IA04 – TD7 : Simulation

L'objectif est prendre en main la plateforme agents MASON dédiée à la simulation. On s'inspirera du tutoriel vu en cours et de la documentation MASON. Télécharger l'archive MASON à partir du site Moodle.
L'espace est représenté par une grille 2D de taille fixée à l'initialisation. Des insectes placés aléatoirement peuvent se déplacer sur la grille et chercher de la nourriture afin de conserver leur énergie. Ils peuvent manger, charger de la nourriture et se déplacer avec. Sans énergie un insecte meurt. L'objectif est de garder le plus longtemps possible au moins un insecte vivant.

Etape 1 : Mise en place de l'environnement

Mettre en place le modèle de la simulation : champ, agents insectes, objets inertes. On choisira un champ de type `SparseGrid2D` qui autorise plusieurs éléments dans une même cellule. Toutes les caractéristiques du modèle devront être paramétrées dans une classe Constants. Placer sur le champ les insectes et la nourriture.
Mettre en place la visualisation : interface graphique, console. La visualisation aura une représentation (portrayal) de type `SparseGridPortrayal2D` afin de représenter le champ du modèle.

Caractéristiques statiques

Initialisées à la création de l'insecte, elles spécialisent son efficacité à réaliser certains comportements. Le nombre de points de capacité est un paramètre global et est à répartir entre les trois composantes suivantes :

- DISTANCE_DEPLACEMENT : (1 + points attribués au déplacement) : l'insecte peut se déplacer de X cases à chaque tour.
- DISTANCE_PERCEPTION : (1 + points attribués à la perception) : l'agent peut percevoir le contenu des n cases adjacentes (voir matrice d'adjacence).
- CHARGE_MAX : (1 + points attribués à la charge) : l'agent peut porter n quantités de nourriture entre 0 et CHARGE_MAX.

Des stratégies différentes pourront être basées sur des répartitions différentes de ce nombre de points.

Caractéristiques dynamiques

- Energie : [0; ENERGIE_MAX]. Un insecte perd des points d'énergie lorsqu'il se déplace. Il en gagne lorsqu'il mange. Les autres actions ne coûtent pas de points d'énergie. Lorsqu'il tombe à 0 point d'énergie un insecte meurt.
- Charge portée : [0 ; CHARGE_MAX] nombre d'unités de nourriture transportées.

Nourriture

La nourriture est une ressource disponible dans l'environnement. A chaque fois qu'un insecte récolte de la nourriture ou s'y nourrit, le nombre d'unités disponibles à cette ressource diminue d'autant. Lorsqu'une source de nourriture s'épuise, une nouvelle apparait à un endroit de l'espace choisi aléatoirement. A l'initialisation les endroits porteurs de nourriture sont aussi placés aléatoirement.

Comportements des insectes

a) vivre : si la jauge d'énergie d'un insecte passe à 0, il meurt et disparait de la simulation.
b) percevoir : à chaque itération, un insecte peut percevoir et mémoriser le contenu des cases adjacentes situées à une distance inférieure à sa distance de perception (DISTANCE_PERCEPTION). Percevoir ne coûte pas d'unité d'énergie et est immédiat (peut donc être réalisé avant **un** des comportements suivants).
c) manger : un insecte doit se nourrir pour rester vivant. Il doit être sur une case adjacente à une case où se trouve de la nourriture ou avoir une charge de nourriture non nulle (charge portée > 0) pour manger (une unité par tour, jusqu'à ce que la jauge d'énergie soit remplie ou que la ressource soit épuisée). Manger une unité de nourriture ajoute un certain nombre de points d'énergie défini par le paramètre FOOD_ENERGY.
d) se déplacer : un insecte peut se déplacer de X cases (entre 1 et sa DISTANCE_DEPLACEMENT). Chaque déplacement, quelle que soit sa distance, coûte une unité d'énergie et sa durée ne dure qu'un pas de temps.
e) charger : un insecte doit être sur une case où se trouve de la nourriture pour charger la nourriture (une unité par tour, action possible jusqu'à ce que sa capacité de stockage soit atteinte ou que le stock de nourriture de la case soit épuisé).

Etape 2

Mettre en place la stratégie des agents insectes (tous les agents peuvent ne pas avoir la même).

Démonstration de la simulation

On donnera à l'aide d'une courbe le résultat (nombres d'itérations où des insectes sont visibles) dans les conditions initiales suivantes :

GRID_SIZE = 30 : largeur de la grille
NUM_INSECT = 10 : nombre initial d'insectes
NUM_FOOD_CELL = 15 : nombre de cellules avec nourriture
MAX_ENERGY = 15 : maximum d'énergie d'un insecte
MAX_LOAD = 5 : charge maximale d'un insecte
CAPACITY = 10 : points de capacité à répartir chez l'insecte entre déplacement, perception et charge
MAX_FOOD = 5 : maximum de nourriture sur une cellule
FOOD_ENERGY = 3 : nombre de points d'énergie donnés par 1 point de nourriture

Matrice d'adjacence :

Les chiffres indiquent les cases situées à une distance de 1 pas et de 2 pas de la position x. L'espace est un tore et s'enroule sur lui-même horizontalement et verticalement.

2	2	2	2	2
2	1	1	1	2
2	1	x	1	2
2	1	1	1	2
2	2	2	2	2

Compléments

Suppression d'un insecte de la grille

Lors de la création d'un insecte, ajouter le `stoppable` retourné par la méthode `scheduleRepeating` :

```
Stoppable stoppable = schedule.scheduleRepeating(insect);
insect.stoppable = stoppable;
```

Pour supprimer un insecte, le faire disparaître et le retirer du scheduling :

```
model.yard.remove(this); //model est le SimState
stoppable.stop();
```

Inspector

Ajouter un inspector du modèle de la simulation dans la visualisation :

```
public Object getSimulationInspectedObject() { return state; }
public Inspector getInspector() {
  Inspector i = super.getInspector();
  i.setVolatile(true);
  return i;
}
```

Indiquer les champs du modèle à inspecter sous forme de Java Bean :

```
private int numInsects;
public int getNumInsects() {
 return numInsects;
 }
public void setNumInsects(int numInsects) {
 this.numInsects = numInsects;
}
```