Université de Paris 2020/2021

« Corona Bounce »

Projet de

Aliche Zahra, Bragina natalia, Golikova Anna, SaHI KENZA, RAVENEAU EMILIEN

Sommaire:

[Cahier des charges 2](#_Toc69656694)

[Les objets principaux 2](#_Toc69656695)

[Réglages possibles 2](#_Toc69656696)

[Coté GUI 3](#_Toc69656697)

[Schéma des classes 3](#_Toc69656698)

[Parties traitées 3](#_Toc69656699)

[Chapitre 1 3](#_Toc69656700)

[Chapitre 2 3](#_Toc69656701)

[Gestion des threads 3](#_Toc69656702)

[Difficultés rencontrées 3](#_Toc69656703)

[Problèmes connus 3](#_Toc69656704)

[Pistes d’extensions 4](#_Toc69656705)

Introduction

Notre projet est une simulation d’épidémie . On peut observer sa propagation dans deux populations virtuelles . Cela permettra en variant les paramètres et scénarios et en analysant les courbes de contamination et de guérison, de faire un point sur l’efficacité des stratégies adoptées et de déduire les facteurs inhibiteurs et stimulateurs de la diffusion de la maladie.

# Cahier des charges

## Les objets principaux

Les individus représentés par la classe Individual, peuvent prendre 4 états :

sain, malade, contagieux et guéri.

Les individus sont perçus comme des points de la Zone d’étude.

Ils ont une vitesse bornée par une valeur maximale et circulent aléatoirement (déplacements libres) si aucun scénario n’est appliqué, de manière restreinte dans le cas contraire.

Un objet « CoquilleBille » encapsule un individu ayant une vitesse 2-dimensionnelle et une position.

« ConfinedBille » est un type spécial de CoquilleBille encapsulant les individus d’une population soumise à un confinement (SoftLockdown)

La Zone représente l’environnement de vie d’une population . Elle est caractérisée par une hauteur et largeur.

Les objets « Wall » sont des frontières restreignant les déplacement des individus (déplacements régionaux uniquement). Chaque frontière a une position X et Y et une largeur . Dans l’implémentation, les frontières sont descendantes .

Les graphes affichent en temps réel le développement de l’épidémie en parallèle avec le nombre de personnes malades/contagieuses, saines et guéries .

En appuyant sur les boutons du menu principal on peut :

1. Lancer le mouvement (« Start »)
2. Arrêter le mouvement et le reprendre (« Pause/Resume »)
3. Changer les réglages (« Settings »)
4. Recharger les populations à nouveau tout en gardant les réglages ainsi que scenarios (« Reset »)
5. Montrer la légende pour chaque élément de la scène (« ? »).

## Réglages possibles

Notre programme permet à l’utilisateur d’effectuer les changements tant les paramètres de populations et de maladie, que de choisir les scenarios appliques par gouvernement.

Manettes de réglages à partir de « Settings » permettent varier les valeurs de paramètres suivantes :

1. Maladie

* Durée de la guérison
* Durée de l’immunité
* Rayon de contamination

1. Population

* Nombre de personnes

1. Espace

* Nombre de murs (qui restreignent le mouvement des points)

Outre cela nous proposons de choisir parmi les scenarios suivants :

* « No scénario » (déplacement aléatoire et non limité)
* « Lockdown » (déplacement s’effectue dans une zone de quelque rayon de position initial d’individu)
* « Bounds » (déplacement aléatoire mais limité par les frontières)
* « Movement limited » ().
* « Lockdown + bounds »

## Coté GUI

Fenêtre s’adapte au dimensions de l’écran.

# Schéma des classes

# Méthodologie et déroulement du projet

## Chapitre 1

Nous avons une population ayant un certain nombre d’individus, ils circulent dans un environnement « Zone ».

Il y a 4 classes représentant les 4 états de santé possible de chaque individu :

Sick , Healthy, Recovered et Incubating .

Au début de la simulation , la population est dispersée aléatoirement sur la surface de la zone .Par la suite , les points circulent avec une certaine vitesse aléatoire bornée par une valeur maximale (que l’on peut changer). La contamination se produit si un point sain est à un rayon\_de\_contamination près d’un point malade/contagieux . Il rentre dans une phase d’incubation(avant l’apparition des symptômes ) avant qu’il ne soit compté parmi les malades. Il devient guéri après une période de temps et acquiert une immunité qui le protège contre une nouvelle infection temporairement,

Différents scénarios :

SoftLockDown  : Dans ce cas de figure, un type spécifique de CoquilleBille(s)

est crée : ConfinedBille (s) caractérisé par une vitesse et un rayon de déplacement restreints .Les individus ne se déplacent que dans un voisinage de leur position initiale et avec une vitesse réduite.

RestictedMouvement (StrictLockdown): Une grande partie de la population est forcée de rester chez elle, aucun déplacement n’est toléré. Une minorité de gens peuvent circuler, cela concerne notamment ceux qui exercent des métiers indispensables au bon fonctionnement et à la sécurité de la population, policiers, docteurs, pompiers, livreurs ,...Ou aux urgences.

Boundaries : Des frontières divisent progressivement la zone en sous-zones pour limiter les déplacements (déplacements régionaux uniquement) .

Progressivement ? Les frontières ne sont pas placés instantanément mais petit à petit le long de la zone . Les points sont bornés par ces dernières et ne peuvent changer de région .

Boundaries + Softlockdown : Une combinaison entre frontières et confinement cités plus haut .

Les statistiques montrent bien que le taux de contamination diminue conséquemment.

## Chapitre 2

## Vue – Modèle - Controller

## Gestion des threads

Dans notre projet nous avons implémenté le multi-thread architecture.

Etant donné nous avons 5 threads : l’un qui est principale (celui de App.java) et deux pour chaque une des populations (un pour gérer le déplacement des points, l’autre pour gérer les cycles de maladie et sauvegarder l’historique).

A part de main thread, les threads se créent dans deux classes : classe Population et classe Zone.

Le thread de timer crée dans la Population partagent les classes suivantes :

* Classe Incubating – pour gérer la durée d’incubation de maladie, temps écoulé - Incubating devient Sick. Durée = durée de non contamination.
* Classe Sick - pour gérer la durée de maladie après le contact – dès que le temps s’écoule l’individu Sick change la classe pour Recovered. Durée = Durée de maladie.
* Classe Recovered – pour gérer la durée de non contamination, temps écoulé - recovered devient healthy. Durée = durée de non contamination.
* Classe Population – pour sauvegarder la statistique afin de la représenter en graph.

Le thread de timer crée dans la Zone partagent les classes suivantes :

* Classe Zone – pour effectuer le déplacement des individus.

Ou est Mur ?

L’animation dans GUI est attachée au thread principal

Reentrantlock

# Difficultés rencontrées

# Problèmes connus

# Pistes d’extensions

* Frontières fermée, ouverture s’effectue après.
* Réglage de la vitesse des points.