Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ
по курсу «Моделирование»

«Марковские процессы»

Студент: ИУ7-73Б _____ М. Д. Маслова
(группа) (подпись, дата) (И. О. Фамилия)

Преподаватель: _____ И. В. Рудаков
(подпись, дата) (И. О. Фамилия)

2022 г.

СОДЕРЖАНИЕ

1 Задание

Разработать программное обеспечение, предоставляющее возможность определения вероятности и времени пребывания системы массового обслуживания в каждом состоянии в установившемся режиме работы.

Реализовать графический интерфейс, позволяющий задать количество состояний системы (их не более десяти) и матрицу интенсивностей переходов.

2 Теоретическая часть

2.1 Марковские процессы

Случайный процесс, протекающий в некоторой системе S, называется ***марковским***, если для каждого момента времени вероятность любого состояния системы в будущем зависит только от ее состояния в настоящем времени и не зависит от того, когда и каким образом система пришла в это состояние, то есть не зависит от того, как процесс развивался в прошлом.

2.2 Предельные вероятности состояний

Для марковских процессов используются уравнения Колмогорова, составляющиеся по следующему правилу:

1. В левой части каждого уравнения стоит производная вероятности состояния.
2. Правая часть содержит столько членов, сколько стрелок связано с этим состоянием; если стрелка направлена из состояния соответствующий член имеет знак «-», если в состояние — знак «+».
3. Каждый член равен плотности вероятности перехода (интенсивности), соответствующей данной стрелке, умноженной на вероятность того состояния, из которого исходит стрелка.

То есть строится система уравнений, которые имеют вид:

$$P_i'(t) = \sum_{j=1}^{n} \lambda_{ji} P_j(t) - P_i(t) \sum_{j=1}^{n} \lambda_{ij}, \tag{2.1}$$

где $P_i(t)$ – вероятность того, что система находится в i-ом состоянии;
n — число состояний в системе;
λ_{ij} — интенсивность перехода системы из i-ого состояния в j-ое.
Одно из уравнений данной системы заменяется условием нормировки:

$$\sum_{i=1}^{n} P_i(t) = 1. \tag{2.2}$$

В силу того, что ***предельные вероятности состояний постоянны***, для их определения в уравнениях Колмогорова необходимо ***заменить их про-***

изводные нулями и решить полученную систему линейных алгебраических уравнений.

Отметим, что предельная вероятность состояния показывает ***среднее относительное время пребывания*** системы в этом состоянии.

2.3 Точки стабилизации

Для определения точек стабилизации системы определяются вероятности состояний с некоторым малым шагом Δt. Точка стабилизации считается найденной, если приращение вероятности, а также разница между ранее найденной предельной вероятностью состояния и вычисленной вероятностью, достаточно малы, то есть выполняются соотношения:

$$|P_i(t + \Delta t) - P_i(t)| < \varepsilon, \tag{2.3}$$

$$|P_i(t) - \lim_{t\to\infty} P_i(t)| < \varepsilon, \tag{2.4}$$

где ε — заданная точность.

3 Практическая часть

3.1 Текст программы

На листинге 3.1 представлены функции расчета предельных вероятностей по заданной матрице интенсивностей переходов.

Листинг 3.1 – Реализация функций расчета предельных вероятностей

```
import numpy as np
import scipy as sp

def GetKolmogorovEquations(matrix):
    statesNumber = matrix.shape[0]

    rightSide = np.zeros([statesNumber] * 2)

    for state in range(statesNumber):
        rightSide[state][state] = -sum(matrix[state, :])
        rightSide[state] += matrix[:, state]

    return rightSide


def GetNormalizationEquation(statesNumber):
    return 1, np.zeros(statesNumber) + 1


def CalculateMarginalProbabilities(matrix):
    statesNumber = matrix.shape[0]

    leftSide = np.zeros(statesNumber)
    rightSide = GetKolmogorovEquations(matrix)

    leftSide[-1], rightSide[-1] = GetNormalizationEquation(
       statesNumber)

    return sp.linalg.solve(rightSide, leftSide)
```

На листинге 3.2 представлены функции расчета точек стабилизации.

Листинг 3.2 – Реализация функций расчета точек стабилизации

```
def GetProbabilitiesDerivatives(probabilities, _, matrix):
    derivatives = np.zeros(len(probabilities))

    for i, probability in enumerate(probabilities):
        derivatives[i] = np.dot(probabilities, matrix[i, :])

    return derivatives


def CalculateStabilizationTime(matrix, marginalProbabilities):
    timeList = np.arange(0, TIME_MAX, TIME_STEP)

    statesNumber = matrix.shape[0]
    probabilities0 = np.array([0] * statesNumber)
    probabilities0[0] = 1

    kolmogorovMatrix = GetKolmogorovEquations(matrix)
    probabilities = sp.integrate.odeint(
                                GetProbabilitiesDerivatives,
                                probabilities0,
                                timeList,
                                (kolmogorovMatrix,))

    stabilizationTimes = np.zeros(statesNumber)

    for state in range(statesNumber):
        curStateProbabilities = probabilities[:, state]

        found = False

        for i in range(1, len(curStateProbabilities)):
            previousProbability = curStateProbabilities[i - 1]
            curProbability = curStateProbabilities[i]

            if (abs(curProbability - previousProbability) < EPS
               and abs(curProbability
                       - marginalProbabilities[state]) < EPS):
                found = True
                break

        stabilizationTimes[state] = (timeList[i] if found
                                                 else None)
```

3.2 Полученный результат

На рисунке 3.1 представлен пример работы программы для системы с тремя состояниями. На рисунке 3.2 представлен соответствующий график зависимости вероятности от времени.

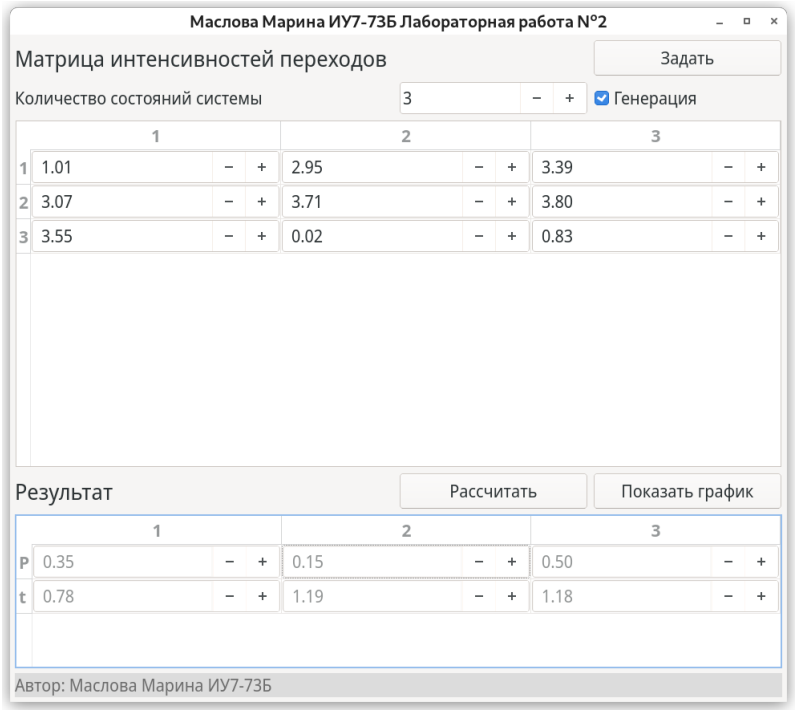

Рисунок 3.1 – Расчет для системы с тремя состояниями

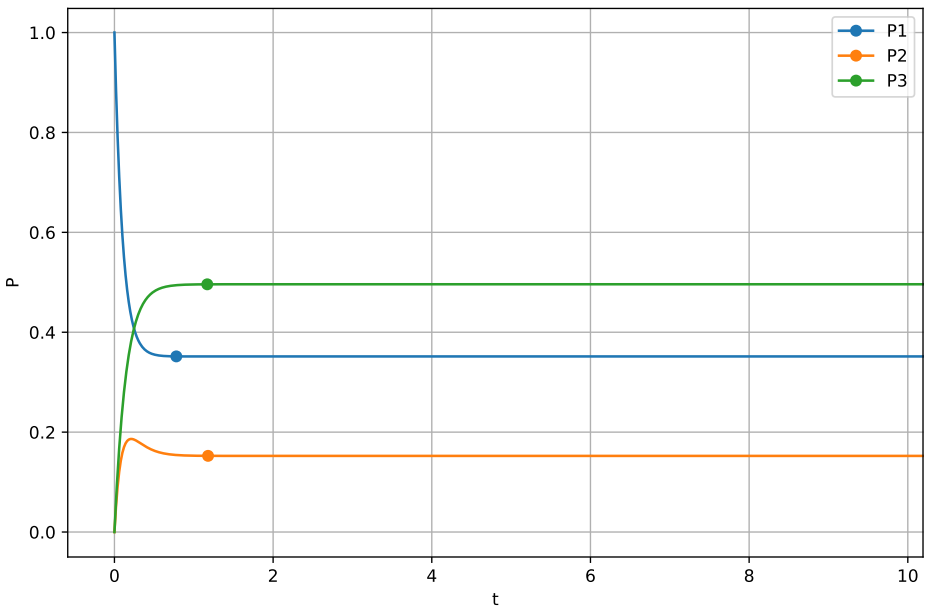

Рисунок 3.2 – График зависимости вероятности состояния от времени для системы с тремя состояниями

На рисунке 3.3 представлен пример работы программы для системы с пятью состояниями. На рисунке 3.4 представлен соответствующий график зависимости вероятности от времени.

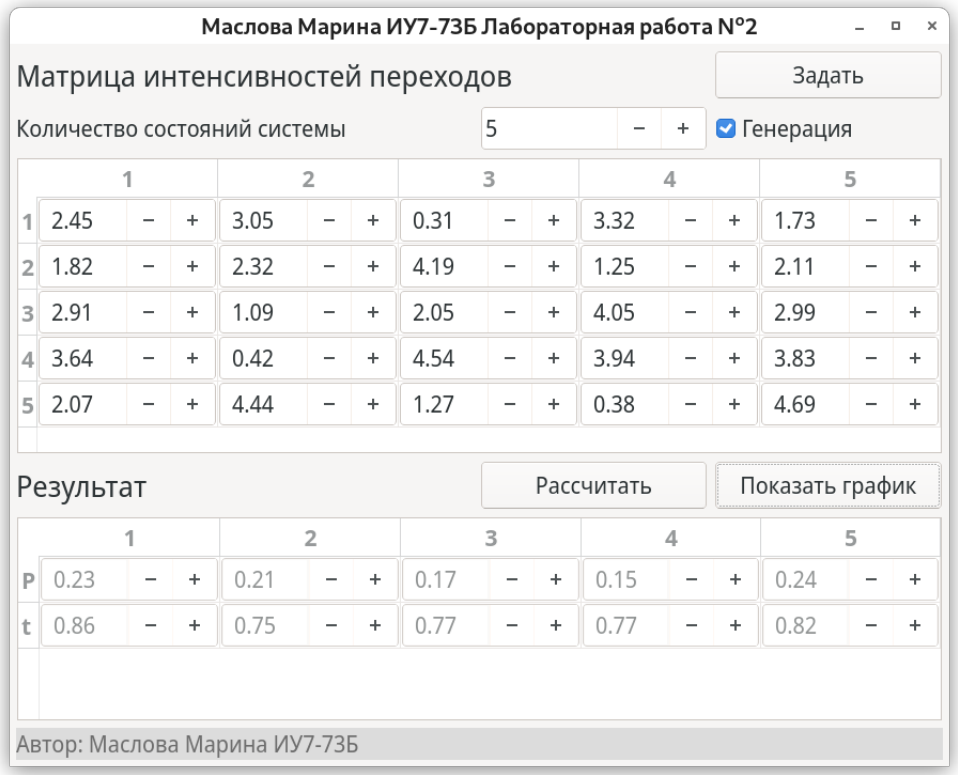

Рисунок 3.3 – Расчет для системы с пятью состояниями

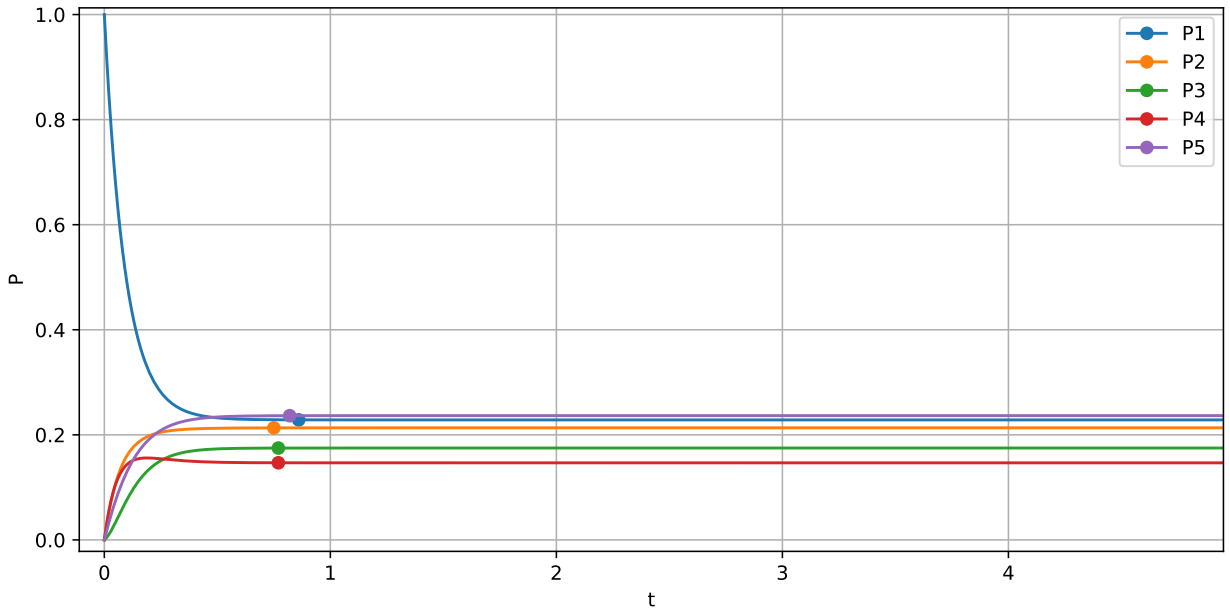

Рисунок 3.4 – График зависимости вероятности состояния от времени для системы с пятью состояниями