Rapport FoSyMa

Master 1 ANDROIDE
Arthur Esquerre-Pourtère, Jerome Arjonilla

2020

Introduction

Ce rapport a pour but de présenter les différents choix techniques mis en place pour le projet de l'UE "FoSyMa".

Pour rappel, l'objectif du projet est d'essayer de capturer des ennemis ("Wumpus") grâce à des agents intelligents. Pour cela, nos agents doivent explorer une carte inconnue, patrouiller dans cette carte, détecter et capturer les "Wumpus". De plus, ils ont la possibilité de communiquer entre eux afin de se partager des informations, toutefois leur rayon de communication est limité.

Ce rapport s'effectuera en plusieurs parties. Au départ, nous expliquerons globalement et rapidement les différents mécanismes et comportements. Et dans une seconde partie, nous expliquerons en détails certains points, et algorithmes.

Explication générale

Au départ, nos agents ont pour objectif d'explorer la carte car, ils apparaissent dans une carte inconnue.

Pour réussir leurs explorations, les agents vont en direction des noeuds ouverts les plus proches. De plus, afin de mieux explorer ils peuvent communiquer avec d'autres agents.

Pendant cette communication, les agents échangent les informations qu'ils ont sur la carte, afin de combler leur carte personnelle. Si après cet échange, ils n'ont toujours pas fini de découvrir la carte, alors ils vont s'organiser afin de découvrir les autres points ouverts. Et si, au contraire, ils ont fini d'explorer la carte, alors ils vont calculer des patrouilles et partir en exploration.

Après avoir déterminé sa patrouille, l'agent va l'explorer. Si après plusieurs explorations, il n'observe pas de "Wumpus", alors il va prendre une autre patrouille. Et si, au contraire, il observe un "Wumpus", alors plusieurs choix s'offrent à lui. S'il estime qu'il peut capturer le "Wumpus" seul, alors il essaye de le capturer. Si, au contraire, il estime qu'il a besoin d'aide, alors il va chercher d'autres agents en renfort, et ensemble, ils vont se coordonner pour bloquer le "Wumpus".

Après avoir réussi à bloquer un "Wumpus", l'agent ou les agents, vont envoyer des messages aux autres agents pour les prévenir qu'ils ne sont plus disponibles et que certaines positions ne sont plus disponibles. Ainsi, les autres agents pourront recalculer leur patrouille, en prenant en compte ces nouvelles informations.

Explication précise

Communication

Les agents ont la possibilité de communiquer entre eux pendant toute la durée du processus, et sur de nombreux sujets différents. Pour cela, nous avons créé pour chaque agent différentes variables permettant de définir au mieux les besoins de l'agent.

Ainsi, en fonction des besoins de chaque agents et de leur behaviour courant, ils peuvent communiquer sur les sujets suivant :

- Besoin d'avoir plus d'informations sur la carte
- Besoin d'aide pour explorer les noeuds ouverts
- On signale à un autre agent qu'on a pris sa patrouille
- On souhaite prévénir qu'un agent et des positions ne sont plus disponibles
- On souhaite échanger des noeuds à aller observer pour se débloquer
- On souhaite créer une coalition
- On avertit les autres agents que l'on chasse un "Wumpus"

- On souhaite connaître la position des autres agents pour savoir si on a attrapé un "Wumpus"

À l'initialisation, chaque agent possède le behaviour "StartCommunicationBehaviour". Ce behaviour permet d'envoyer des pings aux autres agents et de recevoir des pings des autres agents, ces "pings" permettent d'indiquer aux autres agents que l'on souhaite démarrer une conversation avec eux. Pour envoyer des pings, il faut qu'au moins une des variables traitant des besoins de l'agent soit "true". Ce behaviour permet aussi de gérer toutes les communications "courtes" entre agents qui ne nécessitent pas de démarrer une conversation.

D'autres behaviours, utilisants "ConversationBehaviour", permettent de gérer les conversations entre agents. Dans ce cas les agents vont arrêter de ce déplacer et s'envoyer des messages contenants les différentes informations à s'échanger. Si les agents ont fini leur conversation, ou qu'ils n'ont pas reçu de messages depuis un certain temps, alors ils vont s'envoyer un message pour s'informer que la conversation est terminée et recommencer à ce déplacer.
De plus, les agents utilisent des timer pour savoir à quand remonte la dernière conversation qu'ils ont eu avec un agent, ceci afin que deux agents ne lancent pas des conversations en boucle, sans ne plus pouvoir se déplacer.

Exploration de la carte

Dans un premier temps, les agents vont explorer la carte en allant sur les noeuds ouverts les plus proches. Ils vont, par la suite, discuter avec d'autres agents (S'ils en croisent) et si, après leur discussion, ils n'ont toujours pas comblé toute la map, alors ils vont se répartir les noeuds ouverts. La répartition des noeuds s'effectue par un seul agent, c'est l'agent qui a reçu la demande d'aide.

À partir des noeuds ouverts, l'agent essaye de déterminer 2 clusters. Pour cela, il part des noeuds ouverts et regarde leurs voisins. Par la suite on note, pour chaque voisin, l'ensemble des noeuds ouverts qu'il peut atteindre en un seul déplacement. Après que ceci soit fait, on refait l'algorithme, en partant des voisins. Ainsi, au prochain tour, on aura pour chaque voisin la liste des noeuds ouverts accessibles en 2 déplacements. On continue l'algorithme jusqu'à ce qu'on obtienne deux ensembles distincts contenants tout les noeuds ouverts.

Après que l'agent ait déterminé les deux clusters, il va distribuer les clusters en fonction de sa position et de la position de l'agent qui a fait la demande. Grâce à cela, les agents n'iront visiter que leur clusters. Et si, en chemin, il rencontre un autre agent, notre agent va refaire l'algorithme en fonction de son cluster de points.

Pour plus de renseignements sur cette partie, il faut regarder les algorithmes des classes "Kmeans" et "DetermineNextPosition"

Recherche du Wumpus

Après avoir fini d'explorer la map, chaque agent calculera sa patrouille et la patrouille des autres agents. Nous avons fait le choix de faire des patrouilles car, en faisant cela, les agents peuvent savoir en permanence où se situent les autres agents. Et ainsi, si un agent trouve un "Wumpus", il pourra facilement aller dans une autre patrouille pour chercher un autre agent. En effet, la méthode pour calculer les patrouilles étant déterministe, chaque agent sait où se situent les patrouilles des autres agents.

Dans un premier temps, pour déterminer les patrouilles, on cherche les n points les plus importants, avec n qui correspond au nombre d'agents. Quand on a déterminé les noeuds les plus importants, on part de ces noeuds, et on cherche le meilleur voisin. On continue l'algorithme jusqu'à ce qu'on puisse observer tous les noeuds.

Pour déterminer le meilleur voisin, on attribue une note en fonction du nombre de voisins de chaque noeud. Ainsi, plus un noeud possède de voisin, et plus ce noeud est important. De plus, afin de déterminer au mieux le meilleur voisin, on prend en compte les futurs déplacements. Ainsi, pour déterminer la note pour un noeud, on fait en fonction de son nombre de voisin, mais aussi en fonction du nombre de voisins de ses voisins. De plus, lors du calcul, on ne prend seulement en compte les voisins qui ne sont pas

observables par une partition d'un agent.

Après avoir déterminé les patrouilles, les agents explorent leur patrouille respective. Et si un agent sent un "Wumpus", il va essayer de le chasser. Et si au contraire, il ne sent rien, il continue d'explorer sa patrouille. Mais après avoir fait l'exploration de sa patrouille plusieurs fois, l'agent va explorer une autre patrouille.

Ainsi, en explorant, une autre patrouille, il peut potentiellement observer que certains agents ne sont plus disponibles car ils ont réussi à capturer un "Wumpus". Si c'est le cas, notre agent, va re-calculer une nouvelle patrouille, en prenant en compte le fait que certains agents ne sont plus disponibles, et que certains points ne le sont plus.

Pour plus de renseignements sur cette partie, il faut regarder les algorithmes de la classe "DeterminePatrouille".

Détection d'un Wumpus et chasse

Option "chasse seul"

Lorsqu'un agent détecte la présence d'un "Wumpus" en le sentant, il va commencer par évaluer, grâce à sa connaissance de la carte, si il peut chasser le "Wumpus" seul (i.e. il va calculer si le sous-graphe dans lequel est le "Wumpus" est un arbre). Dans ce cas, il va se contenter de suivre le "Wumpus" jusqu'à l'avoir bloqué. L'agent considérera qu'il a bloqué le "Wumpus" s'il sent toujours son odeur et qu'après avoir essayé d'avancer n fois sur la même case mais cela sans succès (actuellement n=20). Cette option de "chasse seul" peut être activée/désactivée facilement dans le code car elle se base sur les hypothèses, fortes, que le "Wumpus" se déplace à la même vitesse, ou moins rapidement, que l'agent et qu'il peut être senti dans un rayon de 1 exactement.

Création de coalition

Si l'agent évalue qu'il a besoin d'autres agents pour s'occuper du "Wumpus", ou que l'option "chasse seul" est désactivée, il ira chercher k autres agents (Actuellement k=2. Si on connaissait le nombre de "Wumpus" à l'avance on pourrait déterminer la valeur de k afin que les agents se partagent les "Wumpus" à chasser). Pour aller chercher les autres agents, l'agent qui a détecté le "Wumpus" va se rendre aux centres de leur patrouille. Si l'agent rencontre un autre agent, il va l'inviter à rejoindre sa coalition pour aller chasser le "Wumpus" ensemble et, en fonction de leur positions, l'agent ayant lancé la conversation partagera les differentes tâches : aller chasser le "Wumpus" ou aller chercher d'autres agents en plus. Si l'agent attend trop longtemps dans le centre d'une patrouille sans rencontrer un autre agent, celui-ci va considérer que l'agent qui est censé s'occuper de cette patrouille est actuellement occupé et il va aller au centre de la prochaine patrouille. L'interêt de se rendre aux centres des patrouilles est que les agents passent souvent par ce centre, cela permet donc de les trouver plus rapidement.

Chasse en groupe

Lorsqu'un agent part chercher le "Wumpus", étant donné que celui-ci s'est probablement un peu déplacé, l'agent va commencer par chercher ce "Wumpus" près de l'endroit où il a été senti pour la dernière fois. Lorsqu'un agent va sentir ce "Wumpus", il va le suivre tout en envoyant des messages contenants sa position. En plus de la position de l'agent, ce message contiendra aussi un compteur. Ainsi les agents aux alentours, s'ils ne sont pas encore en train de chasser pourront se mettre à chasser aussi, et si ils sont déjà en train de chasser alors, ils pourront s'aider de la position de l'agent pour savoir où aller. En effet, afin de se coordonner pour attraper le "Wumpus", les agents vont à la case où ils sentent le "Wumpus" qui est la plus éloignée des autres agents. Ce qui a pour effet d'entourer le "Wumpus", ceci fonctionne très bien à condition que le rayon d'odeur soit de un. Le compteur présent dans les messages est utile dans le cas où les messages ne sont pas délivrés dans l'ordre, ainsi si l'on reçoit un message de la part d'un agent qui nous avait déjà envoyé une position plus récente, on peut ignorer ce message.

Détection du blocage d'un Wumpus

Lorsqu'un agent pense avoir attrapé un "Wumpus" pendant une chasse en groupe (i.e. l'agent sent toujours l'odeur et n'a pas réussi à d'avancer n fois sur la même case), celui-ci va envoyer un message aux agents situés autour de lui afin de leur demander leur position, ceci afin de savoir si la case sur laquelle il n'arrive pas à avancer est occupée, ou non, par un autre agent. Seul les agents immobilisés vont répondre à ce message, ceci

afin que des agents encore en train de ce déplacer n'induisent pas en erreur les autres agents. Si l'un des agents se trouve sur cette case, alors notre agent s'en va chercher le "Wumpus" ailleurs. Sinon, si aucun agent n'a répondu qu'il se trouvait sur cette case au bout de quelques secondes, alors notre agent considère que c'est bien un "Wumpus" qui se trouve sur cette case. Dans ce cas il arrêtera de bouger, et préviendra les autres agents qui passent près de lui qu'il a attrapé un "Wumpus", en partageant sa position et celle du "Wumpus".
Cette méthode est très efficace quand un grand nombre d'agents entourent un "Wumpus", ainsi les agents qui ne sont pas nécessaires au blocage (c'est à dire ceux qui ne sont pas placés juste à côté), peuvent partir chasser un autre "Wumpus".

Conclusion

Nous avons beaucoup apprécié programmer ce projet. En effet, ceci est notre premier projet dans le domaine des systèmes multi-agents. Et ainsi, nous avons pu voir et appréhender de nombreux concepts, et notamment certaines difficultés que peuvent engendrer les systèmes multi-agents, en particulier au niveau des communications et des comportements asynchrones.

Il y a toutefois des points dans notre travail qui pourraient être améliorés, en effet dans certaines situations il est possible que nos agents n'arrivent pas, ou aient beaucoup de mal, à capturer le "Wumpus".

Par exemple, si un "Wumpus" avait la possibilité de déterminer s'il a été repéré, et fuyait donc à un autre endroit de la carte. Dans cette situation nos agents perdraient sa trace en allant chercher des renforts et il aurait sûrement été préférable de suivre le "Wumpus" sans aller chercher de renforts.

De plus, nos algorithmes ont été conçus pour fonctionner de manière optimale dans le cas où le rayon d'odeur du "Wumpus" vaut un. Si ce n'est pas le cas, nos agents risquent de ne pas réussir à attraper ce "Wumpus". On pourrait améliorer nos agents pour les rendre plus robustes à ce type de situations.