les collisions entre agents, et donc la complexité calculatoire est la meilleure.

4.3 Résultat

Graphique à mettre Il s'agit de l'implémentation la plus triviale des trois, facile à implémenter, mais qui cependant est la moins performante.

5 Coopérative avancée

5.1 Idée globale

On souhaite utiliser une structure de données partagée qui correspond à une table de réservation où l'on va stocker des triplets qui correspondent à la position à un instant donné, si une case est réservée, les autres agents ne pourront pas y accéder, évitant ainsi les collisions.

5.2 Implémentation

Le chemin de chaque agent prend en compte celui des agents précédents, en effet, pour un agent donné on calcule un chemin partiel d'un nombre de case constant qui dirige l'agent vers son objectif. A différence des méthodes précédentes, on utilise comme heuristique la vraie distance, il s'agit de la distance minimale qui sépare l'agent de sa destination. Afin de la calculer on utilise l'algorithme A étoile et on calcule le chemin en sens inverse on obtient alors la vrai distance en regardant la valeur g dans les noeud fermés.

5.3 Résultat

Graphique à mettre

6 Comparaison

Troisième partie
Conclusion