Technische Universität Berlin **SoSe 2020**
Fakultät II, Institut für Mathematik
Sekretariat MA 6–2, Antje Schulz
Prof. Dr. Michael Joswig
Dr. Frank Lutz, Sarah Morell, Manuel Radons

Programmierprojekt Computerorientierte Mathematik II

Abgabe des ersten Teils: 26.06.2020 über den ComaJudge bis 18:00 Uhr

Thema

Im Rahmen des diesjährigen Programmierprojekts wird die Ausbreitung eines Virus in einer Population als Zufallsprozess auf einem Graphen simuliert. Das hier präsentierte Modell einer Virusausbreitung ist stark vereinfacht und dient der Einübung der in der CoMa unterrichteten Methoden. Insbesondere ist es für konkrete Aussagen über das aktuelle Geschehen im Zusammenhang mit SARS-CoV-2/COVID-19 nicht geeignet.

Ablauf

Das Projekt besteht aus *zwei Teilen*: der *Modellierung* und der *Visualisierung*. Die Modellierung wird, so wie auch die bisherigen Programmieraufgaben, über den Comajudge abgegeben. Die Abgabefrist ist der 26. Juni 2020, 18:00 Uhr. Die Visualisierung soll in selbstständiger Arbeit die Bibliothek `matplotlib` verwenden. Dieser Teil muss bis zum 10. Juli 2020 einer Tutorin oder einem Tutor vorgestellt werden.

1. Teil (Modellierung)

In unserem Modell werden Personen durch Knoten eines einfachen Graphen modelliert, die Beziehungen zwischen den Personen durch (ungerichtete, ungewichtete) Kanten. Zu diesem Zweck sind zwei Klassen zu schreiben.

Klasse Node

Schreiben Sie eine Klasse `Node` mit folgenden Attributen:

- `key` Schlüssel eines Graphen-Knotens als nicht-negative ganze Zahl.
- `coordinates` Gitterkoordinaten eines Graphen-Knotens als 2-Tupel nicht-negativer ganzer Zahlen.
- `infection_date` Zeitpunkt der Infektion als nicht-negative ganze Zahl. Wird auf -1 gesetzt, falls nicht infiziert.
- `neighbors` Liste von Schlüsseln von Nachbarknoten, in aufsteigender Reihenfolge sortiert.
- `active` Wahrheitswert. Wird nach durchlaufener Infektion auf `False` gesetzt. Dies entspricht der Immunisierung oder dem Versterben der simulierten Person.

Im Konstruktor sollen die ersten vier Attribute übergeben werden. Das Attribut `active` wird mit `True` initialisiert.

Klasse Virus

Schreiben Sie eine Klasse `Virus` mit folgenden Attributen:

- `time` Zeitpunkt als nicht-negative ganze Zahl.
- `incubation_period` Zeitspanne bis zum Ausbruch der Krankheit als nicht-negative ganze Zahl. Als vereinfachende Annahme gehen wir davon aus, dass ein Infizierter vom Ausbruch der Krankheit an (also nach Ablauf der Inkubationsperiode) und für die Dauer der Krankheit ansteckend ist.
- `contageous_period` Zeitspanne, während der die infizierte Person ansteckend ist, als nicht-negative ganze Zahl.
- `transmissibility` Ansteckungswahrscheinlichkeit als Float im Intervall $[0, 1]$.
- `graph` Liste von `Node`-Objekten, in aufsteigender Reihenfolge der Schlüssel sortiert.

Im Konstruktor sollen Initialisierungswerte für alle Attribute übergeben werden. Schreiben Sie des Weiteren zwei Methoden `time_step(self)` und `time_steps(self, n)`. Die Methode `time_steps` führt `n` mal `time_step` aus. Die Methode `time_step` führt die Simulation der Virusausbreitung wie folgt durch:

1. `self.time` wird um 1 inkrementiert.
2. Für jede `Node` `Nd` in `self.graph` wird, in der aufsteigenden Reihenfolge der Keys der Knoten, falls der Knoten aktiv ist und das Infektionsdatum *nicht* -1 ist, das Folgende durchgeführt:
   - Falls `self.time` *größer* ist als

     `Nd.infection_date` + `self.incubation_period` + `self.contageous_period`,

     setze `Nd.active` auf `False`.
   - Falls die vorige Bedingung nicht erfüllt ist, aber `self.time` *größer* ist als

     `Nd.infection_date` + `self.incubation_period` ,

     führen Sie für jeden Schlüssel `k` in `Nd.neighbors`, in aufsteigender Reihenfolge, für den Knoten `Nd_k` mit dem Schlüssel `k`, falls `Nd_k.infection_date` gleich -1 ist, das Folgende aus: Falls `random.random()` kleiner ist als `self.transmissibility`, dann setze `Nd_k.infection_date` auf `self.time`.
     Hinweis: Führen Sie die Zufallsabfrage für jeden Nachbarn einzeln durch, nicht einmal für alle. Importieren Sie hierzu `random` einmalig außerhalb Ihrer Klassen. Nutzen sie dazu nur den Befehl `import random`. Führen Sie keine Kürzel ein, wie zum Beispiel mittels `import random as rd`.

Abgabe

Es gibt auch bei dem Modellierungsteil des Projekts die zwei bisherigen Abgabemöglichkeiten Jupyter-Hub und Abgabe über das Terminal. Im Jupyter-Hub wird für Sie automatisch das Abgabeverzeichnis `co2_Projekt` angelegt. Bei der Abgabe per Skript ist der Name der Aufgabe ebenfalls `co2_Projekt`.

2. Teil (Visualisierung)

Schreiben Sie eine Klasse `VisualizeVirus`. Der Konstruktor dieser Klasse soll als Parameter ein Objekt vom Typ `Virus` entgegennehmen. Stellen Sie Methoden zur Visualisierung der Simulation der Virus-Ausbreitung bereit. Eine zentrale Anforderung ist, dass Sie die Methoden der `Virus`-Klasse zur Durchführung der Simulation nutzen, während die Methoden von `VisualizeVirus`

lediglich der visuellen Aufbereitung der so gewonnenen Daten dienen. Benutzen Sie zur visuellen Aufbereitung die Bibliothek `matplotlib` in geeigneter Weise.
Sie sind bei der Bearbeitung dieser Teilaufgabe relativ frei. Das Kriterium für eine erfolgreiche Abnahme ist, dass die Ausbreitung des Virus anhand der Visualisierung nachvollziehbar sein soll. Die Ergebnisse sind bis Freitag, den 10. Juli, bei Ihrer Tutorin oder Ihrem Tutor vorzustellen.
Hinweis: Eine andere Bibliothek als `matplotlib` darf *nur in Absprache mit und Genehmigung durch Ihre Tutorin oder Ihren Tutor verwendet werden*, da dies unter Umständen einen erheblichen Zusatzaufwand bei der Betreuung verursachen kann.