Mini-projet

Simulation d'un écosystème

La simulation informatique est une technique utile pour comprendre les évolutions possibles et parfois difficiles à prévoir d'un système complexe : trafic routier, écosystème, mouvement de foule... Différentes observations peuvent être obtenues : imaginer s'il existe un moyen de mieux réguler des embouteillages, trouver une façon de combattre une espèce invasive etc. Dans le cas d'un écosystème, on s'intéresse ici à évaluer la robustesse (solidité, fragilité...) de l'équilibre qui compose la nature qui nous entoure.

Le but de ce projet en termes de programmation objet est de mettre en place un tel moteur de simulation en utilisant une modélisation objet des données et des algorithmes. Une des qualités qui sera évaluée sera de penser votre code pour qu'ils puissent facilement s'adapter à des ajouts futurs. Un autre des objectifs de ce projet est de mener à bien une campagne d'expérimentations menant à évaluer la qualité de la robustesse du système.

Déroulement du mini-projet :
Le mini-projet s'étale sur les deux mardis matin du 11 avril puis du 9 mai.
Le projet est à rendre pour le vendredi 19 mai minuit.
Le rendu est une archive contenant votre code et vos données (fichiers d'instances en txt), accompagnés d'un fichier pdf contenant les réponses principales aux questions et des captures d'écran si besoin.
Le projet est à rendre en binôme ou en monôme.

1 Prise en main de l'interface (partie non à rendre)

Cette première partie a simplement pour but de prendre en main l'interface et l'idée générale du projet. Le projet proprement dit commence à la section suivante : il n'est pas interdit de mener en parallèle (surtout en binome), cette partie avec la suite !

Téléchargez l'archive de l'exemple de gestion de GrilleNature qui se compose en 4 classes :

- Disque : (qui permet de stocker les données d'un disque)
- CaseGrille : une case d'une grille possède une couleur de fond et une liste de Disque
- GrilleNature : il s'agit à la fois : d'une classe permettant de stocker une matrice de CaseGrille et également un objet héritant de Jpanel. Un tel objet permet de faire des affiches graphiques dans une fenêtre graphique qui sera ouverte par votre programme.
- et un exemple basique de main manipulant la classe GrilleNature

L'exemple donné par ce Main crée une grille en mettant des cases de couleurs différentes, puis met au hasard un Disque dans certaines cases. L'affichage graphique produit par l'apelle à redessine (qui contient la méthode de Jpanel repaint) permet de donner une représentation graphique de ces données. **Important** l'affichage graphique n'est qu'une représentation des données, vous pouvez lire les commandes qui dessinent les cases et les disques en voyant qu'elles sont indépendantes des données, au sens que la correspondance entre les deux est faite par les lignes de code.

Pour prendre en main ce code, on se propose de faire jouer **l'écosystème proie/prédateur**, par exemple Lapin/Aigle, dont le principe est de visualiser ce qui se passe si le système a trop peu de proies mangées par des prédateurs pour les prédateurs survivent, ou si l'absence de prédateurs, ne fait pas exploser le nombre de proies...

Q 0.1 A partir du code exemple, on veut le comprendre en le modifiant dans le but

- d'adapter le code afin que l'on puisse afficher en ligne deux disques par case : l'un pour les proies, l'autres pour les prédateurs
- d'attribuer une couleur de disques aux proies et une couleur aux prédateurs. Le rayon d'un disque étant proportionnel au nombre de proies ou de prédateur dans une case de la grille.
- de faire un tirage au sort qui attribue aléatoirement des proies et des prédateurs en une quantité faible dans les cases de la grille. On fixe p_1 (respectivement p_2) le pourcentage de chance qu'un lapin (respo. aigle) soit placé dans une case.

Q 0.2 Afin de créer une simulation d'une évolution, on place dans le main une boucle dont les itérations représentent des pas de temps (une heure/journée/semaine/... A chaque itération, on applique l'une après l'autre les "règles" suivantes (que l'on appellent abusivement "loi de la nature")

- Reproduction des proies : en absence de prédateur, s'il y a au moins 2 lapins dans la même case, un 3ème lapin apparaît après p_3 itérations dans la case (quelque soit le devenir de ses parents pendnant les itérations précédéntes)
- Reproduction des prédateurs : en présence de proies dans une case, s'il y a au moins 2 aigles dans la case, un 3ème aigle apparaît après p_4 itérations dans la case.
- Prédation : un prédateur dans une case a p_5% de chance de manger un lapin de la case par itération
- Déplacement : une proie (respectivement un prédateur) a p_6% (resp. p_7%) chance de se déplacer dans une case adjacence : un tirage au sort équiprobable détermine la case. Pour simplifier, on

considère la grille torique.

Q 0.3 Avec un affichage, représenter cette évolution proie/prédateur. Tester-le afin de voir s'il est possible d'arriver à des cas extrêmes d'extinction des deux espèces ou au contraire d'équilibre : pour cela vous utiliser différents jeux d'essai représentés par les 7 paramètres $p_i, i = 1, \ldots, 7$.

2 Enoncé et modélisation

Dans le but de rendre plus réaliste la simulation d'un écosystème, on veut modéliser plusieurs phénomènes physiques et biologiques. Voici un texte décrivant l'écosystème à modéliser.

On considère un écosystème représenté par des zones (cases) d'un terrain. Chaque zone peut être de trois types de terrains différents : forêt, plaine ou désert, qui correspondent à des fourchettes d'eau et de température. Au départ, la grille est coupée en deux parties égales plaine et forêt. Chaque zone contient des animaux, des végétaux et les ressouces eau et température. Une zone bascule d'un type à l'autre si elle perd ses caractèristiques en eau et température : attention une bascuel vers désert ne permet pas le retour vers l'un des deux autres types. Une forêt devient plaine et vice-versa s'il y a beaucoup d'arbres. L'idée général est qu'une espèce a beaucoup plus de chance de pousser/se reproduire et moins de chance de mourir dans son espace naturel. Les animaux sont des insectes, des mamifères et des oiseaux. Parmi les mamifères, il y a les carnivores et les herbivores. Les végétaux sont des arbres ou des vivaces. Dans chaque catégorie, créer au moins deux sous-catégories. Chaque végétal a une température et un niveau d'eau critique où il peut mourir. Un herbivore mange des vivaces : s'il n'y en a pas assez, les herbivores meurent. Il y a au moins deux paires proie/prédateur à considérer. Pour simplier l'énoncé, la reproduction entre animaux est non genrée et la grille est torique pour les déplacements. Les animaux peuvent se déplacer. Si une zone contient beaucoup d'arbre, cela augmente le nombre d'arbres des zones contiguës. Les végétaux et les animaux ont besoin d'eau.

Exercice 1 – modélisation

Q 1.1 Donner une hiérarchie objets pour cet énoncé en pensant à

- l'héritage entre les classes
- des interfaces pour permettre la manipulation d'objets ayant des actions identiques sur la grille : se déplacer, voler, consommer de l'eau...
- les règles/lois sont les applications des méthodes d'interface. Noter qu'une règle peut-être un objet dérivant d'une classe mère règle.
- il est important de penser à maximimer les chances que l'ajout d'un nouvel arrivant puisse être pris en code dans votre code (par exemple, invasion d'une guèpe asiatique etc).

Exercice 2 – Simulation

Q 2.1 Créer la boucle de simulation de manière la plus courte possible : il doit s'agir de lancer les grandes règles qui seront ensuite lancées pour chaque instance d'une classe/interface correspondante.

Exercice 3 – Tests expérimentaux

Q 3.1 Une partie importante du projet est de produire des instances de votre hiérarchie où vous atteignez un équilibre entre les espèces présentes. Puis vous modéliserez une montée en température et une baisse en ressource d'eau afin de voire l'impact de cette évolution sur vos écosystème.