Projet n°10 - Zombie Attack !!

Simulation d'une propagation de zombie.

Contact: Thibault Tubiana, tubiana.thibault@gmail.com

Résumé

Le Virus T s'est échappé du laboratoire secret français D'*Umbrella Corporation*. Un scientifique a confondu son café et la fiole du virus… L'armée a besoin de vos compétences en Python afin de simuler la propagation de cette épidémie!

Il vous faudra donc modéliser la propagation du virus (et de la transformation des personnes) dans l'espace. Nous prenons comme exemple ici la France. La propagation se fait par la contamination d'individus au fur et à mesure du temps et suit un certain nombre de règles biologiques présentées plus loin. Vous pouvez également rajouter vos propres règles, à condition qu'ils ne simplifient pas la difficulté du projet.

Le projet étant à réaliser en Binôme, il est essentiel de bien subdiviser les tâches à réaliser individuellement. Il est également très important de réfléchir **avant** l'implémentation à la manière dont les différentes parties doivent s'imbriquer ensemble.

La partie Projet +++ n'est pas facultative. En fonction du travail rendu, nous estimerons s'il était possible pour vous d'aborder cette partie. Si tel est le cas, vous devrez réaliser au minimum une des tâches qui vous sont proposées. Laquelle de ces tâches vous déciderez d'implémenter n'a pas d'importance.

Description

1 - Gestion de l'espace

1.1 - Fichier d'entrée

Vous modéliserez la propagation de la pandémie en France métropolitaine. Vous avez donc besoin des bordures du territoire. Le fichier d'entrée, **France.svg**, est un fichier au format Scalable Vector Graphics (SVG) qui est un format de dessin vectoriel. Il s'agit d'un langage de balisage similaire à HTML ou XML.

Il convient d'utiliser les modules Python adéquates afin d'extraire les informations de ce fichier. Nous vous conseillons d'utiliser le module **xml.etree.ElementTree** . La balise qui vous intéresse ensuite est la balise **<path>**. L'attribut ***d*** contient entre autre les coordonnées des points qui constituent le polygone représentant la France. Attention: étudiez bien la structure de cette chaîne de caractères qui contient également d'autres informations que vous devrez ignorer. Lire également la partie 3.1 qui vous explique que vous devrez faire une conversion de ces coordonnées pour les utiliser dans votre programme.

1.2 – Délimitation des zones contaminables.

La pandémie ne peut pas se propager sur tout l'espace de la carte. Typiquement, les zones d'eau ne peuvent pas être infectées puisqu'elles sont inhabitées. Pour simplifier le problème, vous ferez l'hypothèse que toute zone extérieur e à la France métropolitaine ne peut pas être infectée (Les zombies n'ont pas beaucoup de mémoire et oublient alors leur passeport…).

Vous devez donc implémenter une fonction permettant de savoir si un point se situe à l'intérieur ou à l'extérieur d'un polygone afin de savoir si la pandémie peut continuer dans une direction donnée ou non (voir Algorithme 1).

```
fonction is_land(point,carte) :
    p = DERNIER_ELEMENT(carte)
    p0    = p.x
    p1    = p.y
    pt0   = point.x
    pt1   = point.y
    wind = 0
    pour chaque point_courant de carte :
        d0 = point_courant.x
        d1 = point_courant.y
        si p1 <= pt1
        alors
            si d1 > pt1 et ((p0-pt0) * (d1-pt1) – (p1-pt1) * (d0-pt0) > 0)
            alors wind = wind + 1
        sinon
            si d1 <= pt1 et ((p0-pt0) * (d1-pt1) – (p1-pt1) * (d0-pt0) < 0)
            alors wind = wind – 1
        p0 = d0
        p1 = d1
    renvoyer wind > 0
```

Algorithme 1 - Un point est-il à l'intérieur d'un polygone ?

2 – Modélisation de l'épidémie.

Vous allez implémentez un modèle probabiliste de propagation de l'infection. Pour cela vous allez suivre quelques règles :

- La pandémie part d'un point unique de la carte (un seul chercheur a été contaminé par le virus)
- A chaque pas de la simulation, chaque point de la carte infecté a une probabilité non nulle d'infecter une zone voisine
- Cette probabilité augmente avec le temps (ce phénomène modélise le temps d'incubation de l'agent infectieux).
- La pandémie ne peut pas s'étendre en dehors de la France.
- 3 statuts infectieux existent :
    - Non contaminé : ces personnes peuvent donc se faire contaminer.
    - En incubation (contaminé) : la personne est contaminé mais n'est pas encore morte, et ne peut donc pas contaminer quelqu'un d'autre. Attention, cette période est relativement courte, de par la toxicité élevée du virus, et de la menace des autres survivants essayant d'éliminer les personnes pouvant se transformer. Dans ce cas, la personne est morte et ne peut pas se transformer (vous pouvez soit supprimer le point, soit lui mettre une couleur de votre choix, noir par exemple).
    - Transformé : ces personnes sont transformé en zombie, et peuvent contaminer d'autres personnes non contaminées.

2.1 – Connectivité

Nous vous proposons deux manière pour représenter les infectés (si vous trouver une autre méthode plus adapté, vous êtes libre de l'utilisé, à condition qu'elle ne simplifie pas la difficulté du projet).

- Sous forme de points généré aléatoirement à un rayon de *n* km autour du point de départ au temps **t=1**, puis autour des points existant au temps $\mathbf{t_n}=\boldsymbol{t_{n+1}}$. Cependant, ceci est la méthode facile, et la note tiendra compte de cela.
- Sous forme d'un graphe (d'après la théorie des graphes). Il existe *n* points au temps t=0. Les points sont connecté entre eux par un facteur de distance (libre à vous de rajouter des facteurs). Au temps t=0, un point A devient contaminé, puis va contaminer tous les points qui sont connecté à lui. Ici les points peuvent être des villes, ou des personnes (attention de ne pas mettre 60 millions de points sinon la simulation risque d'être très longue à l'exécution).

3 – Visualisation de l'épidémie.

Les militaires ne comprennent pas les données brutes… Il faut donc leur faire une visualisation graphique. De plus, elle vous permettra de détecter très rapidement d'éventuels bugs dans vos algorithmes (si l'algorithme 1 est mal implémenté, si les données du SVG ne se chargent pas…).

Vous avez le choix de l'outil de visualisation. Cependant, nous vous conseillons vivement d'utiliser la bibliothèque python **matplotlib** pour afficher la carte au cours de la simulation. Pour l'installation sur un système Linux :

$ sudo apt-get install python-matplotlib

Si cette bibliothèque n'est pas installée sur les machines des salles informatiques, veuillez vous adresser à Cédric Benzino.

3.1 – Affichage de la carte.

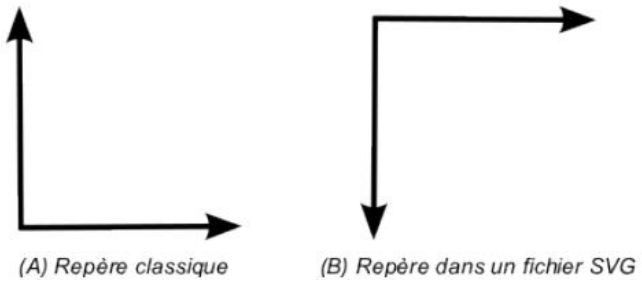

Figure 1 - Les repères utilisés dans ce projet.

Pour afficher la carte, vous devez faire attention au repère du format SVG : celui-ci est inversé par rapport à un graphique classique. Vous devez donc convertir les coordonnées que vous lisez dans le fichier pour qu'elles soient utilisables dans votre programme.

3.2 – Affichage de la progression de l'infection.

Vous choisirez le moyen le plus adapté selon vous pour visualiser l'évolution de la pandémie. Un moyen interactif serait néanmoins idéal. Ci-dessous un segment de code permettant d'afficher un graphique de manière interactive avec le module Python matplotlib.

```python
#!/usr/bin/env python
# -*- coding: utf-8 -*-


import matplotlib.pyplot as pyplot
import matplotlib.animation as mplanim

def run(data):
    """Cette fonction ajout sur le graphique un point de coordonnées
    aléatoires. matplotlib appelle cette fonction tout seul et lui
    envoie les données qui sont sont actuellement affichées. Cette
    fonction doit donc obligatoirement prendre (au moins) ces donnees
    en argument bien qu'on ne s'en servent pas dans ce cas précis."""
    point = random.uniform(-100, 600), random.uniform(-100, 600)
    # pyplot.scatter prend en argument la liste des x et des y
    pyplot.scatter([x], [y])


# initialisation de la figure : nous précison la taille
# de la figure car elle doit être carrée pour que la carte ne soit
# pas déformée
fig = pyplot.figure(figsize=(10, 10))

# dessiner la carte ici...


# initialisation de l'animation : on passe en argument la figure
# ainsi que la fonction qui doit être appelée en boucle avec un
# intervalle interval (en ms).
anim = mplanim.FuncAnimation(fig, run, interval=10)


pyplot.show()
```

Figure 2 - Création d'un graphique interactif avec matplotlib

Deux méthodes peuvent être employées pour la visualisation de la progression de l'épidémie

- Visualisation par des points de couleurs apparaissant sur la carte.
    - Si vous avez utilisé la méthode de création des points par un graphe, vous pouvez représenter les points non contaminés, ce qui permettra de suivre la progression de la maladie par les changements de couleur.

- Sous forme d'*heatmap (cf*. figure 3).

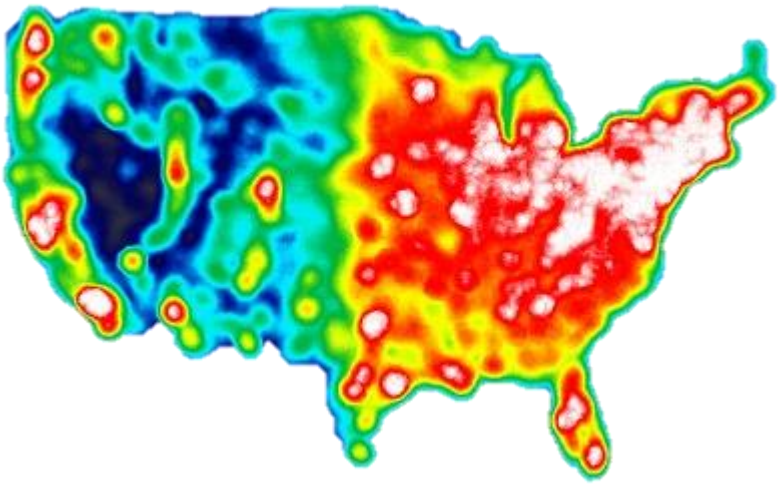

Figure 3 - Exemple d'une heatmap sur une carte des États-Unis.

4 - Projet +++

4.1 – La pandémie voyage.

Jusqu'à présent, la Corse ne pouvait pas être touchée, car c'est une île et que la pandémie ne pouvait pas traverser la mer.

Ajouter dans votre modèle la possibilité pour l'infection de traverser la mer et d'infecter la Corse : si l'infection est proche de la Corse, alors elle a une certaine probabilité de traverser la mer et donc d'infecter la Corse. Vous pourrez si vous le souhaitez faire ne sorte que l'infection de la Corse début sur la côte (car c'est là que les bateaux débarquent les voyageurs).

4.2 – Vitesse de propagation améliorée.

De la même manière que l'infection ne peut pas se répandre dans les zones désertiques, il est logique qu'elle se développe plus vite dans les zones de forte densité de population. Faites en sorte que votre modèle tienne compte de la densité de population soit en vous basant sur des données réelles, soit en définissant vous-même des zones de plus faire densité de population.

4.3 – Plusieurs méthodes de visualisation.

Vous pouvez également représenter la progression de la maladie par les deux méthodes présentées dans cet énoncé, ou bien en rajouter une de votre goût.

4.4 – Les zombies peuvent mourir aussi !

Si l'armée voit votre simulation, elle prendra peur ! Il y a donc de forts risques qu'elle essaye de faire reculer l'épidémie en tentant de tuer les zombies. Tenez-en compte dans votre simulation. L'armée pourra également faire usage d'armes atomiques sur de vastes zones où il ne reste que des zombies.

4.5 – Laissez aller votre imagination !

N'hésitez pas à laisser aller votre imagination et améliorer votre modèle ou votre programme en y ajoutant ce que vous souhaitez : les possibilités sont extrêmement nombreuses.

Idée originale : Benoist Laurent.