GIANG
Elodie
3678177

Rapport Mini-Projet 3

Le but du mini-projet 3 est de réaliser de multiples simulations avec de nombreux paramètres différents afin de collecter des données et ainsi les manipuler afin de construire des graphes pertinents pour évaluer l'impact des variations de différents paramètres sur des indicateurs sélectionnés.

Exercice 1. Collecter des données expérimentales

La première partie de l'exercice 1 consistait donc à réaliser un code python qui puisse nous permettre d'exécuter le simulateur avec les mêmes paramètres de simulation. En effet, l'exécution multiple d'une même simulation permet d'obtenir une moyenne significative du résultat attendu lors du lancement de la simulation avec cet ensemble de paramètres, moyenne qui nous sera utile notamment durant la phase de visualisation des données qui nous sera demandée dans l'exercice 3 du mini-projet. Notre code pour cet exercice se trouve dans le fichier « exercice1.py ». Il nous permet de créer un fichier (dont le nom est défini par l'exécuteur du programme) qui contiendra les sorties d'un nombre déterminé (ici 1000) d'exécution d'une même simulation. Nous avons utilisé la bibliothèque subprocess afin d'exécuter la simulation de l'archive « Virus_old.jar » avec les paramètres : '-stop_all_sane=1' (1), '-printout=2' (2), '-gui=0', '-nb_snapshots=5000'(3). Le paramètre (1) permet d'arrêter la simulation lorsque tous les individus sont guéris car la simulation ne s'arrêtait pas, par défaut. (2) nous permet d'afficher la sortie de la simulation sous forme de chiffres tels que : une ligne correspond à un snapshot, chaque chiffre représente un individu, un individu à 0 n'est pas contaminé, à 1 est contaminé, à 2 est immunisé. (3) permet d'éviter les simulations infinies en nous limitant à 5000 snapshots. Ce nombre de snapshots a été décidé car il correspondait à une durée d'environ 10 min, ce qui nous permettait d'avoir une idée du caractère « infini » de la simulation.
Un petit problème qui m'a été posée lors de la question 1 est l'utilisation du paramètre capture_output=True de la fonction run de subprocess. En effet, dans ma version de python, ce paramètre n'existe pas. Il a fallu alors importer PIPE de la bibliothèque subprocess afin de pouvoir passer en argument du run : stdout=PIPE, stderr=PIPE.
La deuxième partie de l'exercice 1 consiste en l'implémentation d'un code permettant de réaliser plusieurs séries de simulations en faisant varier 1 paramètre. Pour chaque variation, nous nous servirons du code implémenté dans la première partie afin d'obtenir plusieurs sorties de simulation pour une variation donnée. Par la fonction input, je demande à l'utilisateur le nom du fichier dans lequel il souhaite écrire les sorties de chaque variation (chaque variation voit ses sorties de simulations écrites dans un fichier txt), le paramètre qu'il souhaite faire varier (il peut s'agir de : -contagion_period, -immune_period, -infection_period, -nb_infected, -nb_nodes, -travel_distance), la valeur initiale de ses variations (que je stocke dans la variable mini) et le « pas » des variations. Ainsi, il s'agira de faire 50 simulations pour chaque variation. Il y aura un total de 30 variations et chacun sera espacé avec un pas de « pas » (ex : si on souhaite faire varier le paramètre -contagion_period avec mini=3, pas=4, alors -contagion_period prendra les valeurs 3, 7, 11, etc..). Le nom du fichier pour chaque variation sera par exemple de la forme : « contagion_period3.txt » pour un ensemble de simulation dont la valeur de la

variation est 3 et le paramètre à faire varier étant « -contagion_period ». Le code python de cette partie a été implémenté dans le fichier « ex1p2.py ».

Exercice 2. Manipuler des données expérimentales

L'exercice 2 consiste à exécuter le programme écrit à la suite de la deuxième partie de l'exercice 1 sur des paramètres donnés et d'évaluer l'impact des variations de ces paramètres sur : la durée de l'épidémie, la proportion maximale de la population qui est infectée et la distribution des multi-infections.
Voici les paramètres choisis pour chacun des paramètres variables choisis :

Nom de fichier : nb_infected , mini=1, pas =3
Nom de fichier : travel_distance : mini=60, pas = 6
Nom de fichier : infection_period : mini=3, pas =4
Nom de fichier : contagion_period : mini=60, pas = 6
Nom de fichier : immune_period : mini=300, pas =3
Nom de fichier : nb_nodes : mini=5, pas=4

Un problème qui s'est posé directement a été la caractérisation de chaque indicateur donné : comment déterminer la durée d'une épidémie ? Comment calculer la proportion maximale d'individus infectés durant une simulation ? Qu'est ce que la distribution des multi-infections ? La mise en forme de la sortie d'une simulation (en plaçant -printout à 2) nous a énormément facilité la tâche dans cette caractérisation. J'ai décidé de déterminer la durée d'une épidémie en comptant le nombre de snapshots produits lors d'une simulation. Pour cela, il suffisait de compter le nombre lignes générés à la sortie d'une simulation. La proportion maximale d'individus infectés durant une simulation est déterminée par le maximal des proportions d'infectés par ligne (désignés par le chiffre 1). Enfin la distribution des multi-infections consiste à déterminer combien de nœuds ont été infectés une fois, deux fois, trois fois, etc. Puisque nous obtenons plusieurs résultats pour une seule variation (plusieurs exécutions sont effectuées pour chaque variation), il va falloir calculer une moyenne des résultats obtenus pour chaque variation. Pour la durée de l'épidémie ainsi que la proportion maximale d'individus infectés durant une simulation, l'opération est triviale (moyenne des résultats obtenus par la somme des résultats obtenus divisé par le nombre de simulation par variation). Le cas de la distribution des multi-infections m'a été plus complexe. En effet, le résultat de chaque simulation est donné sous la forme d'un dictionnaire dans lequel les clés correspondent au nombre d'infections et les valeurs correspondent au nombre de nœuds ayant eu ce nombre d'infection. Il s'agit donc, pour chaque variation, de calculer la moyenne des distributions obtenues.

Chaque calcul des indicateurs donnés a été séparé dans des fichier différents. Pour le calcul de la durée de l'épidémie pour un paramètre variable donné, nous avons implémenté notre code python dans le fichier « ex2_duree_epi.py ». Celui qui détermine la proportion maximale d'infectés se trouve dans le fichier « ex2_prop_max_infected.py ». Enfin, celui qui permet de déterminer la distribution des multi-infections se trouve dans le fichier « ex2_distri_multi_infected.py ».
Le résultat des fonctions est écrit respectivement dans les fichiers :
ex2_res_duree_epi_nom_parametre_variable .txt,
ex2_res_prop_max_infected_nom_parametre_variable.txt et
ex2_res_distri_multi_infected_nom_parametre_variable.txt avec nom_parametre_variable le nom des paramètres suivis.

Exercice 3. Tracer des données expérimentales

L’exercice 3 permet d’exploiter les données collectées dans l’exercice 2. A l’aide de json, nous pouvons lire les données et les modéliser avec des graphes. Un problème qui m’a été posé a été la modélisation de la distribution des multi-infections. En effet, tandis que pour les 2 autres indicateurs, il a été trivial de les modéliser sous forme de courbe, il a fallu pour celui-ci le modéliser sous forme d’histogramme. Cependant, après de multiples essais, je n’ai pas été en mesure de le modéliser. Le code python pour les deux autres indicateurs se trouvent dans les fichiers ex3_duree_epi_graph.py et ex3_prop_max_infected_graph.py. Le nom de fichier a rentrer pour ces codes sont les noms des fichiers. Il suffit pour cela de taper par exemple : contagion_period. J’ai expliqué les graphiques dans ma vidéo de présentation du projet, mais je vais mettre des exemples à l’écrit ci-dessous.

Evolution de la durée de l’épidémie en fonction de la période de contagion : Lé période de contagion désigne la période durant laquelle un nœud peut contaminer un autre nœud. Dans la logique plus un nœud peut contaminer un autre nœud, plus longue sera l’épidémie. Ici, on peut le voir grâce au graphique qui montre une augmentation exponentielle de la durée de l’épidémie suivant l’augmentation de la période de contagiosité.

Evolution de la durée de l’épidémie en fonction de la période d’immunité : La période d’immunité désigne la période durant laquelle un nœud reste immunisé. Logiquement plus un nœud est immunisé longtemps, et moins l’épidémie durera longtemps, et c’est ce que l’on peut constater avec le graphique.

Evolution de la durée de l’épidémie en fonction de la période d’infection : La période d’infection désigne la durée durant laquelle un nœud reste infecté. Logiquement on devrait avoir une augmentation de la durée de l’épidémie cependant, on regarde une parabole qui montre une augmentation suivie d’une diminution de l’évolution de la durée de l’épidémie au bout d’un certain temps. C’est un comportement imprévisible.

Evolution de la durée de l’épidémie en fonction du nombre d’infectés initial : Logiquement on devrait constater une diminution de la durée de l’épidémie puisque plus il y a d’infectés initiaux et moins il y aura de nœuds à infecter, ce qu’on constate effectivement sur le graphique.