CCBDA Final Project First draft

Andreas Haukeland, Haftau Hailu, Jorrit Palfner, Mar Vidal Segura

In wake of the coronavirus pandemic many nations have imposed drastic isolation measures to combat the spread of the virus. Contact tracing apps are being discussed as a measure to keep the number of new infections low while reducing isolation. In this project, we therefore aim to create a simulator for infection spread and approximate the impact of widespread use of contact tracing apps.

Project

Functionality & Scope:

We propose to make a simulator of infection spread based on activity patterns of individuals from different categories. The activity patterns, density, and other parameters can be modified by the user to create different scenarios. For interaction with the simulator, we will create a web application which serves as a dashboard from which the simulator can be run. The output will be shown in the dashboard to be able to compare the different scenarios.

In particular, we want to compare how infections spread with and without the use of infection-tracking with the prototype proposed by Apple/Google to see it’s impact.

Simulator

Simulation Description:

The simulation sketches a simple scenario of agents commuting to work for daily eight hour shifts by different means of transport, public or individual, from one district to work for approximately 30 minutes each way. Agents practice careful social distancing therefore spend the rest of the day at home. At home and work the agents meet the same other agents every day. However, agents using public modes of transport interact with different agents selected at random every time. Agents using individual means of transport which could be transport by car, foot or bicycle do not have interaction during their way to work.

Architecture:

Web-service works as the controller for the simulator. When someone inputs data in the web-service, it will deploy an EC2 instance which initializes the simulator.

Regions (locations like work, homes, etc..) are modeled as processes in which interactions take place.
Agents are of different classes (Student, Workers, Children, Elderly) with unique identifiers and daily activity patterns. Agents are modeled as objects and can have two variable statuses location and health.
Database stores interactions of two people by identifiers plus date, time as well as their statuses. Interactions are modeled as functions.

Tracing App:

Stores the tuple (id1, id2, time) for each interaction between agents in the same region. This is done every X minutes.
At the end of every day, the app is notified of users who have gotten infected. Their ids are then used to filter the interactions and warn every agent who has been in contact with that user.

Initialization:

Agents are created with unique IDs and assigned to a mode of transport, office, mode of transport and home. They will also be assigned an activity pattern which determines where they will be at different times during the day. For the start an 8 hour workday is assumed for every agent.

Agent Classes:

Classes are composed of: ID, assigned_regions, region_present (in which region the agent resides at the moment), health_status (H= healthy, I= infected spreading the disease, IT= infected tested and isolated),

Interactions:

Infections are spread by interactions. Agents interact when they are in the same region.

- Model #1 for interactions:
  We assume that an agent interacts with all of the agents that are in the region at the same time. For each interaction between an infected and non-infected there is a probability given by the region_risk and infection_risk that an infection takes place.
  Parameters for infections:
  - Region_risk: risk of infection differs from region to region
    - Transport: infection risk medium, Home: infection risk very high, Office: infection risk medium
  - Infection_risk: the general infection risk given by user

Activity patterns:

Activity pattern is a distribution based on a user category of where they are during the day.

- Model #1:
  We have 1 user category. All the users in this category have the pattern HWM (home,work,home). The workplace and home they are connected to is determined at initialization of the simulation.

Fixed parameters of model:

Infection status: susceptible, infected(tested, not tested), infectious, treated and cured.
Characterization: students, workers, stay-at-home or elderly.
Regions: Offices(30 agents per room for 8h), Transport(Public(50 random agents 30 min), Individual(no risk)), Home

Input parameters:

Incubation Time
Symptomatic Time (Distribution)
Mortality rate
Infection rate (How many agents does an infected agent infect)
Number of citizens
Density

Output parameter:

# Cases per time (infected agent)
# Interactions per time
# Casualties per time and total
# Cured per time and
# Percentage of cured agent vs total of cured + dead
(Types of heatmaps would be really cool, but requires use of geo)

Web Application

The purpose is the web application is to be a user interface to run the simulator from with varying input parameters and display the simulation results.

Architecture:
Using elastic beanstalk specifics are subject of research and still to be determined.

Project Organization

Coding languages:

- Python3 + Django for the server

Resources and Services:

- EC2: to hold the virtual server.
- Elastic beanstalk: to deploy, manage and scale the virtual servers.
- DynamoDB: to store all simulator data.
- Elastic & Kibana: to design visualization.

Use of time:

Architectural design	3.75h	15h
Research and documentation on services and resources	2.5h	10h
Coding	25h	100h
Meetings • Project sprint meeting (30') • Group meeting before project sprint meeting (30')	5h	20h
Written documentation • Draft • Presentation • Final report	3.75h	15h
Total		160h

Global Milestones:

- Simulator with basic functionality (2 regions, 1 agent class, cmd IO)
- Set up Database
- Automated launch of ec2 instance and Database to run the simulator
- Add tracing app function to the simulator
- Visual representation of simulation results
- Connecting the web service to the simulator, and receiving input /output in web (development can start when output of simulator is defined)
- More advanced regions(urban vs rural) and user patterns (differ them in terms of classes) in simulator
- Implement auto-scaler for simulator (if performance is becoming a bottleneck)
- Looking at alternatives for using geographical data for

1st sprint Milestones:

- Simulator with basic functionality (2 regions, 1 agent class, cmd IO)
- Set up Database
- Automated launch of ec2 instance and Database to run the simulator
- Research web application architecture
- Refine project draft with professor feedback + research

Extra Numbers to research:

Average household distribution (number of agent living in the same household)
Modal Split (% of modes of transportation used by agent)
Population distribution
Number of interactions