Отчет по лабораторной работе №1

Тема: Полнодоступная двухсервисная модель Эрланга с одинаковыми интенсивностями обслуживания

Студент	Ким Реачна
Группа	НПИбд-01-20
Преподаватели:	Доцент Маркова Е.В.
Количество баллов:	______ баллов из 20 б.

Содержание

Список иллюстраций

1 Теоретические сведения

Исследуется сота сети связи емкостью C. Пусть пользователям сети предоставляются услуги двух типов. Запросы в виде двух пуассоновский потоков (ПП) с интенсивностями λ_1, λ_2 поступают в соту. Среднее время обслуживания запросов на предоставление услуг каждого типа μ_1^{-1}, μ_2^{-1} соответственно. Исследуются основные характеристики модели для случая $\mu_1 = \mu_2 = \mu$.

В классификации Башарина-Кендалла $MM|MM|C|0$.

Основные обозначения:

- C - пиковая пропускная способность соты;
- λ_1, λ_2 - интенсивность поступления запросов на предоставление услуги 1, 2-го типа [запросов/ед.вр.];
- μ^{-1} - среднее время обслуживания запроса на предоставление услуги 1, 2-го типа [запросов/ед.вр.];
- ρ_1, ρ_2 - интенсивность предложенной нагрузки,создаваемой запросами на предоставление услуги 1, 2-го типа;
- $X(t)$ - число запросов, обслуживаемых в системе в момент времени $t, t \geq 0$ (случайный процесс (СП), описывающий функционирование системы в момент времени $t, t \geq 0$);
- X - пространство состояний системы;
- n - число обслуживаемых в системе запросов;
- B_1, B_2 - множество блокировок запросов на предоставление услуги 1, 2-го типа;
- S_1, S_2 - множество приема запросов на предоставление услуги 1, 2-го типа.

Схема модели (рис. 1.1):

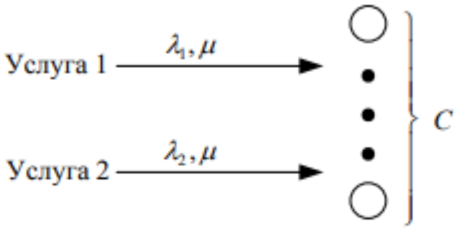

Рис. 1.1: Схема полнодоступной двухсервисной модели Эрланга с одинаковыми интенсивностями обслуживания

Пространство состояний системы (рис. 1.2):

$$X = 0, ..., C, |X| = C + 1 \tag{1.1}$$

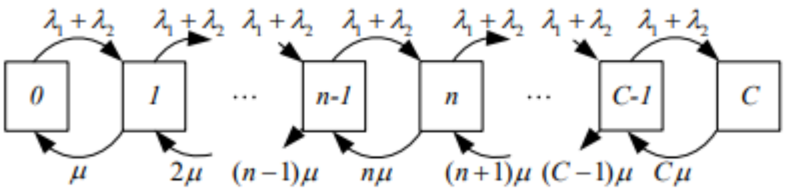

Рис. 1.2: Диаграмма интенсивностей переходов для полнодоступной двухсервисной модели Эрланга с одинаковыми интенсивностями обслуживания

Множество блокировок запросов на предоставление услуги i-типа, i=1,2:

$$B_1 = B_2 = C \tag{1.2}$$

Множество приема запросов на предоставление услуги i-типа, i=1,2:

$$S_i = \bar{B}_i = X \backslash B_i = \{0, 1, , , ., C - 1\} \tag{1.3}$$

Система уравнений глобального баланса (СУГБ):

$$\begin{cases}(\lambda_1+\lambda_2)p_0=\mu p_1,\\(\lambda_1+\lambda_2+n\mu)p_n=(\lambda_1+\lambda_2)p_{n-1}+(n+1)\mu p_{n+1}, n=\overline{1,C-1},\\C\mu p_C=(\lambda_1+\lambda_2)p_{C-1}\end{cases} \tag{1.4}$$

Система уравнений локального баланса (СУЛБ):

$$(\lambda_1+\lambda_2)p_{n-1}=n\mu p_n, n=\overline{1,C} \tag{1.5}$$

Стационарное распределение вероятностей состояний системы:

$$p_n=\left(\sum_{i=0}^{C}\frac{(\rho_1+\rho_2)^i}{i!}\right)^{-1}.\frac{(\rho_1+\rho_2)^n}{n!}, n=\overline{0,C} \tag{1.6}$$

Доказательство:

Используя СУЛБ, найдем стационарное распределение вероятностей состояний системы $p_n, n=\overline{1,C}$:

$$p_n[C]=p_n=p_{n-1}.\frac{\lambda_1+\lambda_2}{n\mu}=p_{n_1}.\frac{\rho_1+\rho_2}{n}=...=p_0.\frac{(\rho_1+\rho_2)^n}{n!}, n=\overline{1,C}$$

Для нахождения вероятности p_0 воспользуемся условием нормировки $\sum_{n=0}^{C}p_n=1$:

$$p_0=p_0[C]=\left(\sum_{n=0}^{C}\frac{(\rho_1+\rho_2)^n}{n!}\right)^{-1}$$

Основные вероятностные характеристики (ВХ) модели:

- Вероятность блокировки по времени E_i запроса на предоставление услуги i-типа, $i=1,2$:

$$E_1 = E_2 = E[C] = \sum_{n \in B_i} p_n = p_C[C] \tag{1.7}$$

- Вероятность блокировки по вызовам B_i запроса на предоставление услуги i-типа, $i = 1, 2$:

$$B_i = \frac{\lambda_i}{\lambda_1 + \lambda_2} . E[C], \tag{1.8}$$

где $\frac{\lambda_i}{\lambda_1+\lambda_2}$ — вероятность того, что поступит запрос на предоставление услуги i -типа;

- Вероятность блокировки по нагрузке C_i запроса на предоставление услуги i-типа, $i = 1, 2$:

$$C_1 = C_2 = E[C] \tag{1.9}$$

- Среднее число $\bar{N}$ обслуживаемых в системе запросов:

$$\bar{N} = \sum_{n \in X} n p_n . \tag{1.10}$$

2 Численный анализ

Создание программы, реализующей расчет распределения вероятностей, вероятности блокировки, среднего количества обслуженных запросов для любых значений исходных данных. Программа должна выводить на экран:

- значение распределения вероятностей;
- значения вероятностей блокировки;
- значение среднего числа заявок.

Код написан на языке Python в Google Colab:

```
[1]  import math #for power and factorial function
     import matplotlib.pyplot as plt #for data visualization
     import numpy as np #for array computing
```

Рис. 2.1: Импортируйте необходимую библиотеку python

Определение функции для вычисления вероятности того, что система находится в исходном состоянии p_0:

```
[2]  def p_0(C, rho):
         # rho = rho1+rho2
         probab = 0
         if C != 0:
             sum = 0
             for n in range(C):
                 sum += math.pow(rho, n)/math.factorial(n)
             probab = 1/sum
         return probab
```

Рис. 2.2: Определение функции для вычисления вероятности p_0

Определение функции для вычисления вероятности того, что система находится в n состоянии, т.е. на сервере обрабатывается n запросов p_n:

[3]
```
def p_n(C, n, rho):
    p_nC = ((math.pow(rho, n) / math.factorial(n))) * p_0(C=C, rho=rho)
    return p_nC
```

Рис. 2.3: Определение функции для вычисления вероятности p_n

Определение функции для вычисления вероятности того, что по крайней мере один запрос заблокирован, т.е. сервер заполнен p_C:

[4]
```
def p_C(C, rho):
    # compute E, C1, C2
    return p_n(C=C, n=C, rho=rho)
```

Рис. 2.4: Определение функции для вычисления вероятности p_C

Определение функций для вычисления среднего число запросов в системе $\bar{N}$:

[5]
```
def mean_request(C, rho):
    expecation = 0
    for n in range(C):
        expecation += n * p_n(C=C, n=n, rho=rho)
    return expecation
```

Рис. 2.5: Определение функции для вычисления среднего число

Для расчета основных вероятностных характеристик модели были взяты следующие параметры:

$$C = 100, \mu_1 = \mu_2 = \mu = 0.8, \lambda_1 = 40, \lambda_2 = 60 \tag{2.1}$$

Заметить что:

$\rho_1 = \lambda_1 / \mu_1$

$\rho_2 = \lambda_2 / \mu_2$

$\rho = \rho_1 + \rho_2$ может варьироваться от 0 до $\lambda_1/\mu_1 + \lambda_2/\mu_2 = 125$

```
[6]
C = 100
mu = 0.8
lambda_1 = 60
lambda_2 = 40
```

Рис. 2.6: Основные вероятностные характеристики

Тестирование нескольких значений n = 30, 33, ..., 50 с увеличением на 3:

```
[7]
Rho = (lambda_1 + lambda_2) / mu
for n in np.arange(30, 50, 3):
    print("Вероятность того, что система находится в {}-ый состоянии, равна {}".format(n, p_n(C=C, n=n, rho=Rho)))
```

```
Вероятность того, что система находится в 30-ый состоянии, равна 1.6775189879197e-22
Вероятность того, что система находится в 33-ый состоянии, равна 1.0008566328447777e-20
Вероятность того, что система находится в 36-ый состоянии, равна 4.563020800711849e-19
Вероятность того, что система находится в 39-ый состоянии, равна 1.6252963492341122e-17
Вероятность того, что система находится в 42-ый состоянии, равна 4.608604721396451e-16
Вероятность того, что система находится в 45-ый состоянии, равна 1.0572211764713933e-14
Вероятность того, что система находится в 48-ый состоянии, равна 1.9897520720548972e-13
```

Рис. 2.7: Распределение вероятностей для n = 30,33,...,50 с увеличением на 3

```
[8]
array_p_n = np.array([], 'float64')
for n in range(C):
    array_p_n = np.append(array_p_n, p_n(C=C, n=n, rho=Rho))
print("Вероятность нахождения системы в каждом состоянии: ", array_p_n)
```

```
Вероятность нахождения системы в каждом состоянии:  [5.50842105e-53 6.88552631e-51 4.30345395e-49 1.79310581e-47
 5.60345566e-46 1.40086391e-44 2.91846649e-43 5.21154730e-42
 8.14304266e-41 1.13097815e-39 1.41372268e-38 1.60650305e-37
 1.67344068e-36 1.60907757e-35 1.43667641e-34 1.19723034e-33
 9.35336201e-33 6.87747207e-32 4.77602227e-31 3.14211991e-30
 1.96382495e-29 1.16894342e-28 6.64172398e-28 3.60963260e-27
 1.88001698e-26 9.40008489e-26 4.51927158e-25 2.09225536e-24
 9.34042572e-24 4.02604557e-23 1.67751899e-22 6.76418947e-22
 2.64226151e-21 1.00085663e-20 3.67961997e-20 1.31414999e-19
 4.56302080e-19 1.54156108e-18 5.07092461e-18 1.62529635e-17
 5.07905109e-17 1.54849119e-16 4.60860472e-16 1.33971067e-15
 3.80599624e-15 1.05722118e-14 2.87288363e-14 7.64064796e-14
 1.98975207e-13 5.07589814e-13 1.26897454e-12 3.11023171e-12
 7.47651852e-12 1.76332984e-11 4.08178204e-11 9.27677736e-11
 2.07070923e-10 4.54102902e-10 9.78670047e-10 2.07345349e-09
 4.31969477e-09 8.85183354e-09 1.78464386e-08 3.54096004e-08
 6.91593757e-08 1.32998800e-07 2.51891666e-07 4.69947138e-07
 8.63873415e-07 1.56498807e-06 2.79462155e-06 4.92010837e-06
 8.54185481e-06 1.46264637e-05 2.47068644e-05 4.11781073e-05
 6.77271502e-05 1.09946672e-04 1.76196590e-04 2.78792073e-04
 4.35612614e-04 6.72241689e-04 1.02475867e-03 1.54331125e-03
 2.29659413e-03 3.37734430e-03 4.90893067e-03 7.05306131e-03
 1.00185530e-02 1.40710014e-02 1.95430575e-02 2.68448592e-02
 3.64739935e-02 4.90241848e-02 6.51917351e-02 8.57785989e-02
 1.11690884e-01 1.43931551e-01 1.83586163e-01 2.31800710e-01]
```

Рис. 2.8: Распределение вероятностей для каждого n

```
[9]   1 sum(array_p_n)

1.0000000000000002
```

Рис. 2.9: Провека через нахождение общей суммы

Вероятность блокировки по времени E:

```
[10]  1 # just apply the function p_C defined above
      2 E = p_C(C=C, rho=Rho)
      3 print("Вероятность блокировки по времени E =", E)

Вероятность блокировки по времени E = 0.2897508878816568
```

Рис. 2.10: Вероятность блокировки по времени

Вероятность блокировки по вызовам:

```
[11]  1 # use the relation between B1, B2 with E as given in lecture
      2 B_1 = (lambda_1 / (lambda_1 + lambda_2)) * E
      3 B_2 = (lambda_2 / (lambda_1 + lambda_2)) * E
      4 print("B1 = {}, B2 = {}".format(B_1, B_2))

B1 = 0.1738505327289941, B2 = 0.11590035515266273
```

Рис. 2.11: Вероятность блокировки по вызовам:

Среднее число обслуживаемых в системе запросов:

```
[12]  1 N = mean_request(C=C, rho=Rho)
      2 print("Среднее число N обслуживаемых в системе запросов = ", N)

Среднее число N обслуживаемых в системе запросов =  96.02491121183434
```

Рис. 2.12: Среднее число обслуживаемых запросов

- Построение график зависимости вероятности блокировки от интенсивности поступления запросов на обслуживание

```
[13]
# Incrementation of lambda between 0 and C, with incremental step of 0.4
lambda__ = np.arange(0, lambda_1+lambda_2, 0.4)

# array_E of p_C when lmabda change
rho = lambda__ / mu
array_E = []
for ro in rho:
    array_E.append(p_C(C=C, rho=ro))
plt.plot(lambda__, array_E, color='blue', label='$E=p_C$')
plt.title("Вероятность блокировки по времени и вероятность блокировки по нагрузке в зависимости от интенсивности запросов")
plt.xlabel('Интенсивность поступления запросов $\lambda_1+\lambda_2$')
plt.ylabel('Значение вероятность')
plt.legend()
plt.show()
```

Рис. 2.13: Построение график вероятностей блокировки по времени и нагрузке

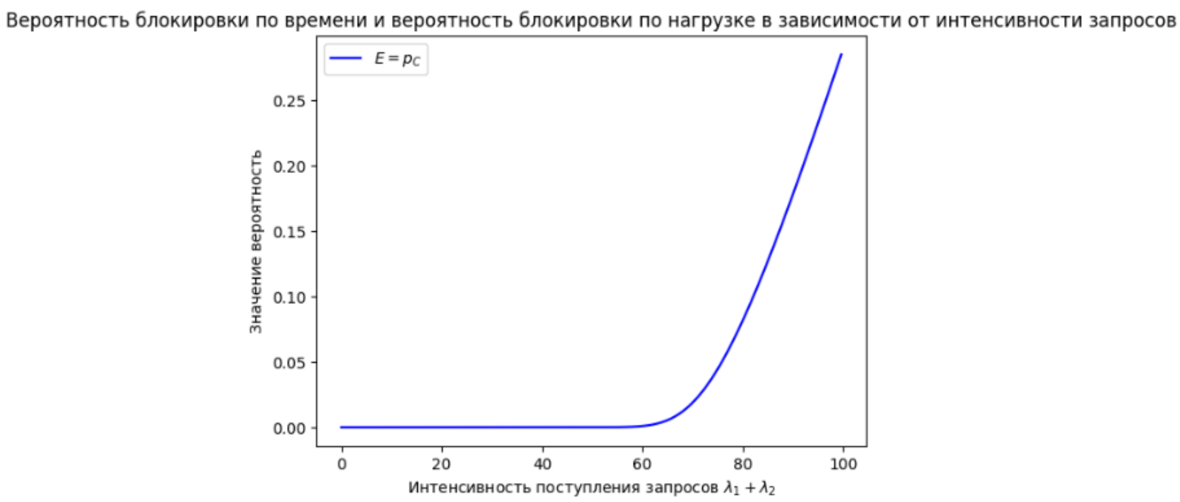

Рис. 2.14: График вероятностей блокировки по времени и нагрузке

```
[14]
# array_E of p_C when lmabda change
B_1_array = []
for lam, E in zip(lambda__, array_E):
    B_1_array.append(lam / (lam + lambda_2))
B_2_array = []
for lam, E in zip(lambda__, array_E):
    B_2_array.append(lambda_2 / (lam + lambda_2))
plt.plot(lambda__, B_1_array, color= 'blue', label="$B_1$")
plt.plot(lambda__, B_2_array, color= 'green', label="$B_2$")
plt.title("Вероятность блокировки по вызовам в зависимости от интенсивности запросов")
plt.xlabel('Интенсивность поступления запросов $\lambda_1+\lambda_2$')
plt.ylabel('Вероятность блокировки')
plt.legend()
plt.show()
```

Рис. 2.15: Построение график вероятностей блокировки по вызовам

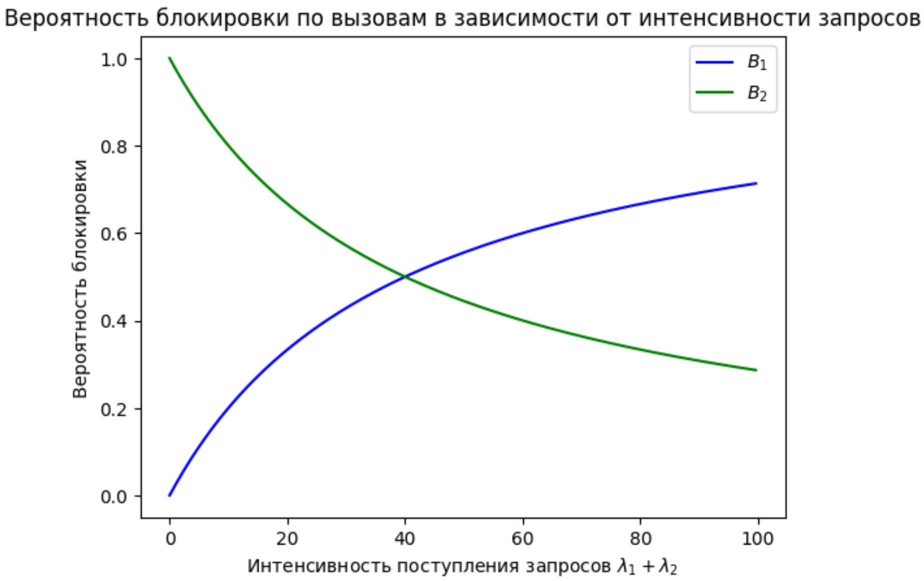

Рис. 2.16: График верояностей блокировки по вызовам

- Построение график среднего числа обслуживаемых запросов от интенсивности поступления запросов:

[15]

```
mean_request_array = []
rho_ = lambda__ / mu
for rho in rho_:
    mean_request_array.append(mean_request(C=C, rho=rho))

plt.plot(rho_, mean_request_array, color="green", label = 'N')
plt.title("Среднее число обслуживаемых в зависимости от интенсивности запросов")
plt.xlabel('Интенсивность поступления запросов $\lambda_1 + \lambda_2$')
plt.ylabel('Среднее число запросов')
plt.legend()
plt.show()
```

Рис. 2.17: Построение график среднего числа обслуживаемых запросов

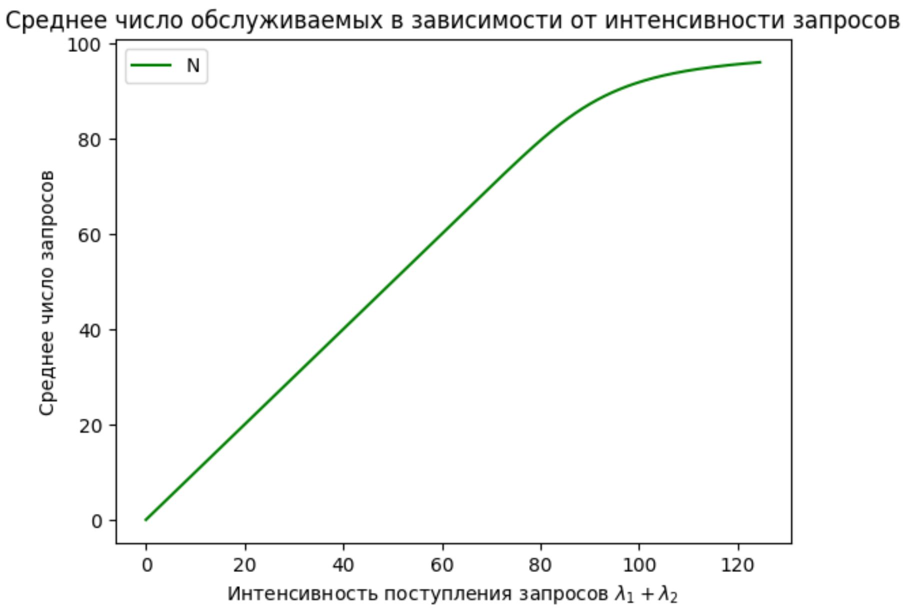

Рис. 2.18: График среднего числа обслуживаемых запросов