Отчет по лабораторной работе №2

Тема: Неполнодоступная двухсервисная модель Эрланга с одинаковыми интенсивностями обслуживания и зарезервированной емкостью

Выполнила: Ким Реачна

Содержание

1 Теоретические сведения 4

2 Численный анализ 9

Список иллюстраций

1.1 Схема неполнодоступной двухсервисной модели Эрланга с одинаковыми интенсивностями обслуживания и зарезервированной емкостью 5
1.2 Диаграмма интенсивностей переходов для неполнодоступной двухсервисной модели Эрланга с одинаковыми интенсивностями обслуживания и зарезервированной емкостью 6

2.1 Импортируйте библиотеку 9
2.2 Функция рассчета стационарного распределения вероятностей состояний систем p_0 и p_n 10
2.3 Начальные значения 10
2.4 Стационарное распределение вероятностей системы в исходном состоянии p_0 10
2.5 Стационарное распределение вероятностей для каждого n от 0 до C и сумма вероятностей 11
2.6 Вероятность блокировки по времени E_1 11
2.7 Вероятность блокировки по времени E_2 11
2.8 Среднее число обслуживаемых запросов $\bar{N}$ 12
2.9 Построение графика зависимости вероятности блокировки по времени от интенсивности поступления запросов E_1, E_2 12
2.10 Среднее число обслуживаемых запросов $\bar{N}$ 13

1 Теоретические сведения

Рассмотрим звено сети емкостью C. Пусть пользователям сети предоставляются услуги двух типов. Запросы на предоставление услуг представляют собой (ПП) с интенсивностями λ_1, λ_2. Среднее время обслуживания запросов каждого типа μ_1^{-1}, μ_2^{-1} соответственно. Рассмотрим случай $\mu_1 = \mu_2 = \mu$.

Часть пропускной способности соты зарезервирована для обслуживания запросов на предоставление услуги 1-го или 2-го типа. Оставшаяся часть пропускной способности является полнодоступной для запросов на предоставление услуг обоих типов. Предположим, что сначала заполняется полнодоступная емкость.

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

Основные обозначения:

- C - пиковая пропускная способность соты;
- g - полнодоступная часть пропускной способности соты;
- $C - g$ - пропускная способность, зарезервированная для обслуживания запросов на предоставление услуги 1-го или 2-го типа;
- 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.

Пусть часть пропускной способности соты $C - g$ зарезервирована для обслуживания запросов на предоставление услуги 1-го типа.

Схема модели (рис. 1.1):

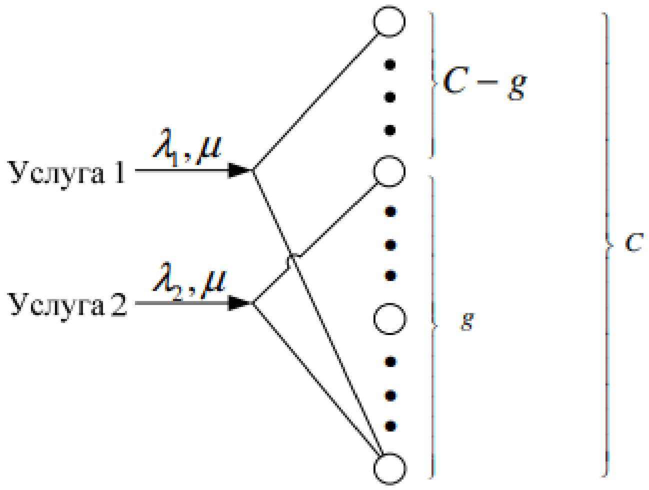


Рис. 1.1: Схема неполнодоступной двухсервисной модели Эрланга с одинаковыми интенсивностями обслуживания и зарезервированной емкостью

Пространство состояний системы (рис. 1.2):

$$X = \{0, ..., C\}, |X| = C + 1 \tag{2.1}$$

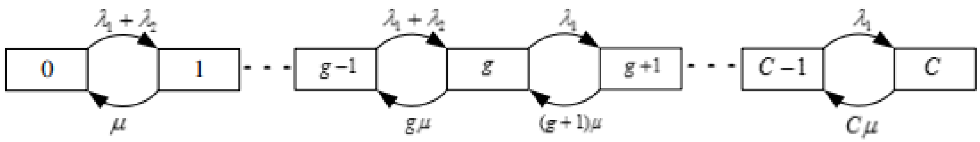


Рис. 1.2: Диаграмма интенсивностей переходов для неполнодоступной двухсервисной модели Эрланга с одинаковыми интенсивностями обслуживания и зарезервированной емкостью

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

$$B_1 = \{C\} \tag{2.2}$$

$$B_2 = \{g, g+1, ..., C\} \tag{2.3}$$

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

$$S_1 = \bar{B_1} = X \backslash B_1 = \{0, ..., C-1\} \tag{2.4}$$

$$S_2 = \bar{B_2} = X \backslash B_2 = \{0, ..., C_2-1\} \tag{2.5}$$

Система уравнений глобального баланса (СУГБ):

$$\begin{cases} (\lambda_1+\lambda_2)p_0 = \mu p_1, \\ (\lambda_1+\lambda_2+n\mu)p_n = (\lambda_1+\lambda_2)p_{n-1} + (n+1)\mu p_{n+1}, n = \overline{1, g-1}, \\ (\lambda_1+g\mu)p_g = (\lambda_1+\lambda_2)p_{g-1} + (g+1)\mu p_{g+1}, \\ (\lambda_1+n\mu)p_n = \lambda_1 p_{n-1} + (n+1)\mu p_{n+1}, n = \overline{g+1, C-1}, \\ C\mu p_C = \lambda_1 p_{C-1} \end{cases} \tag{2.6}$$

Система уравнений локального баланса (СУЛБ):

$$\begin{cases} (\lambda_1 + \lambda_2)p_{n-1} = n\mu p_n, n = \overline{1, g}, \\ \lambda_1 p_{n-1} = n\mu p_n, n = \overline{g+1, C} \end{cases} \tag{2.7}$$

Обозначим:

$$\rho_1 = \frac{\lambda_1}{\mu}, \rho_2 = \frac{\lambda_2}{\mu}$$

Стационарное распределение вероятностей состояний системы:

$$p_n = \begin{cases} p_0 . \frac{(\rho_1+\rho_2)^n}{n!}, n = \overline{1, g} \\ p_0 . \frac{(\rho_1+\rho_2)^g . (\rho_1)^{n-g}}{n!}, n = \overline{g+1, C} \end{cases} \tag{2.8}$$

где:

$$p_0 = \left(\sum_{n=1}^{g} \frac{(\rho_1 + \rho_2)^n}{n!} + \sum_{n=g+1}^{C} \frac{(\rho_1 + \rho_2)^g . (\rho_1)^{n-g}}{n!} \right)^{-1} \tag{2.9}$$

Основные вероятностные характеристики модели:

- Вероятность блокировки по времени E_1 запроса на предоставление услуги 1-го типа

$$E_1 = \sum_{n \in B_1} p_n = p_C \tag{2.10}$$

- Вероятность блокировки по времени E_1 запроса на предоставление услуги 2-го типа

$$E_2 = \sum_{n \in B_2} p_n = p_g + p_{g+1} + ... + p_C = \sum_{n=g}^{C} p_n \tag{2.11}$$

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

$$\bar{N} = \sum_{n \in X} n p_n \tag{2.12}$$

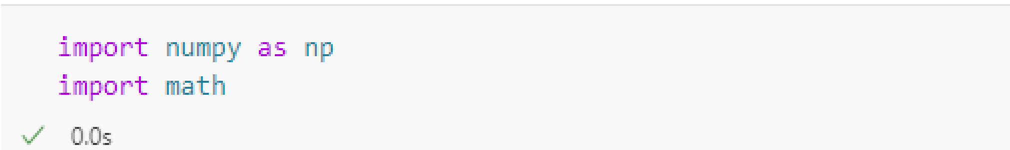
2 Численный анализ

Для расчета основных вероятностных характеристик модели были взяты следующие параметры:

$$C = 50, g = 35, \mu_1 = \mu_2 = \mu = 0.8, \lambda_1 = 30, \lambda_2 = 20 \tag{2.13}$$

Код написан на языке Python в Google Colab:

`python Import library import numpy as np import math`



```
import numpy as np
import math
✓ 0.0s
```

Рис. 2.1: Импортируйте библиотеку

```python
def probability_0(rho1, rho2, g, C):
    '''
    Определите функцию для вычисления вероятности того, что система находится в состоянии 0, то есть все серверы системы бездействуют или свободны.
    '''
    sum_  = 0
    for n in np.arange(C+1):
        if n <= g:
            sum_ += math.pow(rho1 + rho2, n) / math.factorial(n)
        else:
            sum_ += math.pow(rho1 + rho2, g) * math.pow(rho1, n - g) / math.factorial(n)
    return 1 / sum_

def probability_n(n, rho1, rho2, g, C):
    '''
    Определенная функция для вычисления вероятности стационарности системы, если система стационарна в состоянии n, где 1 <= n <= C.
    '''
    p_n = 0 # initialize p_n
    p_0 = probability_0(rho1=rho1, rho2=rho2, g=g, C=C)
    if n == 0:
        # print("n = 0")
        p_n = p_0
    elif n < g + 1:
        # print("1 to g")
        p_n = p_0 * math.pow(rho1 + rho2, n) / math.factorial(n)
    elif n < C +1:
        # print("g+1 to C")
        p_n = (p_0 * math.pow(rho1 + rho2, g) * math.pow(rho1, n - g)) / math.factorial(n)
    return p_n
```
✓ 0.0s

Рис. 2.2: Функция рассчета стационарного распределения вероятностей состояний систем p_0 и p_n

```python
C = 50
g = 35
lambda1 = 30
lambda2 = 20
mu = 0.8
rho1 = lambda1/mu
rho2 = lambda2/mu
```
✓ 0.0s

Рис. 2.3: Начальные значения

```python
p_0 = probability_0(rho1=rho1, rho2=rho2, g=g, C=C)
print("Распределения вероятностей того, что система находится в состоянии 0-го: {}".format(p_0))
# this is valide for the above system configuration only.
```
✓ 0.0s

```
Распределения вероятностей того, что система находится в состоянии 0-го: 1.2137960510504379e-24
```

Рис. 2.4: Стационарное распределение вероятностей системы в исходном состоянии p_0

```python
# print values of probability of systems and compare total to 1
probability_n_array = []
for n in np.arange(C+1):
    probability_n_array.append(probability_n(n=n, rho1=rho1, rho2=rho2, g=g, C=C))
print("Распределения вероятностей для каждого n:", probability_n_array)
print("probability_n_array размер: ", len(probability_n_array))
print("Сумма всех вероятностей: ", sum(probability_n_array))
```

✓ 0.0s

```
Stationary probability in each state: [
    1.2137960510504379e-24, 7.586225319065237e-23, 2.3706954122078865e-21, 4.938948775433097e-20, 7.717107461614214e-19,
    9.646384327017766e-18, 1.0048317007310175e-16, 8.971711613669797e-16, 7.009149698179529e-15, 4.867465068180229e-14,
    3.042165667612643e-13, 1.7285032202344562e-12, 9.002620938721128e-12, 4.328183143615926e-11, 1.9322246176856814e-10,
    8.050935907023673e-10, 3.1448968386811227e-09, 1.1562120730445302e-08, 4.014625253626841e-08, 1.3206004123772503e-07,
    4.126876288678907e-07, 1.2282369906782463e-06, 3.4893096326086537e-06, 9.481819653827864e-06, 2.4692238681843397e-05,
    6.173059670460849e-05, 0.00014839085746300116, 0.0003434973552384286З, 0.0007667351679429211, 0.0016524464826356059,
    0.0034425968388241785, 0.006940719433113264, 0.013556092642799344, 0.025674417884089664, 0.04719562111045894,
    0.08427789484010526, 0.08778947379177629, 0.08897581803220571, 0.08780508358441354, 0.08442796498501301,
    0.07915121717344971, 0.07239440595132594, 0.06463786245654102, 0.05637022888651833, 0.04804280871010085,
    0.040035673925084045, 0.03263777765631851, 0.026040780044934986, 0.020344359410105456, 0.015569662813856218,
    0.011677247110392162,
]
probability_n_array размер:  51
Сумма всех вероятностей:  1.0
```

Рис. 2.5: Стационарное распределение вероятностей для каждого n от 0 до C и сумма вероятностей

```python
# blocking probability of service of type-1 if when just the probability_n when n = C
def p_C(C, rho1, rho2, g):
    return probability_n(n=C, rho1=rho1, rho2=rho2, g=g, C=C)
E1 = p_C(C=C, rho1=rho1, rho2=rho2, g=g)
print("Вероятность блокировки по времени E1 = p_C = {}".format(E1))
```

✓ 0.0s

```
Вероятность блокировки по времени E1 = p_C = 0.011677247110392162
```

Рис. 2.6: Вероятность блокировки по времени E_1

```python
# blocking proability of service of type-2 if when system is at least in state g and onward to C
def sum_pngc(rho1, rho2, g, C):
    return sum([probability_n(n=i, rho1=rho1, rho2=rho2, g=g, C=C) for i in np.arange(g, C+1)])
E2 = sum_pngc(rho1=rho1, rho2=rho2, g=g, C=C)
print('Вероятность блокировки по времени E_2 = {}'.format(E2))
```

✓ 0.0s

```
Вероятность блокировки по времени E_2 = 0.900178259372141
```

Рис. 2.7: Вероятность блокировки по времени E_2

```python
# calculate average number of service being served in system which is just the expecation of system station probability
def average_N_floored(rho1, rho2, g, C):
    return math.floor(sum([n * probability_n(n=n, rho1=rho1, rho2=rho2, g=g, C=C) for n in np.arange(C+1)]))

def average_N(rho1, rho2, g, C):
    return sum([n * probability_n(n=n, rho1=rho1, rho2=rho2, g=g, C=C) for n in np.arange(C+1)])

# compute the the number of N for the current system configuration

N = average_N(rho1=rho1, rho2=rho2, g=g, C=C)
print("Среднее число N обслуживаемых запросов: {}".format(N))
```

✓ 0.0s

```
Среднее число N обслуживаемых запросов: 39.55764674905677
```

Рис. 2.8: Среднее число обслуживаемых запросов $\bar{N}$

```python
E1_array = []
E2_array = []
lambda1_array = np.arange(lambda1+1)
for lambda1 in lambda1_array:
    rho1 = lambda1/mu
    # rho2 = lambda2/mu # unchange
    # compute E1 and add to array E1_array
    E1_array.append(p_C(C=C, rho1=rho1, rho2=rho2, g=g)) # new rho1 computed in loop
    # compute E2 and add to array E2_array
    E2_array.append(sum_pngc(rho1=rho1, rho2=rho2, g=g, C=C))

import matplotlib.pyplot as plt

plt.plot(lambda1_array, E1_array, color= 'orangered', label="$E_1$")
plt.plot(lambda1_array, E2_array, '',color= 'blue', label="$E_2$")
plt.legend()
plt.title('Вероятность блокировки по времени в зависимости от интенсивности запросов')
plt.xlabel('Интенсивности поступления запросов')
plt.ylabel('Значение вероятности')
plt.show()
```

✓ 3.0s

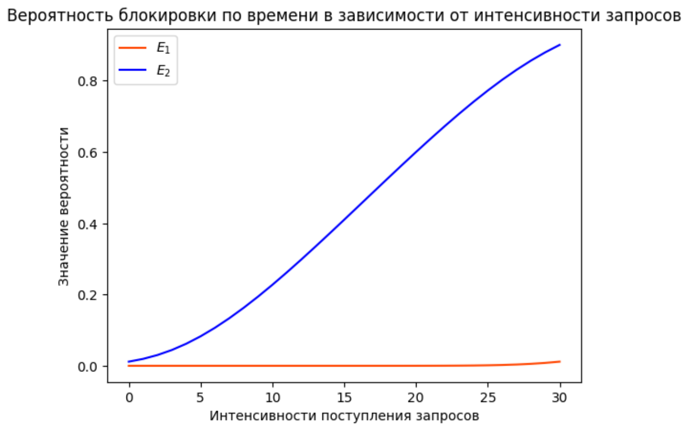


Рис. 2.9: Построение графика зависимости вероятности блокировки по времени от интенсивности поступления запросов E_1, E_2

```python
N_array_floored = []
N_array = []
lambda1_array = np.arange(lambda1+1)
for lambda1 in lambda1_array:
    rho1 = lambda1/mu
    # rho2 = lambda2/mu # unchange
    # compute N and add to array N_array
    N_array_floored.append(average_N_floored(rho1=rho1, rho2=rho2, g=g, C=C)) # new rho1 computed in loop
    N_array.append(average_N(rho1=rho1, rho2=rho2, g=g, C=C))

import matplotlib.pyplot as plt

#plt.plot(lambda1_array, N_array_floored, color= 'magenta', label="$ [N] $")
plt.plot(lambda1_array, N_array, color= 'blue', label="$N$")
plt.title('Среднее число обслуживаемых в зависимости от интенсивности запросов')
plt.legend()
plt.xlabel('Интенсивности поступления запросов')
plt.ylabel('Среднее число запросов')
plt.show()
```

✓ 0.3s

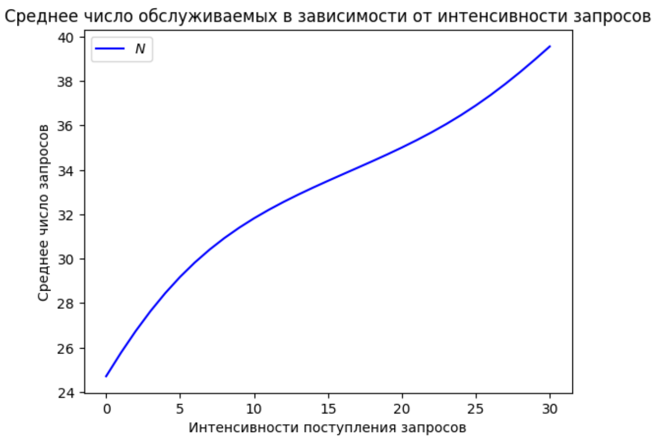


Рис. 2.10: Среднее число обслуживаемых запросов $\bar{N}$