**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ **«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ ИМ. В. Г. ШУХОВА»**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

**ЛАБОРАТОРНАЯ РАБОТА №2**

**Дисциплина: Теория надежности**

Выполнил: ст. группы ВТ-31

Подкопаев Антон Валерьевич

Проверил: доц. каф. ПО и ВТАС

Кабалянц Петр Степанович

**Белгород 2020**

**Задания для выполнения к работе**

Пьяница стоит между двумя пропастями, с одной стороны река, с другой копья. В начальный момент времени пьяница стоит на левой ноге. Его поведение задается графом марковского процесса:

1) Необходимо определить среднее время жизни пьяницы и вероятность упасть в реку.

2) Написать программу, которая имитирует поведение пьяницы и выводит среднее количество переходов до падения с утеса и долю падений в реку.

3) Сравнить теоретическую вероятность падению в реку с долей падения в реку критерием сравнения долей.

Изображение выглядит как текст

Автоматически созданное описание

**Ход выполнения работы**

Составим матрицу переходных вероятностей:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Р | К | Л | О | П |
| Р | 1 | 0 | 0 | 0 | 0 |
| К | 0 | 1 | 0 | 0 | 0 |
| Л | 3/13 | 0 | 0 | 10/13 | 0 |
| О | 0 | 0 | 3/14 | 10/14 | 1/14 |
| П | 0 | 1/11 | 0 | 10/11 | 0 |

Найдем фундаментальную матрицу:

, где E - единичная матрица, а Q - правый нижний блок матрицы переходных вероятностей, отвечающий за переходы в непоглощающие состояния из непоглощающих состояний

|  |  |  |
| --- | --- | --- |
| 1 | -10/13 | 0 |
| -3/14 | 4/14 | -1/14 |
| 0 | -10/11 | 1 |

|  |  |  |
| --- | --- | --- |
| 3,95 | 13,75 | 0,98 |
| 3,83 | 17,88 | 1,28 |
| 3,4 | 16,25 | 2,16 |

N =

**Среднее время пребывания пьяницы на ногах**, если он начал с левой будет 3,83 + 17,88 + 1,28 = 22,99 (23 шага)

Найдем матрицу B:

B = N \* R, где R - левый нижний блок матрицы переходных вероятностей, отвечающий за переходы в поглощающие состояния из непоглощающих

|  |  |
| --- | --- |
| 3/13 | 0 |
| 0 | 0 |
| 0 | 1/11 |

R =

|  |  |
| --- | --- |
| 0,91 | 0.09 |
| 0.88 | 0.12 |
| 0.81 | 0.19 |

B =

**Вероятность пьяницы упасть в реку** - 0.91, значит, из 1000 попыток 910 должны закончиться в реке.

Проведем эксперимент 1000 раз:

Среднее время пребывания пьяницы на ногах, если он начал с левой - 23,73

Пьяница упал в реку 896 раз, а на копья 104 разa.

Проверим гипотезу о том, что экспериментальная вероятность поглотиться в реку (S0) соответствует теоретической вероятности:

Изображение выглядит как снимок экрана

Автоматически созданное описание

*Приложение*

# This Python file uses the following encoding: utf-8

import random

import math

import numpy

from prettytable import PrettyTable

import matplotlib.pyplot as pyplot

i = 3

j = 10

k = 1

lr = i / (i + j)

lo = j / (i + j)

ol = i / (i + j + k)

oo = j / (i + j + k)

op = k / (i + j + k)

po = j / (j + k)

pk = k / (j + k)

p0 = [0, 0, 0, 1, 0]

# Матрица переходных вероятностей

# Р К Л О П

P = [[1, 0, 0, 0, 0 ], # Р

[0, 1, 0, 0, 0 ], # К

[lr, 0, 0, lo, 0 ], # Л

[0, 0, ol, oo, op], # О

[0, pk, 0, po, 0 ]] # П

Q = [[0, lo, 0 ],

[ol, oo, op],

[0, po, 0 ]]

R = [[lr, 0],

[0, 0],

[0, pk]]

# Вероятности перехода из одного положения в другое: из текущего / в текущее

stepProbs = [[0, 0, lr],

[lr, 0, lo],

[ol, oo, op],

[po, 0, pk],

[0, 0, pk]]

def oneDrunkStep(currentState):

# Возвращает новое состояние любителя спиртного

if currentState == 0 or currentState == 4:

return currentState # Из реки и копий уже не выберешься

r = random.random()

if r < stepProbs[currentState][0]:

return currentState - 1

if r > stepProbs[currentState][0] and r < stepProbs[currentState][1] + stepProbs[currentState][0]:

return currentState

else:

return currentState + 1

if \_\_name\_\_ == '\_\_main\_\_':

print("Матрица переходных вероятностей:")

print(P)

N = (numpy.eye(3) - Q)

N = numpy.linalg.inv(N)

B = N @ R

r = 0 # Сколько раз упал в реку

s = 0 # Сколько раз упал на копья

k = 0 # Количество шагов за одну жизнь

averageSteps = 0 # Среднее количество шагов

state = 2 # Текущее положение тунеядца, алкоголика и дебошира

sz = 1000 # Сколько раз неприкаянная душа любителя качественно выпить и грамотно закусить будет возвращатся в его бренное тело, стоящее на левой ноге

for i in range (0, sz):

while state != 0 and state != 4:

state = oneDrunkStep(state)

k += 1

if state == 4:

s += 1

else:

r += 1

averageSteps += k

k = 0

state = 2

averageSteps /= sz

print("Проведем эксперимент {} раз".format(sz))

print("Среднее количество шагов за каждую реинкарнацию любителя спиртного: {}".format(round(averageSteps, 2)))

print("При этом упал в реку {} раз, а на копья {} раз".format(r, s))

#print("Сравним с теоретическими значениями.")

#print("Должен был упасть в реку {} раз, а на копья {} раз.".format(int(B[1][0]\*sz), int(B[1][1]\*sz)))

#print("Cреднее количество шагов: {}.".format(round(numpy.sum([N[1][i] for i in range(0, 3)]), 2)))

K\_t = 1.64 # 6 степеней свободы

print("Проверка гипотезы о том, что экспериментальная вероятность поглотиться в S\_0 (река), начиная с S\_1 (левая нога) соотвествует теоретическому значению:")

p0 = math.sqrt(B[2][0])

K\_v = (r / sz - p0) / (math.sqrt(p0 \* (1 - p0) / 6))

print("К\_в = {}, К\_т: {}\n{}".format(round(K\_v, 2), K\_t, "Гипотеза верна!" if abs(K\_v) < K\_t else "Гипотезу следует отбросить!"))