Python Aufgabe Pilzsammler:innen (Spieltheorie)

Torben Friedrich Görner

Dezember 2022

1 Das Szenario

Täglich gehen Plizsammler:innen als Gruppe in den Wald, um Pilze zu sammeln. Am Abend wollen alle eine Pilzpfanne kochen. Wenn die Pilzsammler:innen am Abend ihre gesammelten Sorten teilen, muss jeder nur noch 1€ für zusätzliche Pilze ausgeben. Wenn beide nicht teilen, müssen beide 3€ für weitere Pilze ausgeben. Wenn A teilt, B aber nicht, muss A 5€ ausgeben und B 0€. Die Kosten können der Matrix entnommen werden.

Die Plizsammler:innen gehen jeden Tag los um Pilze zu sammeln und ggf. zu teilen. Welche Strategie zur Entscheidungsfindung ob geteilt werden soll oder nicht ist am erfolgreichsten ? Um diese Frage zu untersuchen entwickeln wir eine Simulation.

Matrix

Kostentabelle	**A teilt**	**A teilt nicht**
B teilt	1/1	5/0
B teilt nicht	0/5	3/3

Table 1: Kosten für Plizsammler:innen

Bitte schreit eure Ergebnise auf. Nutzt hierfür Word, Google Docs oder ähnliches. Ihr dürft die einzelnen Aufgaben gerne mit eigenen Ideen erweitern.

1.1 Aufgabe 1

Gegeben seien die folgenden Klassen. *Picker*, die Basis Klasse für Pilzsammler:innen von welcher *Coop* erbt. *Coop* ist ein/e Pilzsammler:inn welche immer teilt (immer kooperiert).

```
class Picker:                # Basis Klasse (Oberklasse) für Pilzsammler:innen
  def __init__(self, name):
    self.name = name
    self.last = 0  # Letzte durchgeführte Handlung
    self.costs = 0 # Kosten für Pilzsammler:inn

  def inc(self, n):
    """Erhöhe Kosten um n"""
    self.costs += n

class Coop(Picker):          # Teilt die Pilze immer
  def share(self, opp):
    self.last = 0
    return self.last
```

Entwickle folgende weitere Strategien (Pilzsammler:innen Klassen) nach dem Vorbild von *Coop*.

- Pilzsammler:innen Strategie welche nie teilt (Betray).
- Pilzsammler:innen Strategie welche immer abwechselnd teilt oder nicht teilt (Altern).
- Pilzsammler:innen Strategie welche immer genau das tut, was ihr Gegenüber in der letzten Runde getan hat (TitForTat).
- Pilzsammler:innen Strategie welche immer zufällig entscheidet ob sie teilt (Crazy).
- Pilzsammler:innen Strategie welche immer das Gegenteil tut, was ihr Gegenüber in der letzten Runde getan hat.

2 Aufgabe 2

Erstelle verschiedene Gruppen von Plizsammler:innen. Bilde dazu Gruppen mit jeweils 6 Personen. Verwende hierfür eine Liste in welcher Plizsammler:innen verwaltet werden. Unten ist ein Beispiel abgebildet.

```
ni1  = Coop('Nice1' )
ni2 = Coop('Nice2' )
persons = [ni1, ni2]
```

3 Aufgabe 3

Teste verschiedene Konfigurationen von Gruppen und analysiere deine Ergebnisse. Nutze hierfür den unten abgebildeten Code. Versuche die Struktur nachzuvollziehen und die Simulation zu verstehen.

```
from math import *
from random import randrange, seed, shuffle


## Strategien für Pilzsammler:innen

class Picker:                # Basis Klasse (Oberklasse) für Pilzsammler:innen
  def __init__(self, name):
    self.name = name
    self.last = 0  # Letzte durchgeführte Handlung
    self.costs = 0 # Kosten für Pilzsammler:inn

  def inc(self, n):
    """Erhöhe Kosten um n"""
    self.costs += n

class Coop(Picker):          # Teilt die Pilze immer
  def share(self, opp):
    self.last = 0
    return self.last

## Erweitere hier den Code mit deinen Strategien


def simulateDay(a,b,n):
  """Führt die Gegenüberstellung (Teilen oder nicht teilen) aus."""
  seed()
  ascore = bscore = 0
  for i in range(0,n):
    x = a.share(b), b.share(a)
    if x == (1,1):                  # Beide teilen (Kosten 1)
      a.inc(1); b.inc(1)
    elif x == (0,1):                # B teilt nicht (Kosten A = 3, B = 0)
      a.inc(3); b.inc(0)
    elif x == (1,0):                # A teilt nicht (Kosten B = 3, A = 0)
      a.inc(0); b.inc(3)
    elif x == (1,1):                # Beide teilen nicht (Kosten = 2)
      a.inc(2); b.inc(2)
```

```
## Erstellen von Pilzsammler:innen

## Erstelle hier deine Pilzsammler:innen und
## baue eine Liste 'persons' aus ihnen wie in Aufgabe 2.

persons = []      # Liste der Pilzsammler:innen


def simulateDays(days):
  """Simulation von teilenden oder nicht teilenden Pilzsammler:innen
  über mehrere Tage"""
  seed()
  totalpen = 0

  for day in range(0, days): # ausführen der Gegenüberstellung (Teilen oder nicht)
    shuffle(persons)
    simulateDay(persons[0], persons[1], 6)

  results = {} # dictionary der Personen (Namen) und ihren Kosten

  for person in persons:
    results[person.costs] = person.name
    totalpen += person.costs

  k = results.keys()
  print('\n')

  for i in sorted(k):
    print('%8s' % results[i], '\t', '%0i' % i)

  print('\n\nTotal Penalties Suffered: ',totalpen)

## Das Programm startet hier
simulateDays(100000) # Simulieren von 100.000 Tagen
```

4 Aufgabe 4

Welche Strategien sind wann besonders erfolgreich ?

- Welche Strategie ist in einer gemischten Gruppe am erfolgreichsten ?
- In welchen Gruppen ist die Strategie Betray (niemal teilen) am erfolgreichsten ?
- Gibt es Gruppenkonfigurationen, sodass TitForTat Sammler:innen schlechter abschneiden als andere Strategien ? Wenn ja, welche Strategie kann besser abschneiden und wie sieht die Gruppenkonfiguration aus ?

5 Aufgabe 5

Plotte deine Ergbnise der Simulationen als Diagramm. Unter folgender Quelle findest du eine Einführung in Histogramme, sowie ein Schritt-für-Schritt Beispiel.

`https://www.humaneer.org/blog/how-to-plot-a-histogram-using-python-matplotlib/`

Optional : Überlege dir welche Daten noch relevant sein könnten um sie zu plotten. Recherchiere hierzu bei Bedarf im Netz.