Министерство образования и науки Российской Федерации Федеральное
Государственное бюджетное
образовательное учреждение высшего профессионального образования
«Московский государственный технический университет
имени Н. Э. Баумана»

Домашняя работа
по курсу:
«Аналитические модели и имитационное моделирование
на системном уровне»

Вариант 7

Студент: Никатов В. А.
Группа: РК6-84Б
Преподаватель: Берчун Ю.В.

2020

Содержание

1 Проектирование Call-центра

1.1 Исходные данные

Известно, что среднее время между звонками клиентов составляет T_c секунд, а среднее время обслуживания T_s секунд. Все потоки случайных событий считать пуассоновскими. Если все операторы заняты, звонок теряется.

Таблица 1.1.1. Исходные данные.

Группа	Вариант	T_c	T_s	T_w
РК6-84Б	7	39	229	517

1.2 Система без очереди

1.2.1 Задание

Рассмотреть систему без очереди. Построить графики от числа операторов:

- вероятности отказа (вплоть до обеспечения отказов менее 1%);
- математического ожидания числа занятых операторов;
- коэффициента загрузки операторов.

1.2.2 Решение

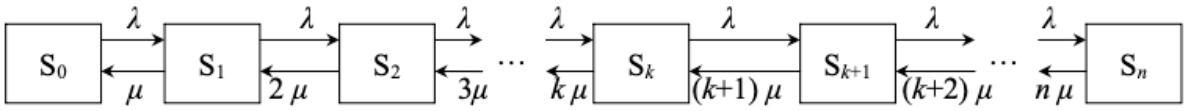

Рисунок 1.2.2.1. Граф состояний системы.

Интенсивность поступления заявок

$$\lambda = \frac{1}{T_c}.$$

Интенсивность обслуживания заявок

$$\mu = \frac{1}{T_s}.$$

Приведенная интенсивность потока

$$\rho = \frac{\lambda}{\mu}.$$

Вероятность того, что все операторы свободны

$$P_0 = \left(1 + \sum_{k=1}^{n} \frac{\rho^k}{k!}\right)^{-1}.$$

Вероятность того, что k каналов обслуживания заняты

$$P_k = \frac{\rho^k}{k!} \cdot P_0;\ k \in [1; n].$$

Вероятность отказа

$$P_{\text{отк}} = P_n = \frac{\rho^n}{n!} \cdot P_0.$$

Вероятность обслуживания заявки равна относительной пропускной способности

$$P_{\text{обс}} = Q = 1 - P_{\text{отк}}.$$

Абсолютная пропускная способность

$$\text{A} = \lambda \cdot Q.$$

Математическое ожидание среднего числа занятых операторов

$$L_{\text{обс}} = \frac{\text{A}}{\mu} = \rho \cdot Q.$$

Коэффициент загрузки операторов

$$q = \frac{L_{\text{обс}}}{n}.$$

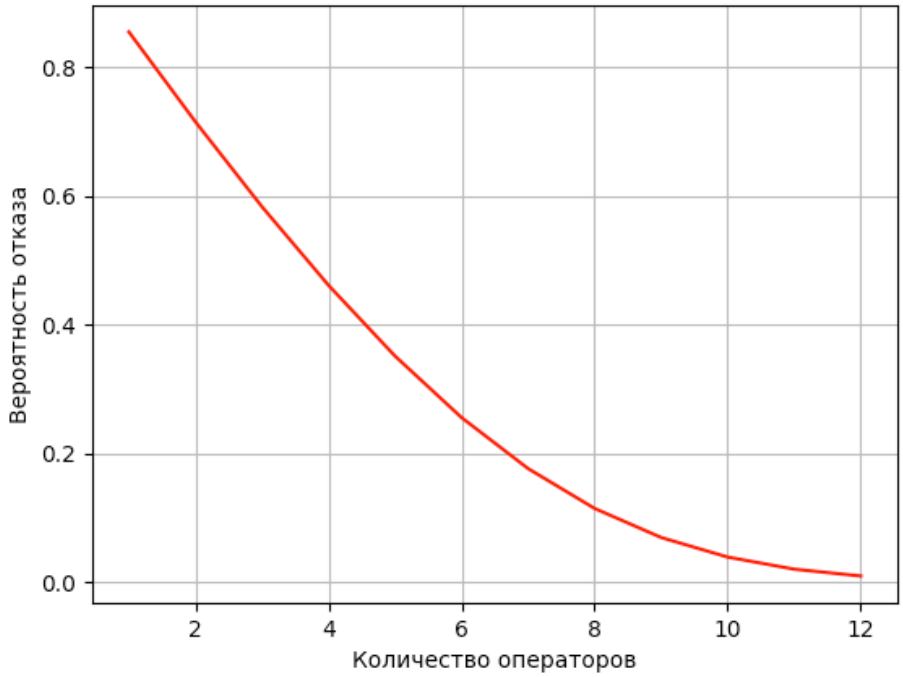

Рисунок 1.2.2.2. График зависимости вероятности отказов от количества операторов.

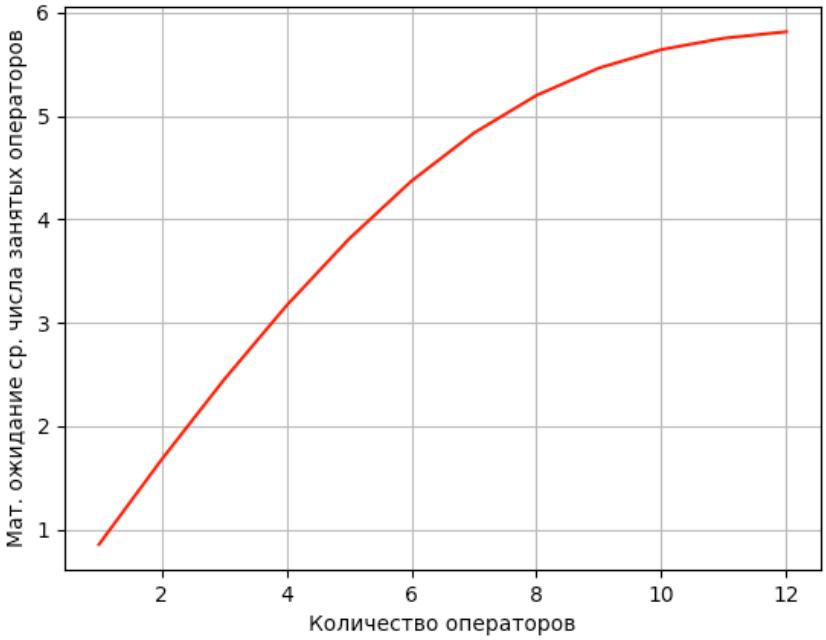

Рисунок 1.2.2.3. График зависимости математического ожидания среднего числа занятых операторов от количества операторов.

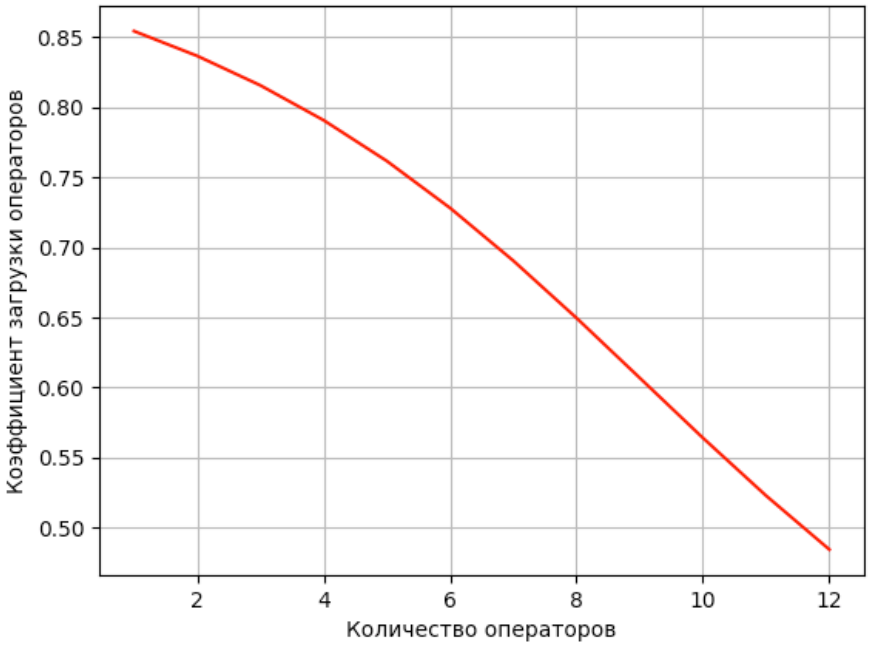

Рисунок 1.2.2.4. График зависимости коэффициента загрузки операторов от количества операторов.

Код программной реализации решения задачи представлен в приложении 1.

1.3 Система с ограниченной очередью

1.3.1 Задание

Рассмотреть систему с ограниченной очередью.

Варьируя число операторов (вплоть до числа каналов, соответствующего 1% отказов в системе без очереди), построить семейства графиков от числа мест в очереди:

- вероятности отказа;
- математического ожидания числа занятых операторов;
- коэффициента загрузки операторов;
- вероятности существования очереди;
- математического ожидания длины очереди;
- коэффициента занятости мест в очереди.

Варьируя число мест в очереди, построить семейства графиков от числа операторов:

- вероятности отказа;
- математического ожидания числа занятых операторов;
- коэффициента загрузки операторов;
- вероятности существования очереди;
- математического ожидания длины очереди;
- коэффициента занятости мест в очереди.

1.3.2 Решение

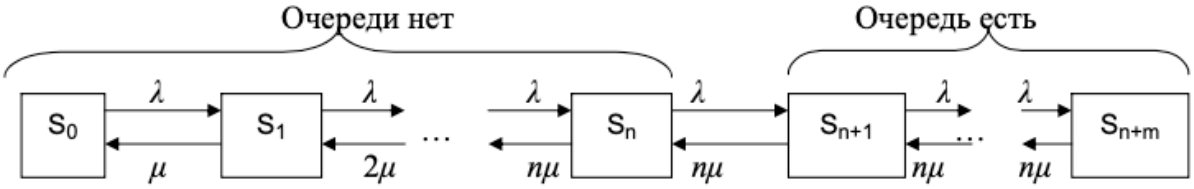

Рисунок 1.3.2.1. Граф состояний системы.

Вероятность того, что все операторы свободны

$$P_0 = \left(1 + \sum_{k=1}^{n} \frac{\rho^k}{k!} + \sum_{i=1}^{m} \frac{\rho^{n+i}}{n^i \cdot n!}\right)^{-1}.$$

Вероятность того, что k каналов обслуживания заняты

$$P_k = \frac{\rho^k}{k!} \cdot P_0;\ k \in [1; n].$$

Вероятность того, что длина очереди составляет i

$$P_{n+i} = \frac{\rho^{n+i}}{n^i \cdot n!} \cdot P_0, i \in [1; m].$$

Вероятность отказа

$$P_{\text{отк}} = P_{n+m} = \frac{\rho^{n+m}}{n! \cdot n^m} \cdot P_0.$$

Вероятность обслуживания заявки равна относительной пропускной способности

$$P_{\text{обс}} = Q = 1 - P_{\text{отк}}.$$

Математическое ожидание числа занятых операторов

$$L_{\text{обс}} = \sum_{k=0}^{n} k \cdot P_k = \rho \cdot Q.$$

Коэффициент загрузки операторов

$$q = \frac{L_{\text{обс}}}{n}.$$

Вероятность образования очереди

$$P_{\text{оч}} = \sum_{i=0}^{m-1} P_{n+i} = \frac{\rho^n}{n!} \cdot \frac{1 - \left(\frac{\rho}{n}\right)^m}{1 - \frac{\rho}{n}} \cdot P_0.$$

Математическое ожидание длины очереди

$$L_{\text{оч}} = \sum_{i=1}^{m} i \cdot P_{n+i} = \frac{\rho^{n+1}}{n \cdot n!} \cdot \frac{1 - \left(\frac{\rho}{n}\right)^m \cdot (1 + m(1 - \frac{\rho}{n})}{\left(1 - \frac{\rho}{n}\right)^2} \cdot P_0$$

Коэффициент занятости мест в очереди

$$h = \frac{L_{\text{оч}}}{m}.$$

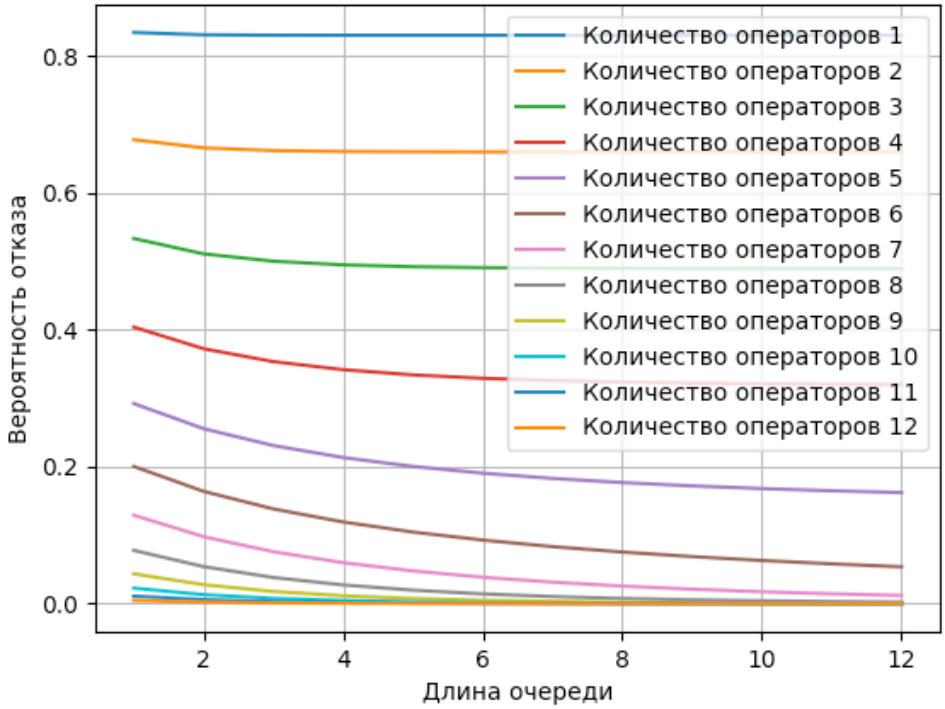

Рисунок 1.3.2.2. График зависимости вероятности отказа от длины очереди.

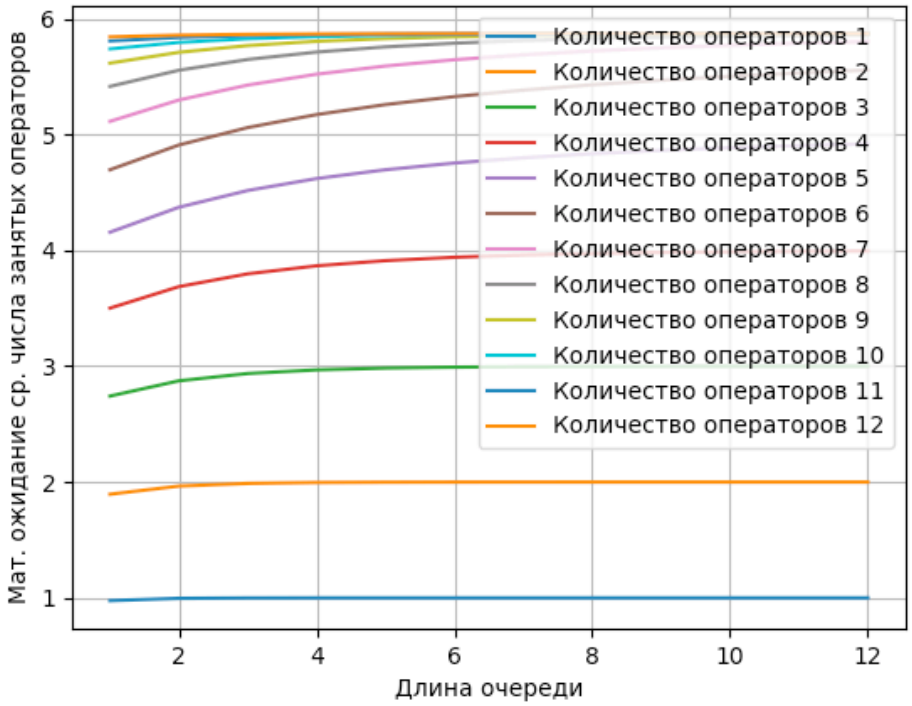

Рисунок 1.3.2.3. График зависимости мат. ожидания числа занятых операторов от длины очереди.

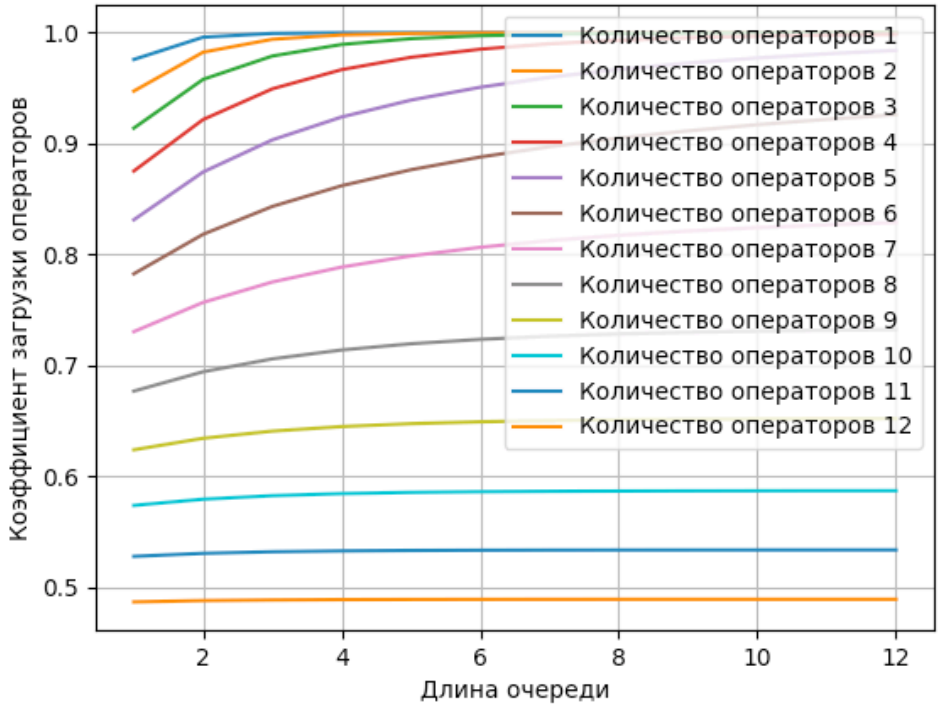

Рисунок 1.3.2.4. График зависимости коэффициента загрузки операторов от длины очереди.

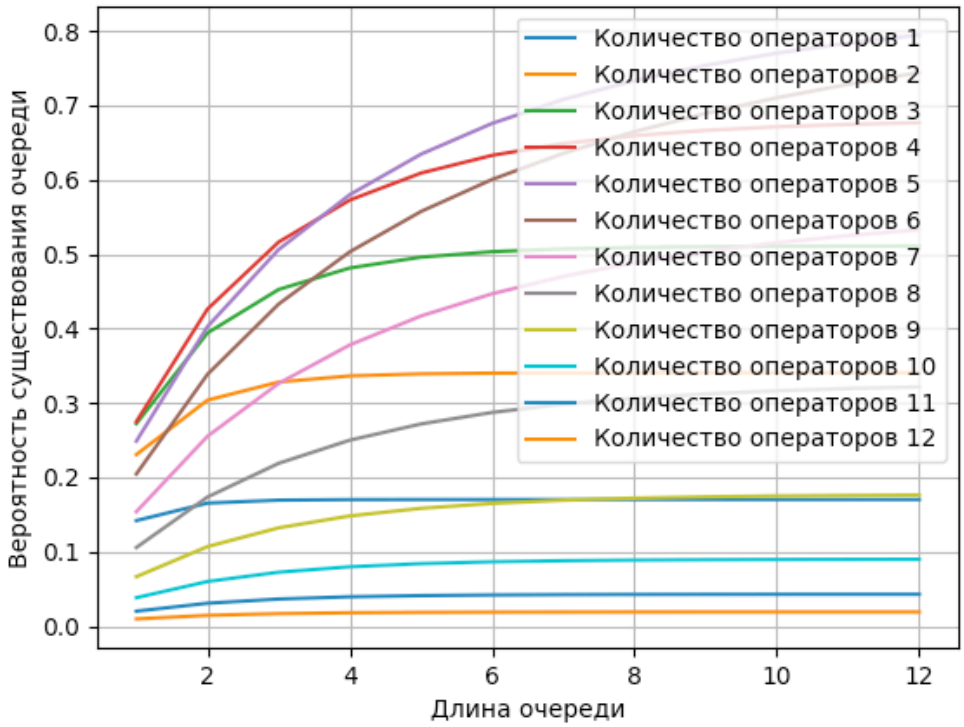

Рисунок 1.3.2.5. График зависимости вероятности существования очереди от длины очереди.

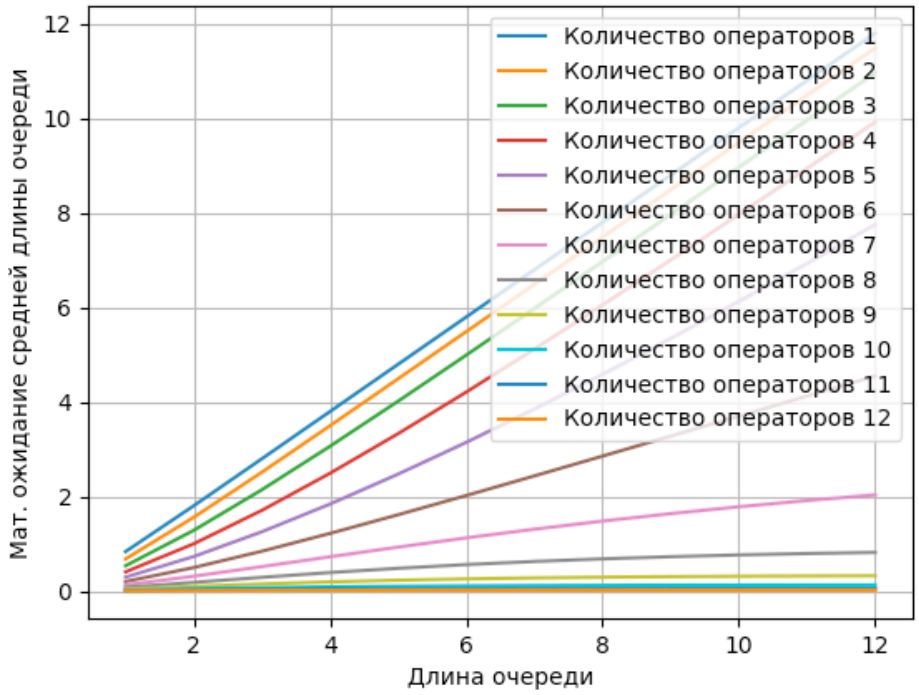

Рисунок 1.3.2.6. График зависимости мат. ожидания длины очереди от длины очереди.

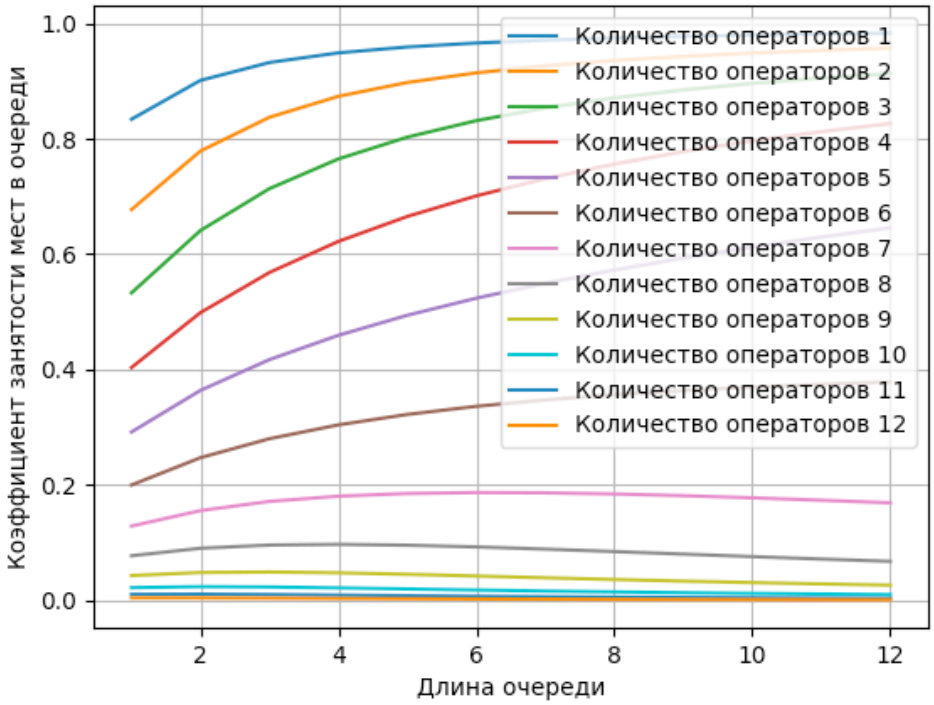

Рисунок 1.3.2.7. График коэффициента занятости мест в очереди от длины очереди.

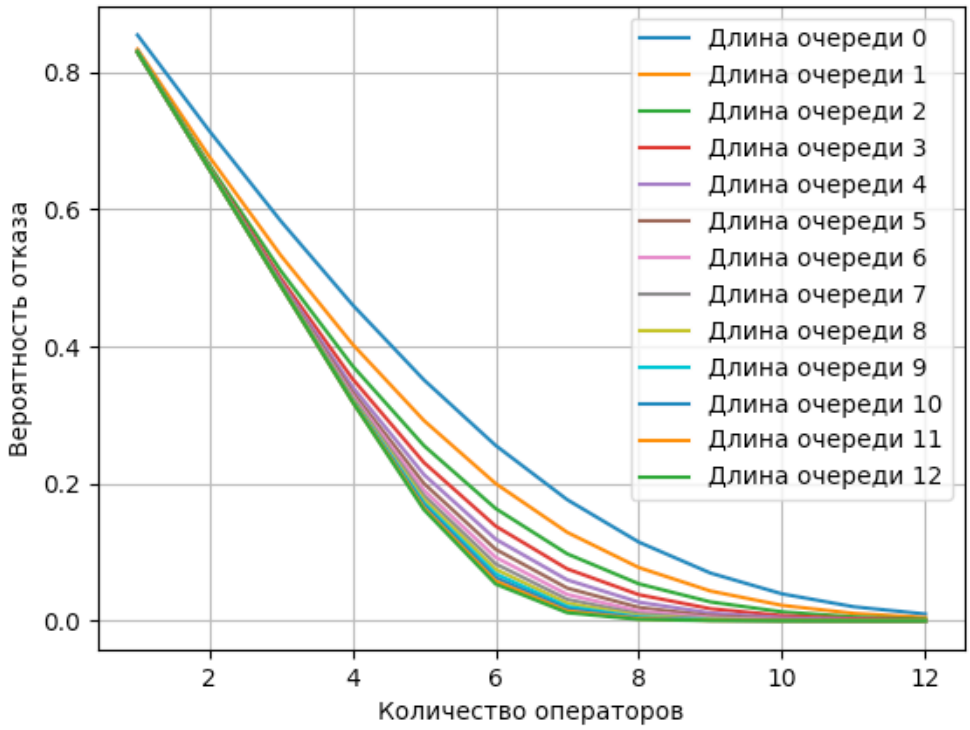

Рисунок 1.3.2.8. График зависимости вероятности отказа от количества операторов.

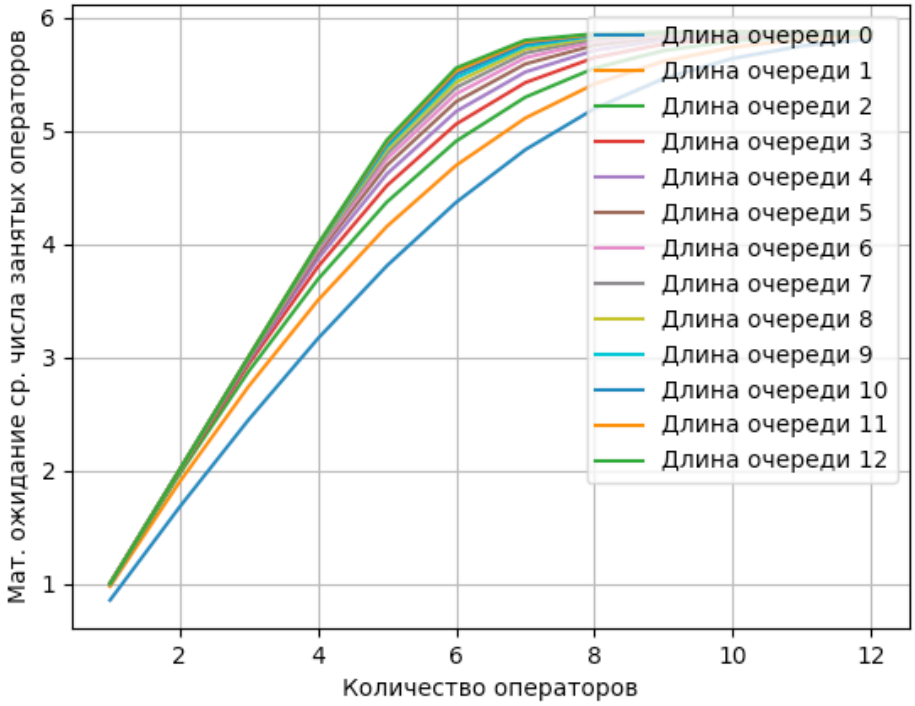

Рисунок 1.3.2.9. График зависимости мат. ожидания числа занятых операторов от количества операторов.

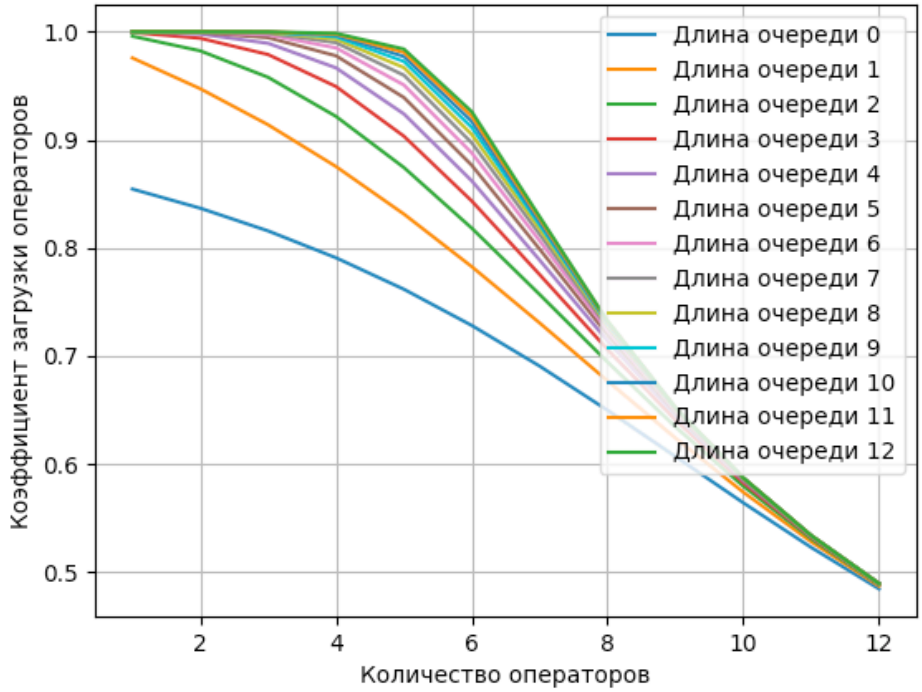

Рисунок 1.3.2.10. График зависимости коэффициента загрузки операторов от количества операторов.

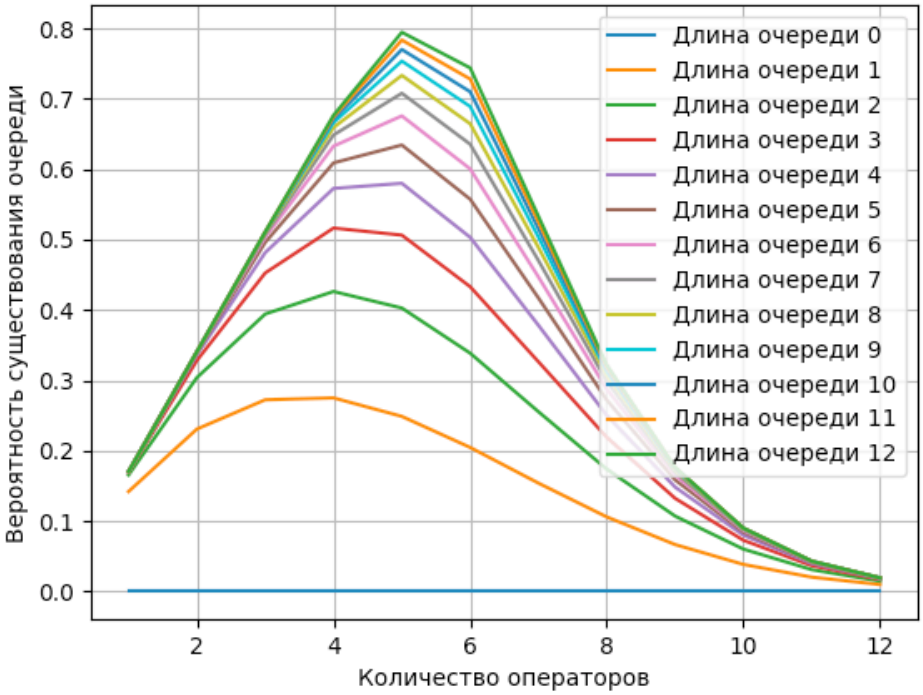

Рисунок 1.3.2.11. График зависимости вероятности существования очереди от количества операторов.

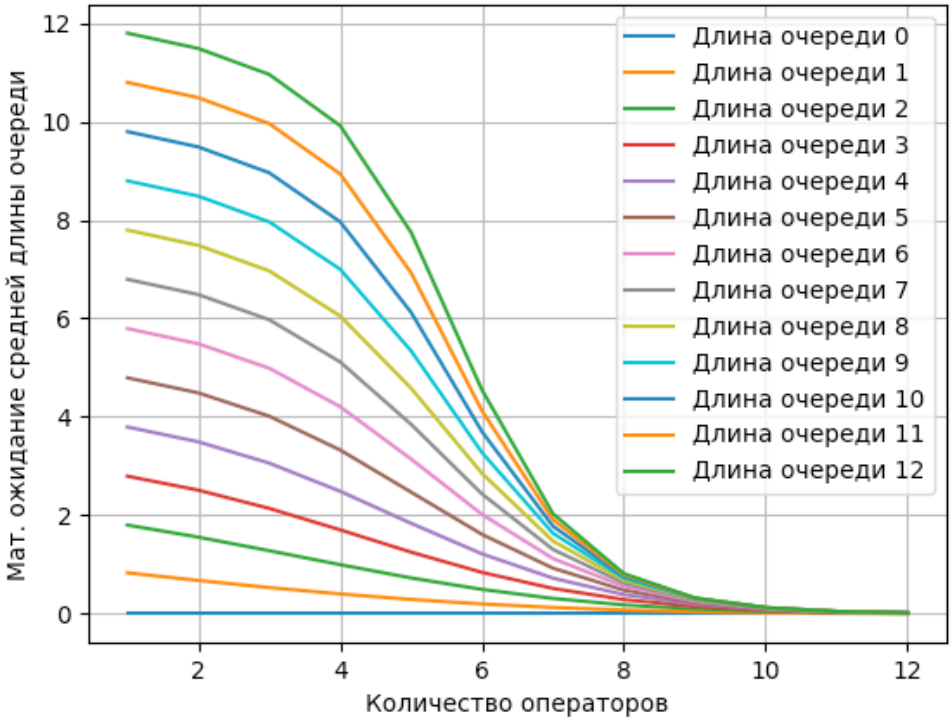

Рисунок 1.3.2.12. График зависимости мат. ожидания длины очереди от количества операторов.

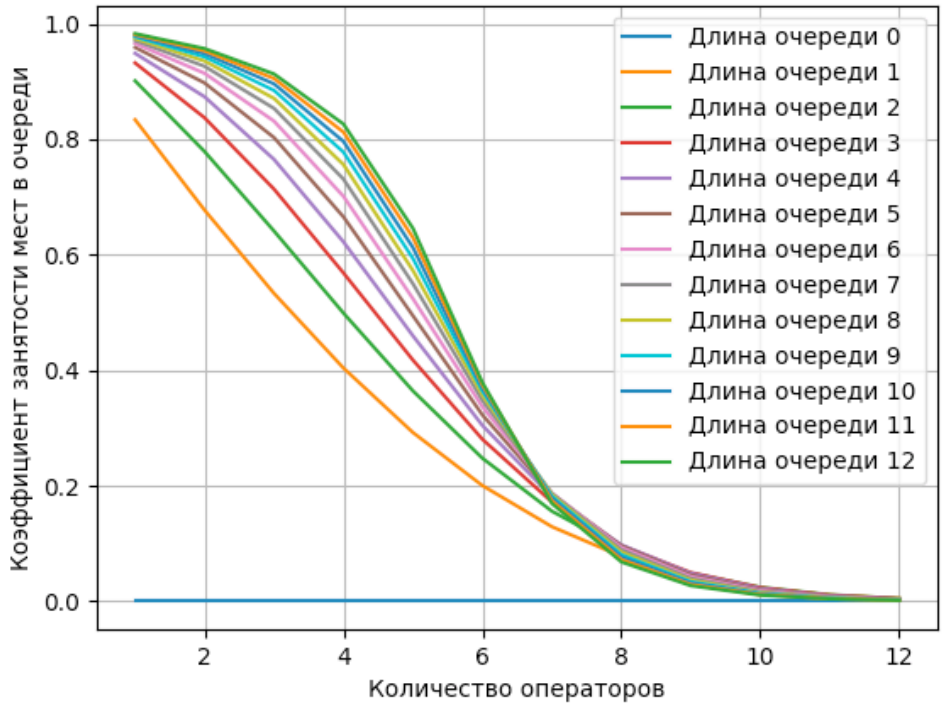

Рисунок 1.3.2.13. График коэффициента занятости мест в очереди от количества операторов.

Код программной реализации решения задачи представлен в приложении 2.

1.4 Система без ограничений на длину очереди

1.4.1 Задание

Рассмотреть систему без ограничений на длину очереди. Построить графики от числа операторов (вплоть до числа каналов, соответствующего 1% отказов в системе без очереди):

- математического ожидания числа занятых операторов;
- коэффициента загрузки операторов;
- вероятности существования очереди;
- математического ожидания длины очереди.

1.4.2 Решение

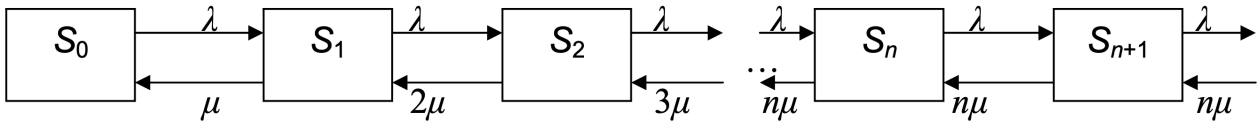

Рисунок 1.4.2.1. Граф состояний системы.

Вероятность того, что все операторы свободны

$$P_0 = \left(1 + \sum_{k=1}^{n-1} \frac{\rho^k}{k!} + \frac{\rho^n}{n! \cdot (1 - \frac{\rho}{n})}\right)^{-1}$$

Вероятность того, что k каналов обслуживания заняты

$$P_k = \frac{\rho^k}{k!} \cdot P_0;\ k \in [1; n].$$

Поскольку очередь не ограничена, вероятность отказа

$$P_{\text{отк}} = 0$$

Вероятность обслуживания заявки равна относительной пропускной способности

$$P_{\text{обс}} = Q = 1 - P_{\text{отк}} = 1.$$

Математическое ожидание среднего числа занятых операторов

$$L_{\text{обс}} = \sum_{k=0}^{n} k \cdot P_k = \rho \cdot Q. = \rho$$

Коэффициент загрузки операторов

$$q = \frac{L_{\text{обс}}}{n}.$$

Вероятность образования очереди

$$P_{\text{оч}} = \frac{\rho^n}{n!} \cdot \frac{n}{n-\rho} \cdot P_0.$$

Математическое ожидание длины очереди

$$L_{\text{оч}} = \frac{\rho^{n+1}}{n!} \cdot \frac{n}{(n-\rho)^2} \cdot P_0$$

При детальном рассмотрении результатов расчета следует иметь ввиду, что при $\rho/n \geq 1$ вероятности $P_0 = P_1 = \cdots = P_n = 0$, т.е. очередь неограниченно возрастает. При количестве операторов $n < 6$ очередь будет неограниченно возрастать, а значит на при $1 \leq n \leq 5$ каждый оператор будет всегда занят, очередь в любом случае будет (при прошествии небольшого количества времени), а математическое ожидание длины очереди не определено, в связи с тем, что очередь неограниченно возрастает. На графике на рисунке 1.4.2.4 мат. ожиданию в данном диапазоне присвоено отрицательное значение.

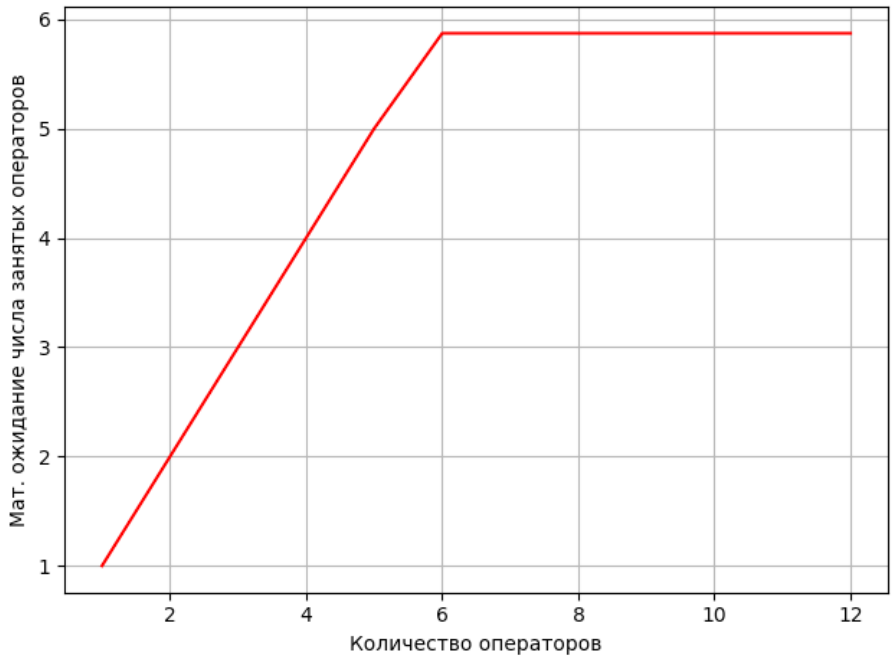

Рисунок 1.4.2.2. График зависимости мат. ожидания числа занятых операторов от количества операторов.

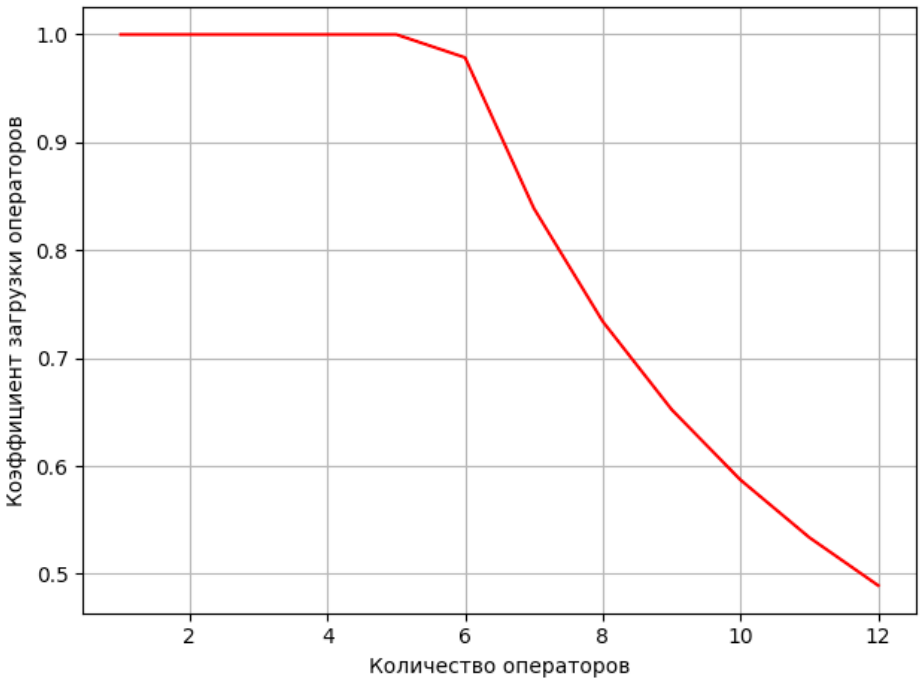

Рисунок 1.4.2.3. График зависимости коэффициента загрузки операторов от количества операторов.

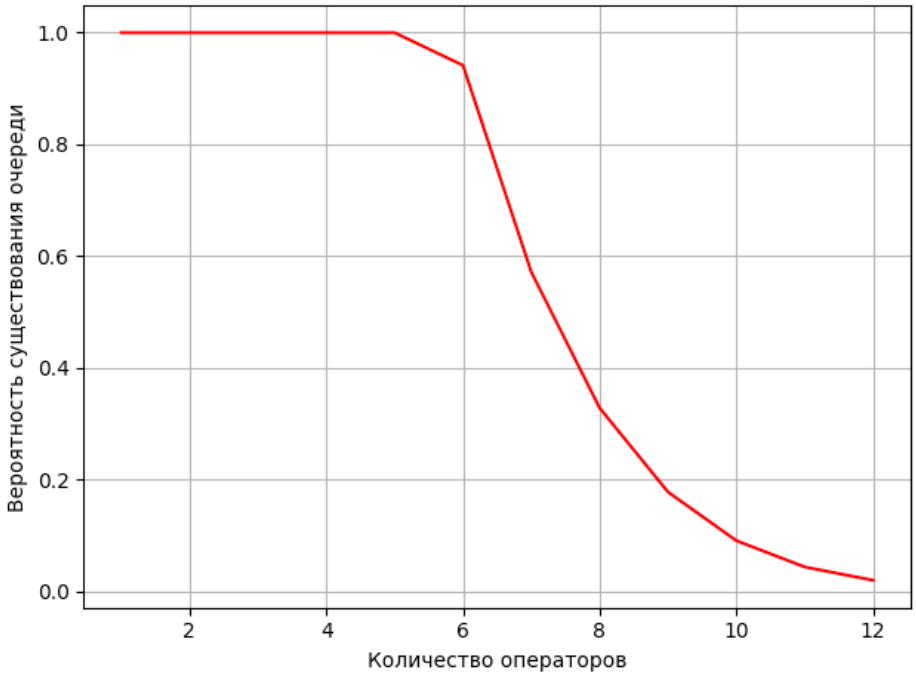

Рисунок 1.4.2.4. График зависимости вероятности существования очереди от количества операторов.

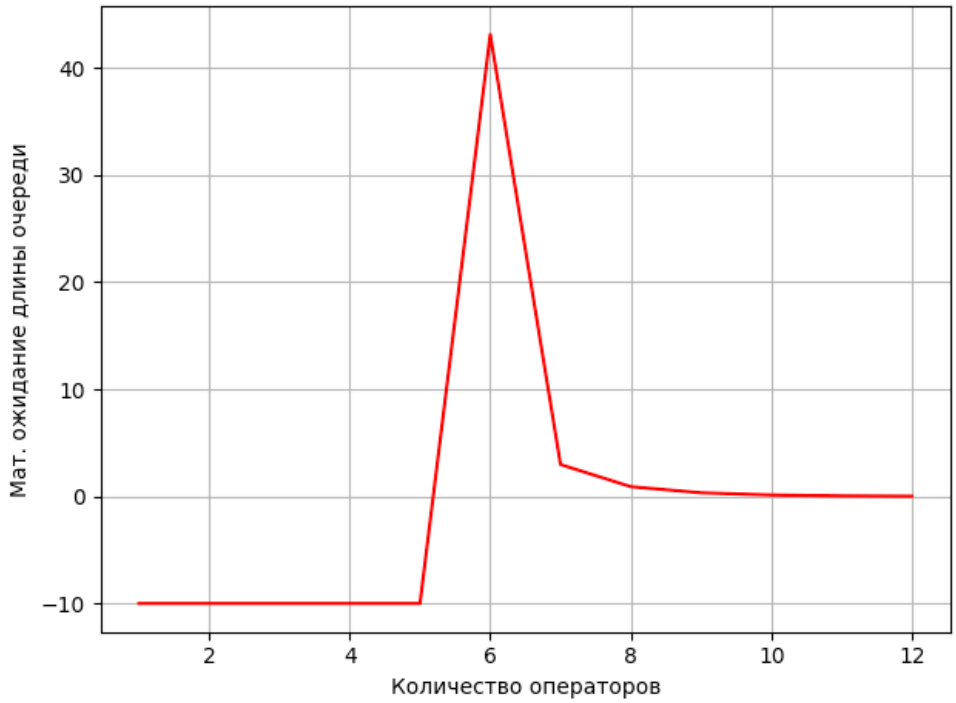

Рисунок 1.4.2.5. График зависимости мат. ожидания длины очереди от количества операторов.

Как было сказано ранее, при количестве операторов $1 \leq n \leq 5$ каждый оператор будет все время занят, а очередь будет неумолимо расти. Чтобы не пострадал масштаб графика на рисунке 1.4.2.5 и можно было без труда рассмотреть вторую половину графика, которая нас и интересует, значение «inf» заменено на отрицательное.

Код программной реализации решения задачи представлен в приложении 3.

1.5 Система с учетом фактора ухода из очереди

1.5.1 Задание

Рассмотреть систему без ограничений на длину очереди, учитывающей фактор ухода клиентов из очереди (среднее приемлемое время ожидания - T_w секунд). Построить графики от числа операторов (вплоть до числа каналов, соответствующего 1% отказов в системе без очереди):

- математического ожидания числа занятых операторов;
- коэффициента загрузки операторов;
- вероятности существования очереди;
- математического ожидания длины очереди.

1.5.2 Решение

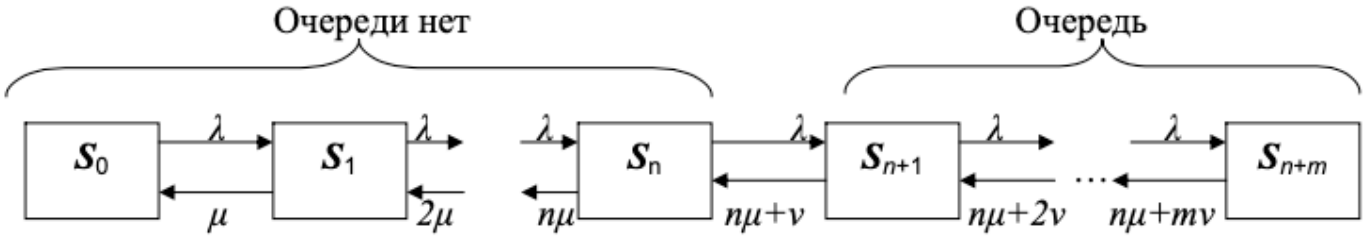

Рисунок 1.5.2.1. Граф состояний системы.

Такая СМО отличается от СМО описанной в предыдущем параграфе наличием интенсивности потока ухода заявок из очереди

$$v = \frac{1}{T_w}.$$

Отношение интенсивности ухода к интенсивности обслуживания обозначается

$$\beta = \frac{v}{\mu}.$$

Вероятность того, что все операторы свободны

$$P_0 = \left(1 + \sum_{k=1}^{n} \frac{\rho^k}{k!} + \frac{\rho^n}{n!} \cdot \sum_{i=1}^{m} \frac{\rho^i}{\prod_{l=1}^{i}(n + l\beta)}\right)^{-1}.$$

Вероятность того, что k каналов обслуживания заняты

$$P_k = \frac{\rho^k}{k!} \cdot P_0; k \in [1; n].$$

Вероятность того, что длина очереди составляет i

$$P_{n+i} = P_n \cdot \frac{\rho^i}{\prod_{l=1}^{i}(n + l\beta)}, i \in [1; m].$$

Необходимо подобрать такое m, при котором после добавлении нового состояния в модель значение P_0 не будет значительно изменяться

$$|P_0^m - P_0^{m+1}| < \varepsilon,$$

$$\varepsilon = 10^{-6}.$$

Математическое ожидание числа занятых операторов

$$L_{\text{обс}} = \sum_{k=1}^{n} k \cdot P_k + \sum_{i=1}^{m} n \cdot P_{n+i} = \sum_{k=1}^{n} k \cdot \frac{\rho^k}{k!} \cdot P_0 + n \cdot \frac{\rho^n}{n!} \cdot P_0 \cdot \sum_{i=1}^{m} \frac{\rho^i}{\prod_{l=1}^{i}(n + l\beta)}.$$

Коэффициент загрузки операторов

$$q = \frac{L_{\text{обс}}}{n}.$$

Вероятность образования очереди

$$P_{\text{оч}} = \frac{\rho^n}{n!} \cdot P_0 \cdot \left(1 + \sum_{i=1}^{m-1} \frac{\rho^i}{\prod_{l=1}^{i}(n + l\beta)} \right).$$

Математическое ожидание длины очереди

$$L_{\text{оч}} = \sum_{i=1}^{m} i \cdot P_{n+i} = \frac{\rho^n}{n!} \cdot P_0 \cdot \sum_{i=1}^{m} \frac{i \cdot \rho^i}{\prod_{l=1}^{i}(n + l\beta)}.$$

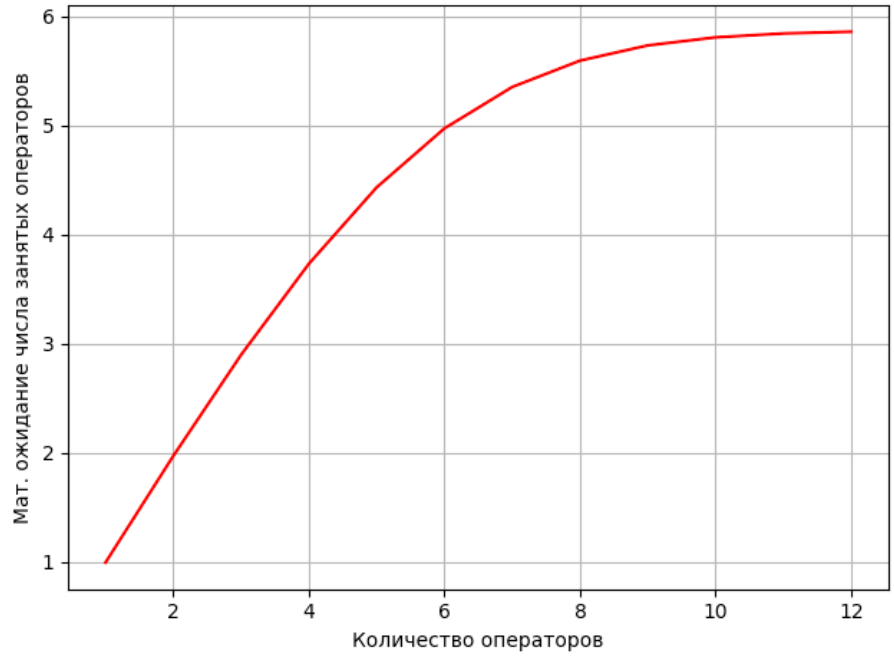

Рисунок 1.5.2.2. График зависимости мат. ожидания числа занятых операторов от количества операторов.

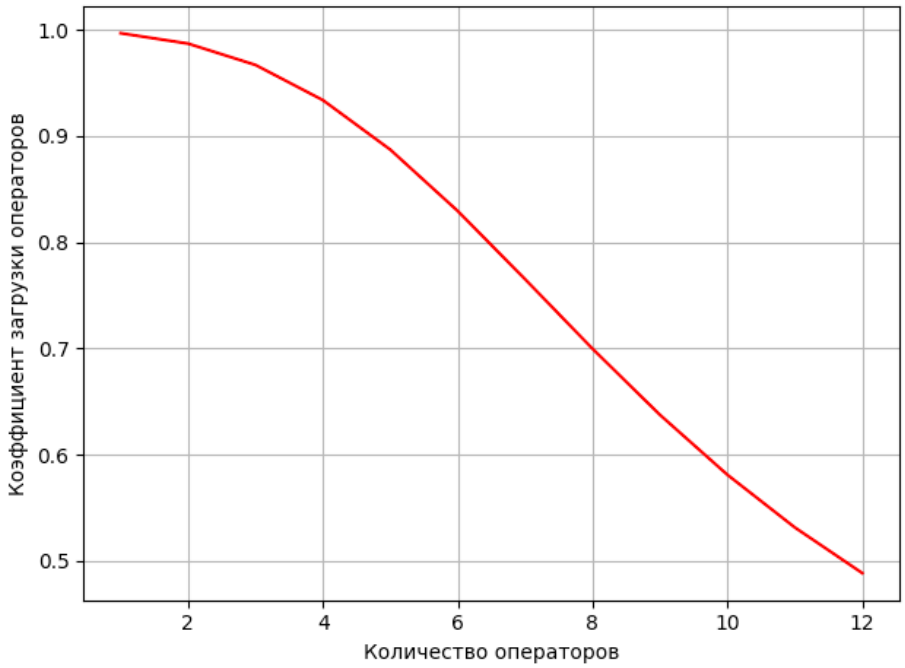

Рисунок 1.5.2.3. График зависимости коэффициента загрузки операторов от количества операторов.

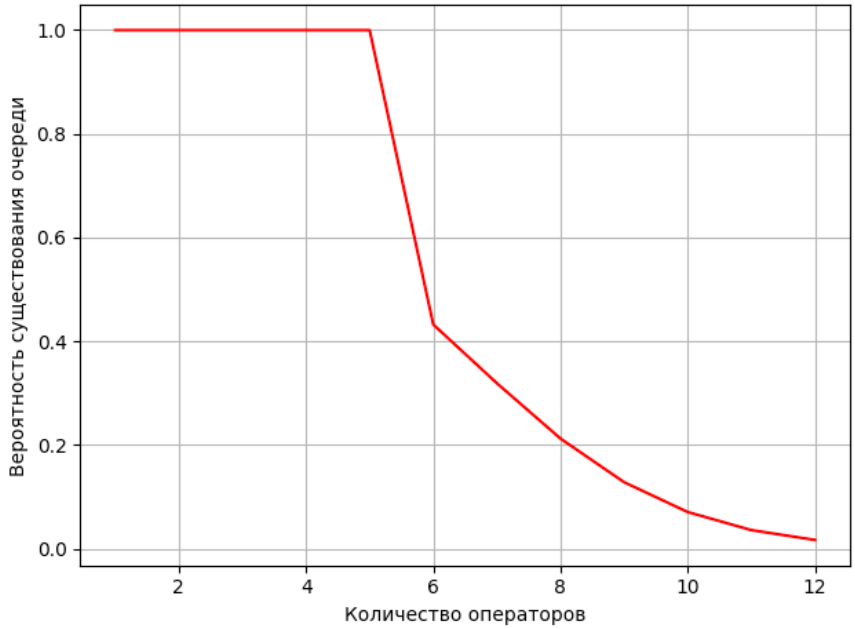

Рисунок 1.5.2.4. График зависимости вероятности существования очереди от количества операторов.

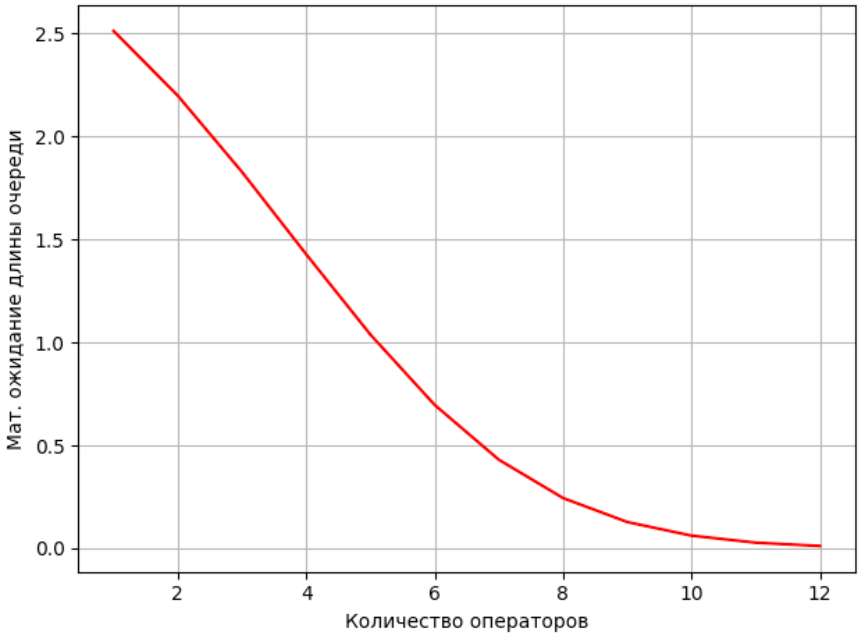

Рисунок 1.5.2.5. График зависимости мат. ожидания длины очереди от количества операторов.

Код программной реализации решения задачи представлен в приложении 4.

2 Проектирование производственного участка

2.1 Исходные данные

Имеется участок с N станками. Среднее время между наладками составляет T_c минут, среднее время наладки - T_s минут. Все потоки случайных событий считать пуассоновскими.

Таблица 2.1.1. Исходные данные.

Группа	Вариант	T_c	T_s	N
РК6-84Б	7	444	12	38

2.2 Задание

Построить графики от числа наладчиков:

- математического ожидания числа простаивающих станков;
- математического ожидания числа станков, ожидающих обслуживания;
- вероятности ожидания обслуживания;
- математического ожидания числа занятых наладчиков;
- коэффициента занятости наладчиков.

2.3 Решение

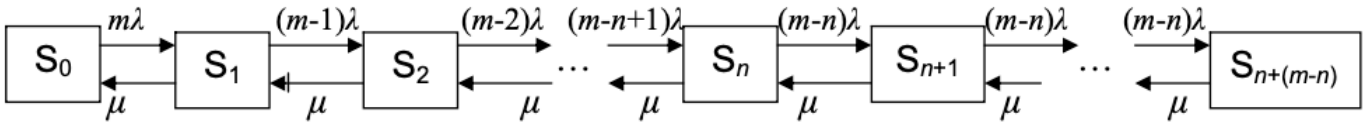

Рисунок 2.3.1. Граф состояний системы.

Интенсивность потока требований каждого источника заявок

$$\lambda = \frac{1}{T_c}.$$

Интенсивность обслуживания источников заявок

$$\mu = \frac{1}{T_s}.$$

Если под обслуживанием находятся k объектов, то интенсивность потока заявок в СМО будет равна $(N-k)\lambda$, а максимальная длина очереди равна $N-n$. Граф такой СМО с обозначенным в качестве количества станков $m = N$ представлен на рис. 2.3.1.

Вероятность того, что все операторы свободны

$$P_0 = \left(\sum_{k=0}^{n} \frac{N!}{(N-k)!} \cdot \rho^k + \frac{N!}{(N-n)!} \cdot \rho^{n+1} \cdot \frac{1 - \left((N-n)\cdot\rho\right)^{N-n}}{1-(N-n)\cdot\rho} \right)^{-1}.$$

Вероятность того, что все операторы заняты

$$P_n = \frac{N!}{(N-n)!} \cdot \rho^n \cdot P_0.$$

Вероятность ожидания обслуживания

$$P_{\text{оч}} = P_n \cdot \frac{1 - \left((N-n)\cdot\rho\right)^{N-n}}{1-(N-n)\cdot\rho} \cdot (N-n)\cdot\rho.$$

Математическое ожидание числа занятых наладчиков

$$L_{\text{обс}} = \sum_{k=1}^{n} k \cdot P_k = P_0 \cdot \sum_{k=1}^{n} \frac{N!}{(N-k)!} \cdot \rho^k \cdot k.$$

Коэффициент загрузки наладчиков

$$q = \frac{L_{\text{обс}}}{n}.$$

Математическое ожидание числа станков, ожидающих обслуживания:

$$L_{\text{оч}} = P_n \cdot \sum_{i=1}^{N-n} i \cdot (N-n)^i \cdot \rho^i$$

Математическое ожидание числа простаивающих станков:

$$L_{\text{смо}} = L_{\text{оч}} + L_{\text{обс}}.$$

В случае, когда $\rho = \frac{1}{(N-n)}$, в слагаемом P_0 возникает неопределенность вида $0/0$.

В таком случае принимаем

$$P_0 = P_0{}' = \left(N - n + 2 + \sum_{k=0}^{n} \frac{N!}{(N-n)^k \cdot (N-k)!} \right)^{-1},$$

$$P_{\text{оч}} = P_{\text{оч}}{}' = (N-n)\cdot P_n.$$

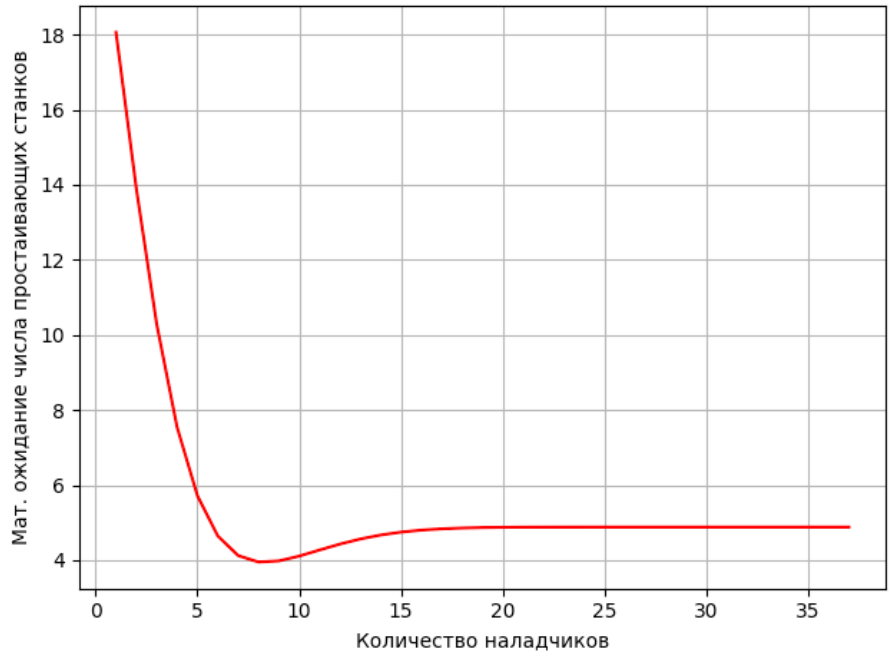

Рисунок 2.3.2. График зависимости мат. ожидания числа простаивающих станков от числа наладчиков.

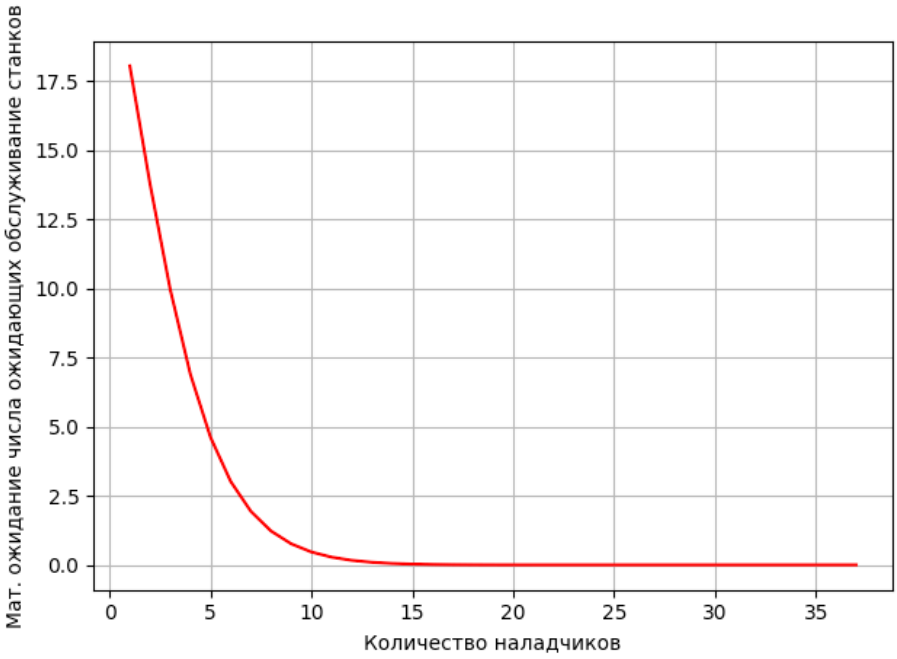

Рисунок 2.3.3. График зависимости мат. ожидания числа станков, ожидающих обслуживания от числа наладчиков.

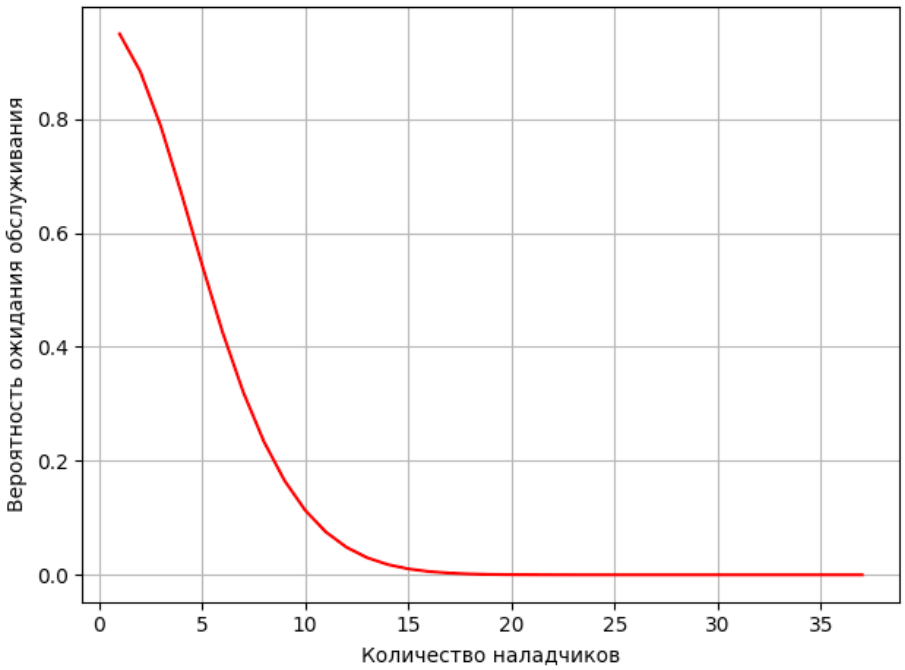

Рисунок 2.3.4. График зависимости вероятности ожидания обслуживания от числа наладчиков.

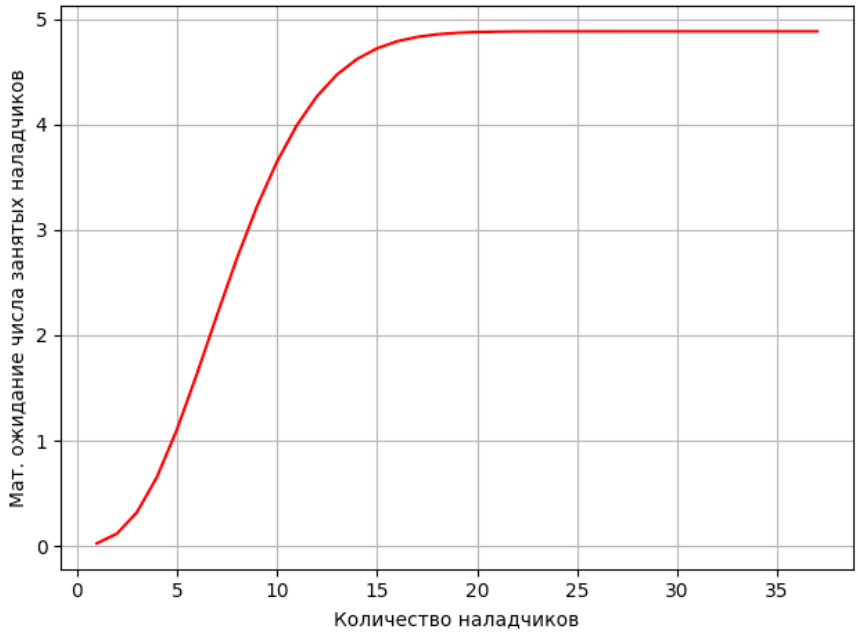

Рисунок 2.3.5. График зависимости мат. ожидания числа занятых наладчиков от числа наладчиков.

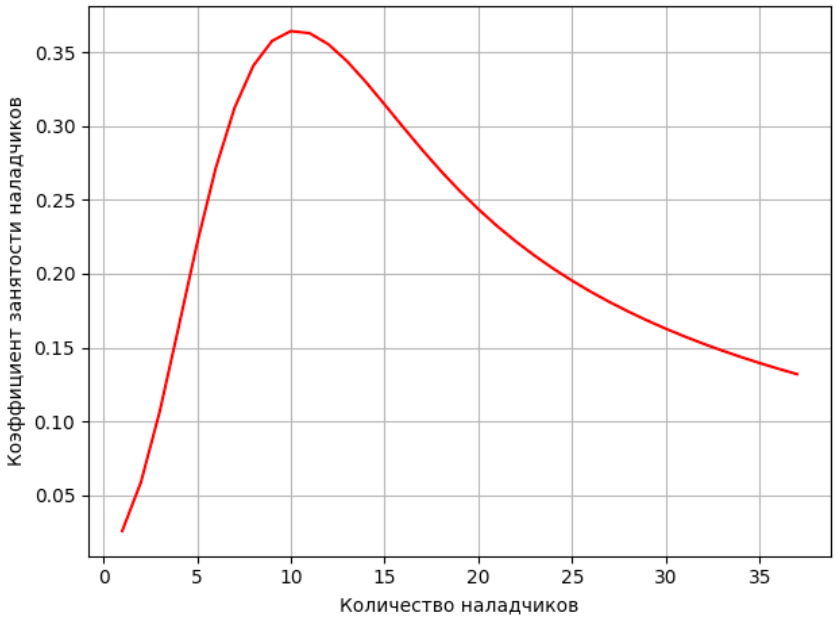

Рисунок 2.3.6. График зависимости коэффициента занятости наладчиков от числа наладчиков.

Код программной реализации решения задачи представлен в приложении 5.

Приложение 1

```
# -*- coding: utf-8 -*-

# +------------------------------+
# | Многоканальная СМО с отказами |
# +------------------------------+

from math import factorial
import matplotlib.pyplot as plt

# Приведенная интенсивность потока
def p_f(Tc, Ts):
    l = 1 / Tc
    m = 1 / Ts
    return l / m

# Вероятность того, что все каналы обслуживания свободны
def p0_f(p, n: int):
    sum = 1
    for k in range(1, n + 1):
        sum += p ** k / factorial(k)
    return 1 / sum

# Вероятность того, что k каналов обслуживания заняты
def pk_f(p0, p, k: int):
    return (p ** k) * p0 / factorial(k)

# Вероятность отказа
def pn_f(Tc, Ts, n):
    p = p_f(Tc, Ts)
    P0 = p0_f(p, n)
    return pk_f(P0, p, n)

# Математическое ожидание среднего числа занятых каналов
def ksr_f(Tc, Ts, n):
    p = p_f(Tc, Ts)
    Pn = pn_f(Tc, Ts, n)
    Q = 1 - Pn
    return p * Q

# Коэффициент загрузки операторов
def q_f(Tc, Ts, n):
    print(ksr_f(Tc, Ts, n), ' / ', n)
    return ksr_f(Tc, Ts, n) / n
```

```
def n_lim_f(Tc, Ts, Pn_delta):
    n = 1
    pn = pn_f(Tc, Ts, n)
    while Pn_delta < pn:
        n += 1
        pn = pn_f(Tc, Ts, n)
    return n

# Функция отрисовки графиков
def plot(x_list: list,
         y_list: list,
         x_tytle: str = "",
         y_tytle: str = ""):
    fig, ax1 = plt.subplots()
    plt.grid(True)
    ax1.plot(x_list, y_list, 'r-')
    ax1.set_xlabel(x_tytle)
    ax1.set_ylabel(y_tytle)

if __name__ == '__main__':
    # Константы
    Tc = 39  # время возникновения заявки
    Ts = 229  # время обслуживания заявки

    oper_tytle = "Количество операторов"
    pn_tytle = "Вероятность отказа"
    ksr_tytle = "Мат. ожидание ср. числа занятых операторов"
    q_tytle = "Коэффициент загрузки операторов"

    pn_delta = 1 / 100  # вероятность отказа, до достижения которой увеличивать предельное количество операторов
    n_lim = n_lim_f(Tc, Ts, 1 / 100)  # предельное количество операторов

    n_list = [n + 1 for n in range(n_lim)]
    Pn_list = [pn_f(Tc, Ts, n) for n in n_list]
    ksr_list = [ksr_f(Tc, Ts, n) for n in n_list]
    q_list = [q_f(Tc, Ts, n) for n in n_list]

    plot(n_list, Pn_list, oper_tytle, pn_tytle)
    plot(n_list, ksr_list, oper_tytle, ksr_tytle)
    plot(n_list, q_list, oper_tytle, q_tytle)
    plt.show()
```

Приложение 2

```
# -*- coding: utf-8 -*-

# +-----------------------------------------+
# | Многоканальная СМО с ограниченной очередью |
# +-----------------------------------------+

from math import factorial
import matplotlib.pyplot as plt


# Приведенная интенсивность потока
def p_f(Tc, Ts):
    l = 1 / Tc
    m = 1 / Ts
    return l / m


# Вероятность того, что все каналы обслуживания свободны
def p0_f(p, n, m):
    s = 1
    for k in range(1, n + 1):
        s += p ** k / factorial(k)
    for i in range(1, m + 1):
        s += (p ** (n + i)) / (factorial(n) * (n ** i))
    return 1 / s


# Вероятность состояния k [1; n + m]
def pk_f(p0, p, k, n):
    if k <= n:
        return p ** k * p0 / factorial(k)
    return p ** k * p0 / (factorial(n) * (n ** (k - n)))


# Вероятность отказа
def pn_f(Tc, Ts, n, m):
    p = p_f(Tc, Ts)
    P0 = p0_f(p, n, m)
    return pk_f(P0, p, n + m, n)


# Математическое ожидание среднего числа занятых операторов
def ksr_f(Tc, Ts, n, m):
```

```
    p = p_f(Tc, Ts)
    Pn = pn_f(Tc, Ts, n, m)
    Q = 1 - Pn
    return p * Q


# Коэффициент загрузки операторов
def q_f(Tc, Ts, n, m):
    return ksr_f(Tc, Ts, n, m) / n


# Вероятность существования очереди
def poch_f(Tc, Ts, n, m):
    s = 0
    p = p_f(Tc, Ts)
    p0 = p0_f(p, n, m)
    for i in range(m):
        s += pk_f(p0, p, n + i, n)
    return s


# Математическое ожидание длины очереди
def loch_f(Tc, Ts, n, m):
    s = 0
    p = p_f(Tc, Ts)
    p0 = p0_f(p, n, m)
    for i in range(1, m + 1):
        s += i * pk_f(p0, p, n + i, n)
    return s


# Коэффициент занятости мест в очереди
def qoch_f(Tc, Ts, n, m):
    try:
        return loch_f(Tc, Ts, n, m) / m
    except ZeroDivisionError:
        return 0


def plot(x_list: list,
         y_list: list,
         x_label: str,
         y_label: str,
         z_label: str):
```

```
    fig, ax1 = plt.subplots()
    plt.grid(True)
    for i, y in enumerate(y_list):
        ax1.plot(x_list, y, label=z_label + ' ' + str(i + 1))
    ax1.set_xlabel(x_label)
    ax1.set_ylabel(y_label)
    ax1.legend(loc='upper right')


if __name__ == '__main__':
    # Константы
    Tc = 39  # время возникновения заявки
    Ts = 229  # время обслуживания заявки
    n_lim = 12  # предельное количество операторов
    m_lim = 12  # предельная длина очереди

    oper_tytle = "Количество операторов"
    m_tytle = "Длина очереди"
    pn_tytle = "Вероятность отказа"
    ksr_tytle = "Мат. ожидание ср. числа занятых операторов"
    q_tytle = "Коэффициент загрузки операторов"
    poch_tytle = "Вероятность существования очереди"
    loch_tytle = "Мат. ожидание средней длины очереди"
    qoch_tytle = "Коэффициент занятости мест в очереди"

    n_list = [n + 1 for n in range(n_lim)]
    m_list = [m + 1 for m in range(m_lim)]

    # Варьируя число операторов, построение семейств графиков от числа мест в
очереди
    Pn_list = []
    ksr_list = []
    q_list = []
    poch_list = []
    loch_list = []
    qoch_list = []

    for n in range(1, n_lim + 1):
        Pn_list.append([pn_f(Tc, Ts, n, m) for m in m_list])
        ksr_list.append([ksr_f(Tc, Ts, n, m) for m in m_list])
        q_list.append([q_f(Tc, Ts, n, m) for m in m_list])
        poch_list.append([poch_f(Tc, Ts, n, m) for m in m_list])
        loch_list.append([loch_f(Tc, Ts, n, m) for m in m_list])
        qoch_list.append([qoch_f(Tc, Ts, n, m) for m in m_list])
```

```
plot(n_list, Pn_list, m_tytle, pn_tytle, oper_tytle)
plot(n_list, ksr_list, m_tytle, ksr_tytle, oper_tytle)
plot(n_list, q_list, m_tytle, q_tytle, oper_tytle)
plot(n_list, poch_list, m_tytle, poch_tytle, oper_tytle)
plot(n_list, loch_list, m_tytle, loch_tytle, oper_tytle)
plot(n_list, qoch_list, m_tytle, qoch_tytle, oper_tytle)

# Варьируя число мест в очереди, построение семейств графиков от числа операторов
Pn_list = []
ksr_list = []
q_list = []
poch_list = []
loch_list = []
qoch_list = []

for m in range(m_lim + 1):
    Pn_list.append([pn_f(Tc, Ts, n, m) for n in n_list])
    ksr_list.append([ksr_f(Tc, Ts, n, m) for n in n_list])
    q_list.append([q_f(Tc, Ts, n, m) for n in n_list])
    poch_list.append([poch_f(Tc, Ts, n, m) for n in n_list])
    loch_list.append([loch_f(Tc, Ts, n, m) for n in n_list])
    qoch_list.append([qoch_f(Tc, Ts, n, m) for n in n_list])

plot(n_list, Pn_list, oper_tytle, pn_tytle, m_tytle)
plot(n_list, ksr_list, oper_tytle, ksr_tytle, m_tytle)
plot(n_list, q_list, oper_tytle, q_tytle, m_tytle)
plot(n_list, poch_list, oper_tytle, poch_tytle, m_tytle)
plot(n_list, loch_list, oper_tytle, loch_tytle, m_tytle)
plot(n_list, qoch_list, oper_tytle, qoch_tytle, m_tytle)

plt.show()
```

Приложение 3

```
# -*- coding: utf-8 -*-

# +------------------------------+
# |    Многоканальная СМО без    |
# | ограничений на длину очереди |
# +------------------------------+

from math import factorial
import matplotlib.pyplot as plt


# Приведенная интенсивность потока
def p_f(Tc, Ts):
    l = 1 / Tc
    m = 1 / Ts
    return l / m


# Вероятность того, что все каналы обслуживания свободны
def p0_f(p, n: int):
    sum = 1
    for k in range(1, n):
        sum += p ** k / factorial(k)
    sum += p ** n / (factorial(n) * (1 - p/n))
    return 1 / sum


# Вероятность того, что k каналов обслуживания заняты
def pk_f(p0, p, k: int):
    return (p ** k) * p0 / factorial(k)


# Вероятность отказа
def pn_f(Tc, Ts, n):
    p = p_f(Tc, Ts)
    P0 = p0_f(p, n)
    return pk_f(P0, p, n)


# Математическое ожидание среднего числа занятых каналов
def ksr_f(Tc, Ts, n):
    p = p_f(Tc, Ts)
    if p/n >= 1:
```

```
            return n
        return p_f(Tc, Ts)


# Коэффициент загрузки операторов
def q_f(Tc, Ts, n):
    return ksr_f(Tc, Ts, n) / n


# Вероятность существования очереди
def poch_f(Tc, Ts, n):
    p = p_f(Tc, Ts)
    if p / n >= 1:
        return 1
    p0 = p0_f(p, n)
    return (p ** n) * n * p0 / (factorial(n) * (n - p))


# Математическое ожидание длины очереди
def loch_f(Tc, Ts, n):
    p = p_f(Tc, Ts)
    if p / n >= 1:
        return -10
    p0 = p0_f(p, n)
    return (p ** (n + 1)) * n * p0 / (factorial(n) * ((n - p) ** 2))


# Функция отрисовки графиков
def plot(x_list: list,
         y_list: list,
         x_tytle: str = "",
         y_tytle: str = ""):
    fig, ax1 = plt.subplots()
    plt.grid(True)
    ax1.plot(x_list, y_list, 'r-')
    ax1.set_xlabel(x_tytle)
    ax1.set_ylabel(y_tytle)


if __name__ == '__main__':
    # Константы
    Tc = 39  # время возникновения заявки
    Ts = 229  # время обслуживания заявки
    n_lim = 12  # предельное количество операторов
```

```
oper_tytle = "Количество операторов"
ksr_tytle = "Мат. ожидание числа занятых операторов"
q_tytle = "Коэффициент загрузки операторов"
poch_tytle = "Вероятность существования очереди"
loch_tytle = "Мат. ожидание длины очереди"

n_list = [n + 1 for n in range(n_lim)]
ksr_list = [ksr_f(Tc, Ts, n) for n in n_list]
q_list = [q_f(Tc, Ts, n) for n in n_list]
poch_list = [poch_f(Tc, Ts, n) for n in n_list]
loch_list = [loch_f(Tc, Ts, n) for n in n_list]

plot(n_list, ksr_list, oper_tytle, ksr_tytle)
plot(n_list, q_list, oper_tytle, q_tytle)
plot(n_list, poch_list, oper_tytle, poch_tytle)
plot(n_list, loch_list, oper_tytle, loch_tytle)

plt.show()
```

Приложение 4

```
# -*- coding: utf-8 -*-

# +------------------------------+
# |    Многоканальная СМО без    |
# | ограничений на длину очереди, |
# |   но с ограниченным временем  |
# |     ожидания в очереди.      |
# +------------------------------+

from math import factorial
import matplotlib.pyplot as plt

def multiply(lst):
    result = 1
    for i in lst:
        result *= i
    return result

# Приведенная интенсивность потока
def p_f(Tc, Ts):
    l = 1 / Tc
    m = 1 / Ts
    return l / m

# Приведенная интенсивность потока ухода
def b_f(Ts, Tw):
    v = 1 / Tw
    m = 1 / Ts
    return v / m

# Вероятность того, что все каналы обслуживания свободны
def p0_f(p, b, n: int, m: int):
    sum = 1
    for k in range(1, n + 1):
        sum += (p ** k) / factorial(k)
    for i in range(1, m + 1):
        sum += (p ** n / factorial(n)) * (p ** i) / multiply([n + l * b for l in range(1, i + 1)])
    return 1 / sum

# Вероятность того, что k каналов обслуживания заняты
def pk_f(p0, p, b, k: int, n: int):
    if k <= n:
        return (p ** k) * p0 / factorial(k)
```

```
    i = k - n
    return pk_f(p0, p, b, n, n) * p ** i / multiply([n + l * b for l in range(1, i + 1)])

def m_f(p, b, n, eps):
    m = 0
    p0 = p0_f(p, b, n, m)
    delta = p0 ** m - p0 ** (m + 1)
    while delta >= eps:
        m += 1
        p0 = p0_f(p, b, n, m)
        delta = p0 ** m - p0 ** (m + 1)
    return m

# Математическое ожидание среднего числа занятых каналов
def ksr_f(p, b, n, m):
    p0 = p0_f(p, b, n, m)
    sum = 0
    for k in range(0, n + 1):
        # sum += k * pk_f(p0, p, b, k, n)
        sum += (k * (p ** k) * p0) / factorial(k)
    sum2 = 0
    for i in range(1, m + 1):
        sum2 += (p ** i) / multiply([n + l * b for l in range(1, i + 1)])

    sum2 *= (n * (p ** n) * p0) / factorial(n)
    sum += sum2
    return sum

# Коэффициент загрузки операторов
def q_f(p, b, n, m):
    return ksr_f(p, b, n, m) / n

# Вероятность существования очереди
def poch_f(p, b, n, m):
    if p / n >= 1:
        return 1
    p0 = p0_f(p, b, n, m)
    sum = 1
    for i in range(1, m):
        sum += p ** i / multiply([n + l * b for l in range(1, i + 1)])
    return (p ** n * p0 / factorial(n)) * sum

# Математическое ожидание длины очереди
def loch_f(p, b, n, m):
    p0 = p0_f(p, b, n, m)
```

```
    sum = 0
    for i in range(1, m + 1):
        sum += i * (p ** i) / multiply([n + l * b for l in range(1, i + 1)])
    return (p ** n * p0 / factorial(n)) * sum

# Функция отрисовки графиков
def plot(x_list: list,
         y_list: list,
         x_tytle: str = "",
         y_tytle: str = ""):
    fig, ax1 = plt.subplots()
    plt.grid(True)
    ax1.plot(x_list, y_list, 'r-')
    ax1.set_xlabel(x_tytle)
    ax1.set_ylabel(y_tytle)

if __name__ == '__main__':
    # Константы
    Tc = 39  # время возникновения заявки
    Ts = 229  # время обслуживания заявки
    Tw = 517  # приемлемое время ожидания
    n_lim = 12  # предельное количество операторов

    oper_tytle = "Количество операторов"
    ksr_tytle = "Мат. ожидание числа занятых операторов"
    q_tytle = "Коэффициент загрузки операторов"
    poch_tytle = "Вероятность существования очереди"
    loch_tytle = "Мат. ожидание длины очереди"

    p = p_f(Tc, Ts)
    b = b_f(Ts, Tw)
    n_list = [n + 1 for n in range(n_lim)]
    m_list = [m_f(p, b, n, 0.000001) for n in n_list]

    ksr_list = [ksr_f(p, b, n, m) for n, m in zip(n_list, m_list)]
    q_list = [q_f(p, b, n, m) for n, m in zip(n_list, m_list)]
    poch_list = [poch_f(p, b, n, m) for n, m in zip(n_list, m_list)]
    loch_list = [loch_f(p, b, n, m) for n, m in zip(n_list, m_list)]

    plot(n_list, ksr_list, oper_tytle, ksr_tytle)
    plot(n_list, q_list, oper_tytle, q_tytle)
    plot(n_list, poch_list, oper_tytle, poch_tytle)
    plot(n_list, loch_list, oper_tytle, loch_tytle)

    plt.show()
```

Приложение 5

```
# -*- coding: utf-8 -*-

# +------------------------------+
# |    Многоканальная СМО без    |
# | ограничений на длину очереди, |
# |   но с ограниченным временем  |
# |     ожидания в очереди.      |
# +------------------------------+

from math import factorial
import matplotlib.pyplot as plt


# Функция отрисовки графиков
def plot(x_list: list,
         y_list: list,
         x_tytle: str = "",
         y_tytle: str = ""):
    fig, ax1 = plt.subplots()
    plt.grid(True)
    ax1.plot(x_list, y_list, 'r-')
    ax1.set_xlabel(x_tytle)
    ax1.set_ylabel(y_tytle)


def pn_f(p, N, n):
    p0 = p0_f(p, N, n)
    return factorial(N) * (p ** n) * p0 / factorial(N - n)


def p0_f(p, N, n):
    if p != 1 / (N - n):
        s = 0
        for i in range(0, n + 1):
            s += (factorial(N) / factorial(N - i)) * p ** i
        s += (factorial(N) / factorial(N - n - 1) * p ** (n + 1) * ((1 - (((N - n) * p) ** (N - n))) /
(1 - (N - n) * p)))
    else:
        s = N - n + 2
        for k in range(1, n + 1):
            s += factorial(N) / ((N - n) ** k * factorial(N - k))
    return 1 / s
```

```
def poch_f(p, N, n):
    if (N - n) * p != 1:
        return pn_f(p, N, n) * (1 - ((N - n) * p) ** (N - n)) * (N - n) * p / (1 - (N - n) * p)
    else:
        return (N - n) * pn_f(p, N, n)


def loch_f(p, N, n):
    s = 0
    for i in range(1, N - n + 1):
        s += i * ((N - n) ** i) * (p ** i)
    return pn_f(p, N, n) * s


def lobs_f(p, N, n):
    s = 0
    for k in range(1, n + 1):
        s += factorial(N) * (p ** k) * k / factorial(N - k)
    return p0_f(p, N, n) * s


def lsmo_f(p, N, n):
    return loch_f(p, N, n) + lobs_f(p, N, n)


def q_f(p, N, n):
    return lobs_f(p, N, n) / n


if __name__ == '__main__':
    # Константы
    Tc = 444  # среднее время между наладками
    Ts = 12  # среднее время наладки
    N = 38  # количество станков
    n_lim = N - 1  # предельное количество операторов

    oper_tytle = "Количество наладчиков"
    lsmo_tytle = "Мат. ожидание числа простаивающих станков"
    loch_tytle = "Мат. ожидание числа ожидающих обслуживание станков"
    poch_tytle = "Вероятность ожидания обслуживания"
    lobs_tytle = "Мат. ожидание числа занятых наладчиков"
    q_tytle = "Коэффициент занятости наладчиков"

    l = 1 / Tc
```

```
m = 1 / Ts
p = l / m
n_list = [n + 1 for n in range(n_lim)]

lsmo_list = [lsmo_f(p, N, n) for n in n_list]
loch_list = [loch_f(p, N, n) for n in n_list]
poch_list = [poch_f(p, N, n) for n in n_list]
lobs_list = [lobs_f(p, N, n) for n in n_list]
q_list = [q_f(p, N, n) for n in n_list]

plot(n_list, lsmo_list, oper_tytle, lsmo_tytle)
plot(n_list, loch_list, oper_tytle, loch_tytle)
plot(n_list, poch_list, oper_tytle, poch_tytle)
plot(n_list, lobs_list, oper_tytle, lobs_tytle)
plot(n_list, q_list, oper_tytle, q_tytle)

plt.show()
```