
# *Systèmes multi-agents*

Problème du voyageur de commerce

Jérémy BEZAMAT
Alexis JUVEN

**Bordeaux INP - Robotique**

16 novembre 2018

**Encadrement** : Pierre-Alexandre FAVIER

# Table des matières

# Table des figures

# Introduction

Ce projet a été réalisé dans le cadre du cours **Systèmes multi-agents** de troisième année de l'ENSEIRB-MATMECA en spécialisation robotique. Cet enseignement présente le concept d'agent autonome, les différents types d'agents, réactifs ou délibératifs, ainsi que les différents types d'architecture multi-agents.

Pour ce projet, nous devions proposer une simulation par modification d'un modèle d'architecture multi-agents existant ou par création d'un modèle original à l'aide de l'environnement NetLogo. Nous avons choisi d'essayer d'adapter le problème du voyageur de commerce à une architecture multi-agents. Ce problème NP-complet consiste à trouver le plus court chemin passant par tous les sommets d'un graphe et revenant à son point de départ. Notre objectif fut de déterminer si une approche agent permet de résoudre un tel problème.

Nous verrons dans un premier temps la description du modèle ainsi que les choix de modélisation. Puis nous ferons une analyse des résultats obtenus.

# Chapitre 1

# Modèle

Nous avons réfléchi à un modèle multi-agents pour résoudre le problème du voyageur de commerce.

## 1.1 Description du modèle

Ici sont décris les agents présents dans le modèle.

### 1.1.1 Graphe

Nous cherchons à résoudre le problème du voyageur de commerce sur un graphe représenté par des agents de type noeud, placés aléatoirement sur l'écran. Chaque noeud est lié aux autres par des liens. Chaque lien stocke sa longueur, ainsi qu'une valeur correspondant à une quantité de phénomènes déposée sur l'arrête. Une noeud, marqué en rouge, correspond à la fourmilière, d'où sortent les fourmis.

### 1.1.2 Fourmis

Le dernier type d'agent du modèle représente les fourmis parcourant le graphe. Chaque fourmi, cherchant à parcourir le graphe sans passer deux fois par le même sommet, garde en mémoire le chemin et la distance qu'elle a parcouru depuis sa sortie de la fourmilière.

### 1.1.3 Affichage

Afin de visualiser le chemin favorisé par les fourmis, les arrêtes sont affichées avec des épaisseurs différentes, dépendant de la proportion de phéromones déposées sur chacune d'elle.

## 1.2 Simulation

Cette section décris le fonctionnement de la simulation.

### 1.2.1 Sélection des arrêtes à parcourir

Initialement, un ensemble de fourmi part de la fourmilière. Une fourmi se déplace en choisissant au hasard un noeud, parmi l'ensemble des noeud qu'elle n'a pas encore

visité. Bien que ce choix soit réalisé au hasard, une arrête a une chance d'être choisie proportionnelle à la quantité de phéromones déposées.

### 1.2.2 Retour à la fourmilière

Une fois chaque sommet visité, une fourmi rentre à la fourmilière, et dépose sur chaque arrête du chemin qu'elle a parcouru, une quantité de phéromones, correspondant à l'inverse de la distance de la boucle effectuée. De cette manière, les arrêtes d'une boucle de petite taille augmentent leurs chances d'être choisies lors des prochains parcours de fourmis.

### 1.2.3 Évaporation des phéromones

Petit à petit, les phéromones déposées sur chaque arrêtes s'évaporent. Ainsi, chaque arrête perd un certain pourcentage de sa quantité de phéromones à chaque tour complet de fourmi, ce qui rend de moins en moins attractive toute arrête qui n'a pas été utilisée depuis longtemps.

# Chapitre 2

# Analyse des résultats

Une fois la modélisation du problème fini, nous avons essayé de voir l'influence de différents paramètre sur la simulation. Notamment l'influence du nombre de noeuds et du nombre de fourmis.

## 2.1 Influence du nombre de noeuds

Le nombre de noeuds est l'un des paramètres les plus important de notre simulation. En effet, après de nombreux test, nous avons remarqué qu'avec un nombre trop important de noeuds, l'algorithme ne semblait pas converger. Nous avons par exemple essayé avec 15 noeuds et 7 fourmis et après 105045 ticks 2.1

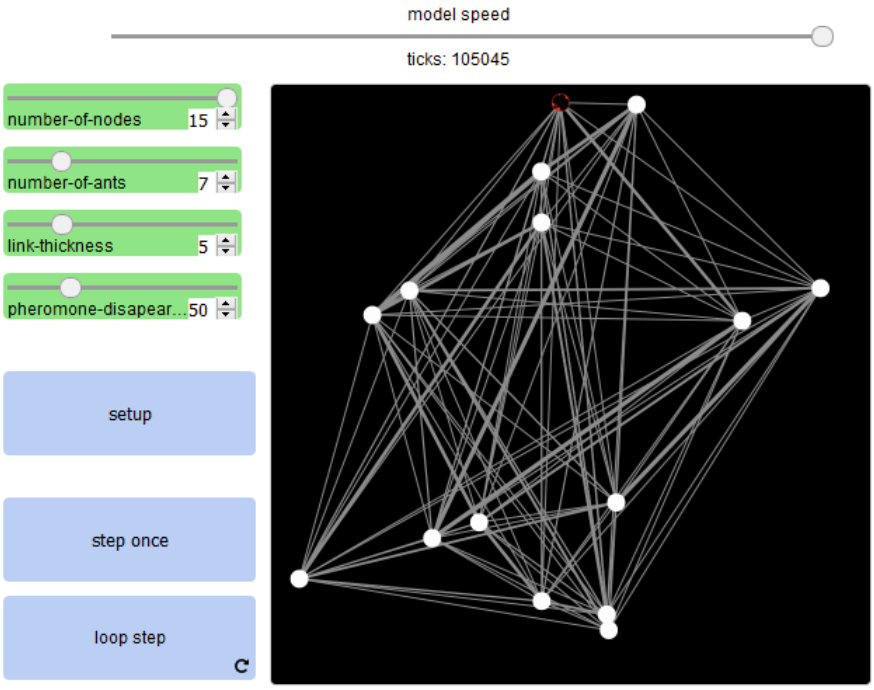



FIGURE 2.1 – Exemple après 105045 itérations 15 noeuds et 7 fourmis

Cependant, pour une nombre plus faible de noeuds, l'algorithme converge vers une solution qui semble effectivement optimale 2.2.

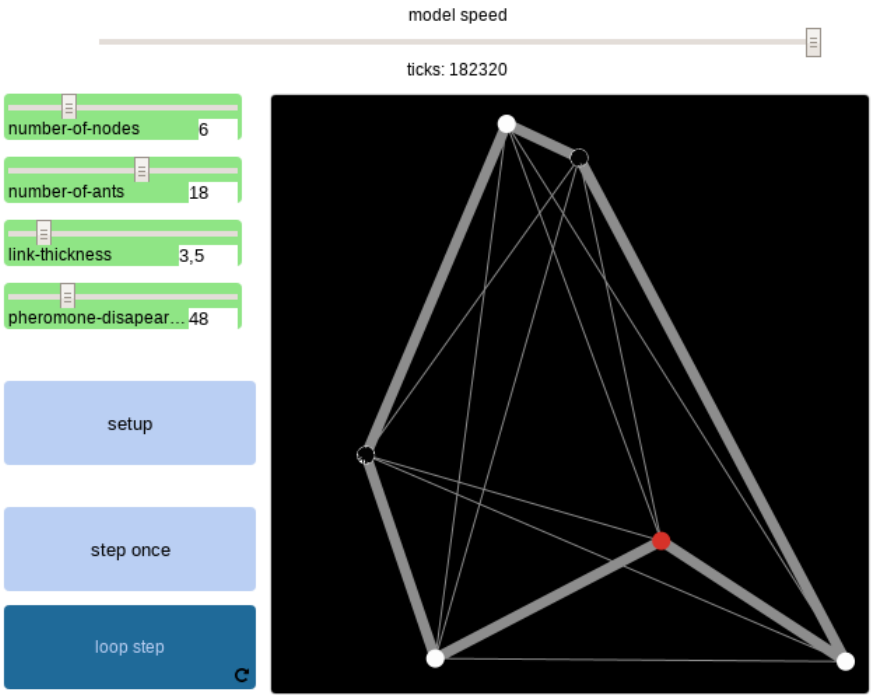



FIGURE 2.2 – Convergence pour 6 noeuds et 18 fourmis

Ils faut quand même prendre ces résultats avec des pincettes car il existe une part d'aléatoire. En effet, la position des noeuds et donc les distances entre chaque noeud est défini aléatoirement. Vous pouvez retrouver plus d'exemple en annexe A.

Le nombre d'itérations à faire avant d'obtenir une convergence semble croître de façon exponentiel en fonction du nombre de noeuds 2.3.

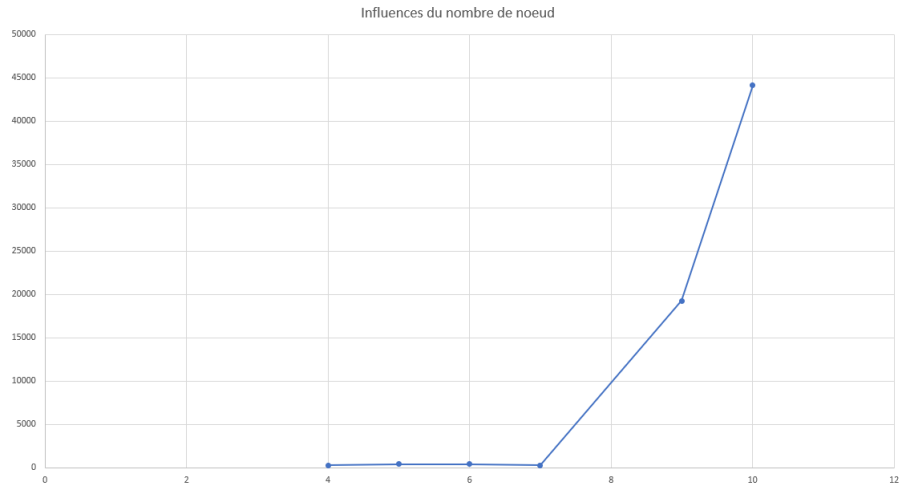



FIGURE 2.3 – Influence des noeuds sur le nombre d'itérations

## 2.2 Influence du nombre de fourmis

Le nombre de fourmis qui parcours les noeuds semble également important dans la convergence de l'algorithme. En effet, d'après nos tests, il semble qu'un nombre trop important de fourmis ralenti la convergence de la simulation. Par exemple, pour un graph comportant 5 noeuds, il faut 49 itérations avec 2 fourmis (2.4) contre 5153 avec 25 fourmis (2.5).

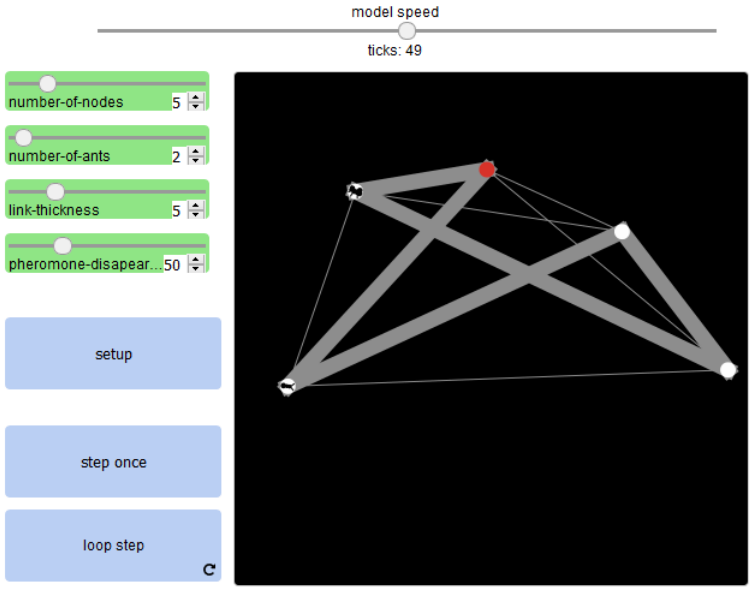



FIGURE 2.4 – Convergence erronée après 49 itérations pour 5 noeuds et 2 fourmis

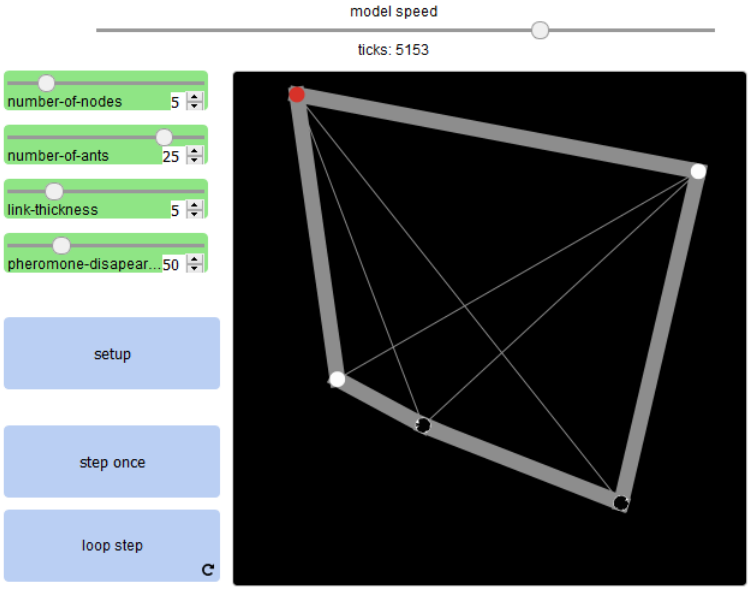



FIGURE 2.5 – Exemple après 5153 itérations 5 noeuds et 25 fourmis

Vous pouvez voir plus d'exemples en annexe B.

Nous avons également tracé le graphique d'influence du nombre de fourmis sur le nombre d'itérations (2.6). Ce graphe montre bien que le nombre d'itération croit avec le nombre de fourmis.

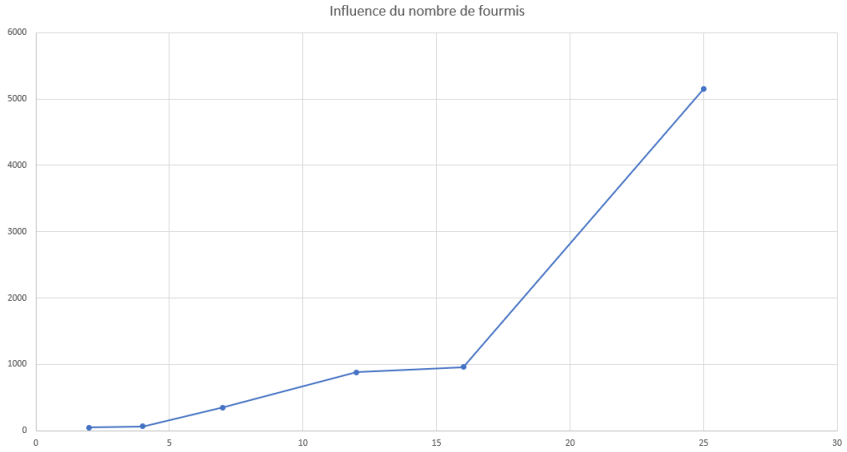



FIGURE 2.6 – Influence du nombre de fourmis sur le nombre d'itérations

Cependant, dans le cas d'un nombre de fourmis trop petit, l'algorithme semble converger vers une solution non optimale. Nous remarquons ça par exemple avec la figure 2.4, le chemin tracé en gris n'est sûrement pas le plus court possible. Le chemin choisit quand il y a une faible population de fourmis semble être le chemin pris par le plus de fourmis au premier tour. L'influence des phéromones est trop important dans ce cas sur le chemin choisit et les fourmis on trop tendance à prendre le même chemin par la suite.

# Conclusion

Ce projet nous a permis de modéliser le problème du voyageur de commerce avec un architecture multi-agents. Pour cela nous avons utilisés des *patches* pour les noeuds du graph et des *turtles* pour modéliser des fourmis se déplaçant entre les noeuds et émettant des phéromones.

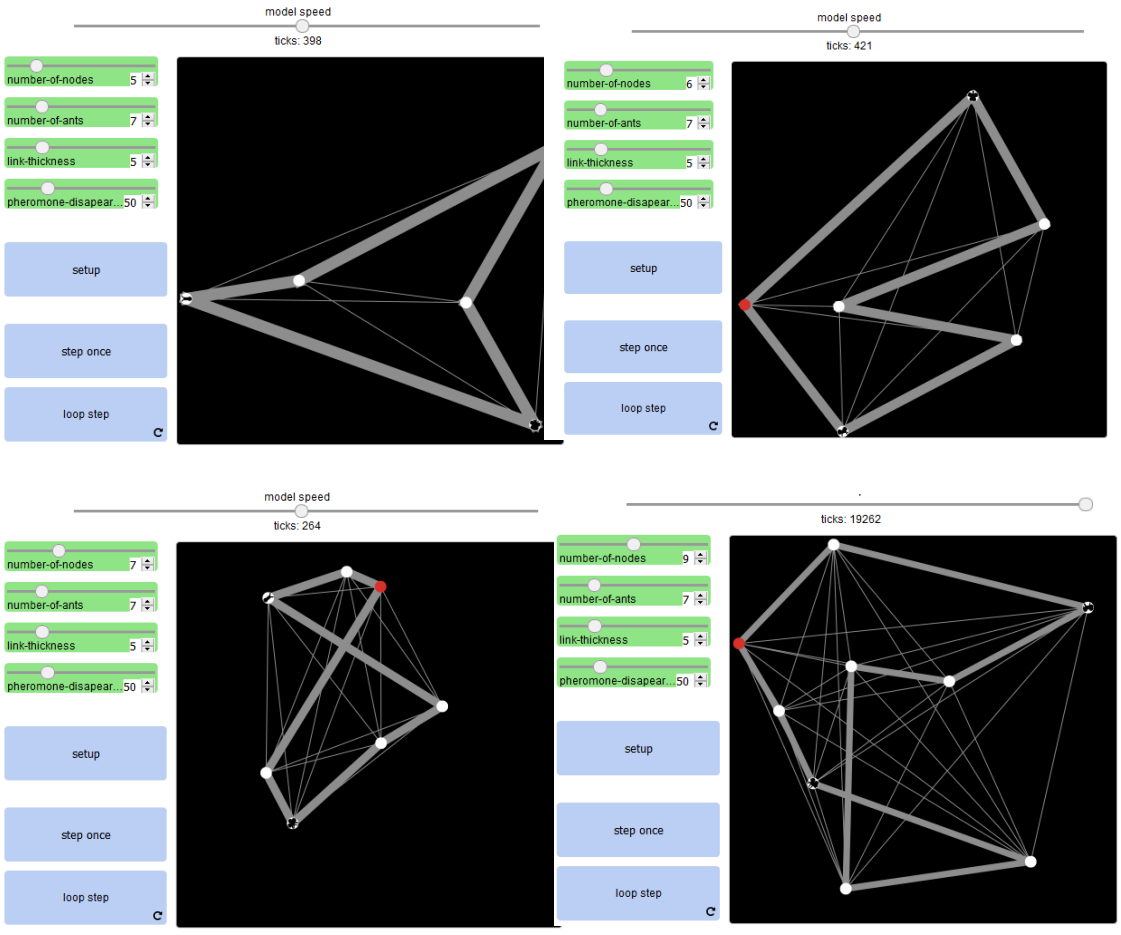
Nous avons analysé l'influence du nombre de noeuds du graph et du nombre de fourmis sur le temps de convergence du modèle. Plus le nombre de noeuds est important et plus il est difficile d'optenir convergence. Le temps avant d'optenir un chemin semble également augmenter avec le nombre de fourmis. Cependant nous avons vu qu'avec un nombre trop faible de fourmis, la solution obtenue n'est pas optimale.

La résolution du problème du voyageur de commerce avec un algorithme multi-agents ne semble pas très efficace. En effet, il converge plutôt lentement sauf configuration aléatoire particulière, il est difficile de définir le nombre de fourmis optimal pour obtenir une convergence rapide vers une solution optimale en fonction du nombre de noeuds du graph.

Il aurait été également bien de pouvoir fixer un graphe plutôt que de constamment en créer un aléatoirement et de pouvoir trouver le *tick* de convergence pour pouvoir réaliser des tests plus révélateur.

# Annexe A

# Exemples influences nombre de noeuds



# Annexe B

# Exemples influences nombre de fourmis