

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 №5
Дисциплина: Теория надежности

Выполнил: ст. группы ВТ-31
Подкопаев Антон Валерьевич
Проверил: доц. каф. ПО и ВТАС
Кабалянц Петр Степанович

Белгород 2020

Задание для выполнения к работе

На сервере есть n каналов передачи сообщений. Среднее время обработки сообщений τ . На сервер поступают сообщения в среднем количестве λ сообщений в минуту. Для хранения сообщений в очереди на сервере выделено место для $(n+1)$ сообщения. Определить основные характеристики сервера:

1. вероятность очереди;
2. среднее число занятых каналов;
3. среднюю длину очереди;
4. среднее число сообщений на сервере.

Предполагается, что сообщение не получает отказ при занятости всех каналов и очереди длины не больше $(n+1)$. Параметры варианта определяются по формулам: $n = 3 + [(i + j) / 8]$, $\lambda = 1 + i / 4$, $\tau = 5 / (5 + j)$. Здесь квадратные скобки означают взятие целой части, а i, j - последние цифры зачетки.

Ход выполнения работы

$$i = 10, \quad j = 1$$

$m = 5$ - допустимый размер очереди

$$n = 3 + [(10 + 1) / 8] = 4 \text{ канала}$$

$$\tau = 5 / (5 + 1) = 0,83 \text{ с} - \text{среднее время обработки сообщения}$$

$$\lambda = 1 + 10 / 4 = 3,5 - \text{среднее кол-во сообщений в минуту}$$

$$\mu = 1 / \tau = 1 / 0,83 = 1,2 - \text{интенсивность потока обслуживания}$$

$$\rho = \lambda / \mu = 3,5 / 1,2 = 2,92 - \text{приведенная интенсивность потока}$$

Вычислим предельные вероятности:

$$P_0 = \left(1 + \rho + \frac{\rho^2}{2!} + \frac{\rho^3}{3!} + \frac{\rho^4}{4!} + \frac{\rho^5 * \left(1 - \left(\frac{\rho}{4} \right)^5 \right)}{4 * 4! * \left(1 - \frac{\rho}{4} \right)} \right)^{-1}$$
$$P_1 = \rho * P_0; P_2 = \frac{\rho^2}{2!} * P_0; P_3 = \frac{\rho^3}{3!} * P_0; P_4 = \frac{\rho^4}{3 * 3!} * P_0; P_5 = \frac{\rho^5}{3^2 * 3!} * P_0;$$
$$P_6 = \frac{\rho^6}{3^3 * 3!} * P_0; P_7 = \frac{\rho^7}{3^4 * 3!} * P_0; P_7 = \frac{\rho^8}{3^5 * 3!} * P_0; P_7 = \frac{\rho^9}{3^6 * 3!} * P_0$$

$$P = [0,046; 0,134; 0,195; 0,189; 0,139; 0,101; 0,074; 0,054; 0,039; 0,029]$$

Посчитаем требуемые характеристики:

$$P_{\text{отк}} = P[n + m] = 0,029 - \text{вероятность отказа в обработке}$$

$$Q = 1 - P_{\text{отк}} = 0,971 - \text{пропускная способность}$$

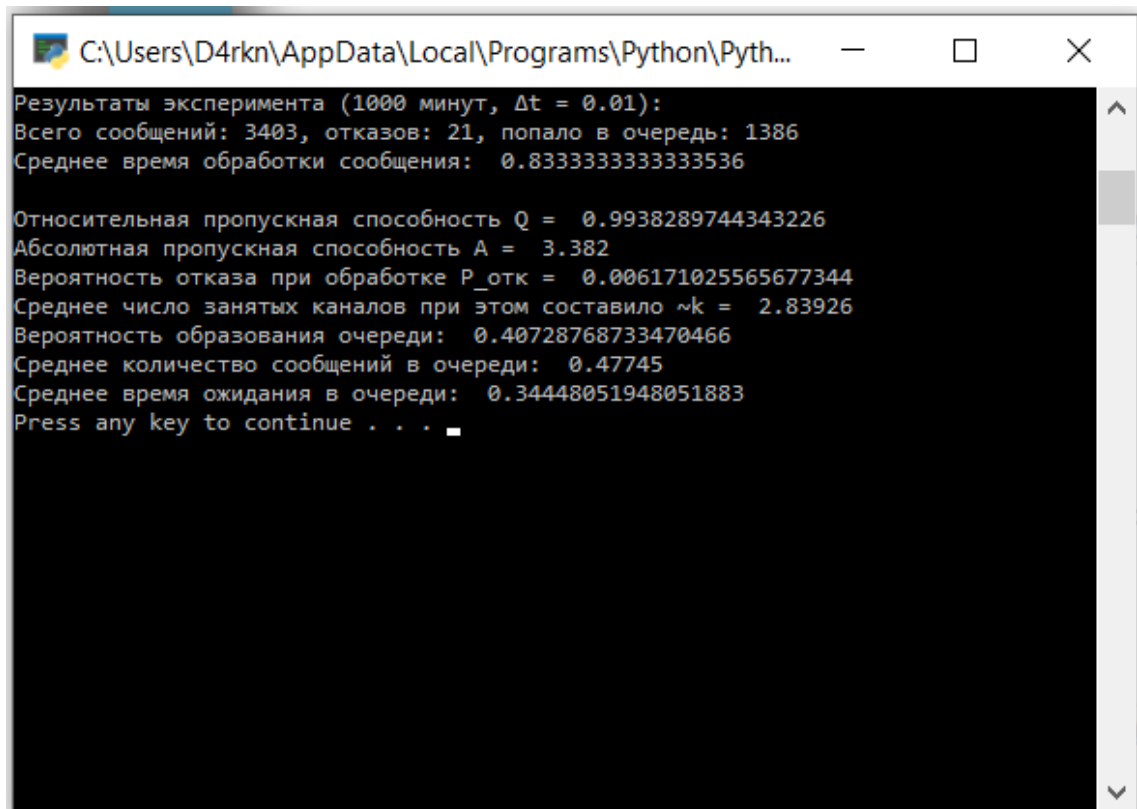
$$A = \lambda * Q = 2 * 0,987 = 3,4 - \text{абсолютная пропускная способность}$$

$$\bar{k} = \rho * \left(1 - \frac{\rho^{n+m}}{n^{m*n!}} P[0]\right) = 2,833 - \text{среднее число занятых каналов}$$

$$P_{\text{оч}} = \sum_{k=n}^{n+m-1} P[k] = 0,406 - \text{вероятность образования очереди}$$

$$L_{\text{оч}} = \frac{\left(\rho^{n+1} * P[0] * \left(1 - \left(m+1 - m * \frac{\rho}{n}\right) * \left(\frac{\rho}{n}\right)^m\right)\right)}{n * n! * \left(1 - \frac{\rho}{n}\right)^2} = 0,708 - \text{средняя длина очереди}$$

$$T_{\text{оч}} = \frac{L_{\text{оч}}}{\lambda} = 0,203 - \text{среднее время ожидания в очереди}$$



```

C:\Users\D4rkn\AppData\Local\Programs\Python\Pyth...
Результаты эксперимента (1000 минут, Δt = 0.01):
Всего сообщений: 3403, отказов: 21, попало в очередь: 1386
Среднее время обработки сообщения: 0.8333333333333333

Относительная пропускная способность Q = 0.9938289744343226
Абсолютная пропускная способность A = 3.382
Вероятность отказа при обработке P_отк = 0.006171025565677344
Среднее число занятых каналов при этом составило ~k = 2.83926
Вероятность образования очереди: 0.40728768733470466
Среднее количество сообщений в очереди: 0.47745
Среднее время ожидания в очереди: 0.34448051948051883
Press any key to continue . . .
  
```

```

import random
import math
import numpy
from scipy.special import factorial as fact

i = 10
j = 1

n = 3 + ((i + j) // 8)          # Количество каналов передачи сообщений
m = n + 1                      # Размер очереди
lamdb = 1 + i / 4               # Интенсивность потока заявок
tau = 5 / (5 + j)              # Среднее время обработки сообщений

print("Количество каналов передачи n = {}\nРазмер очереди m = {}\nСообщений в минуту λ = {}\nСреднее время обработки сообщений τ = {}".format(n, m, lamdb, tau))

mu = 1 / tau                    # Интенсивность потока обслуживания
ro = lamdb / mu                 # Приведенная интенсивность потока заявок (интенсивность нагрузки)
print("Интенсивность потока обслуживания μ = {}\nПриведенная интенсивность потока ρ = {}".format(mu, ro))
P = [0]                         # Предельные вероятности (среднее относительное время, которое канал занят (p0 - все свободны))
for i in range(0, n + 1):
    P[0] += (ro ** i) / fact(i)
P[0] += (ro ** (n + 1)) * (1 - (ro / n) ** m) / (fact(n) * (n - ro))
P[0] = P[0] ** -1
for i in range(1, n + 1):
    P.append((ro ** i) / fact(i) * P[0])
for r in range(1, m + 1):
    P.append((ro ** (n + r)) / (n ** r * fact(n)) * P[0])
print("Предельные вероятности P = {}".format(P))

P_o = P[n + m]                  # Вероятность отказа (все каналы заняты)
Q = 1 - P_o                     # Относительная пропускная способность
A = lamdb * Q                   # Абсолютная пропускная способность
k = ro * (1 - ((ro ** (n + m)) / (n ** m * fact(n)))) * P[0]          # Среднее число занятых каналов
# Средняя длина очереди
L_och = (ro ** (n + 1) * P[0] * (1 - (m + 1 - m * ro/n) * ((ro/n) ** m))) / (n * fact(n) * ((1 - ro/n) ** 2))
T_och = L_och / lamdb
P_och = 0
for i in range(n, n+m):
    P_och += P[i]

print("Теоретические значения:")
print("Относительная пропускная способность Q = ", Q)
print("Абсолютная пропускная способность A = ", A)
print("Вероятность отказа в обработке P_отк = ", P_o)
print("Среднее число занятых каналов ~k = ", k)
print("Вероятность образования очереди: ", P_och)
print("Средняя длина очереди L_оч = ", L_och)
print("Среднее время ожидания в очереди: {}".format(T_och))

# Добавить сообщение в очередь
def pushMessage(message, queue):
    if (len(queue) < m):
        queue.append([message, 0])
        return True
    return False

# Нарастивает время ожидания в очереди
def incWaitTime(queue, dt):
    for i in range(0, len(queue)):
        #if (queue[i][0] > 0):
            queue[i][1] += dt

```

```

# Проверяет каналы на наличие свободных и помещает в них сообщения из очереди
def queueToChannel(channels, queue, averageQueueTime, queueCount):
    pos = freeChannel(channels)
    while (pos != -1 and len(queue) > 0):
        channels[pos] = queue[0][0]
        averageQueueTime[0] += queue[0][1]
        queueCount[0] += 1
        queue.pop(0)
        pos = freeChannel(channels)
# Возвращает номер свободного канала, иначе - -1
def freeChannel(channels):
    for i in range(0, n):
        if channels[i] == 0:
            return i
    return -1
# Обработка уже имеющихся сообщений
def messagesProcessing(channels, dt, queue, averageQueueTime, queueCount):
    for i in range(0, len(channels)):
        if (channels[i] > dt):
            channels[i] -= dt
        else:
            channels[i] = 0
    incWaitTime(queue, dt)
    queueToChannel(channels, queue, averageQueueTime, queueCount)
# Получение нового сообщения
def newMessage(message, channels, queue):
    pos = freeChannel(channels)
    if (pos != -1):
        channels[pos] = message
        return True
    if (pushMessage(message, queue)):
        return True
    return False

channels = [0 for i in range(0, n)] # Каналы связи (0, если не обрабатывается, иначе - оставшееся
время)
queue = []
maxTime = 1000 # Время работы
busyChannels = 0 # Занятые каналы
unProcessedMessages = 0 # Необработанные сообщения
totalMessages = 0 # Всего сообщений
dt = 0.01 # Δt
averageTime = 0 # Среднее время обработки сообщения
averageQueueTime = [0] # Среднее время ожидания в очереди
queueCount = [0] # Общее количество сообщений в очереди
averageQueueCount = 0 # Среднее количество сообщений в очереди
for currentTime in range(0, int(maxTime / dt)):
    if (random.random() < 1 - math.exp(-1 * lambd * dt)): # Если сообщение
        пришло
        message = tau # Назначаем ему
        время обработки
        averageTime += message
        totalMessages += 1
        if(newMessage(message, channels, queue) != True): # Отправляем
        сообщение в свободный канал или очередь
            unProcessedMessages += 1 # Если все каналы
        заняты, сообщение не обработано
        averageQueueCount += len(queue)
        busyChannels += n - channels.count(0)
        messagesProcessing(channels, dt, queue, averageQueueTime, queueCount) # Обрабатываем
        сообщение
    busyChannels = busyChannels / (maxTime / dt)
    averageTime = averageTime / totalMessages

if (queueCount[0] > 0):
    averageQueueTime[0] = averageQueueTime[0] / queueCount[0]
averageQueueCount = averageQueueCount / int(maxTime / dt)

```

```
print("Результаты эксперимента ({} минут,  $\Delta t = \{ \}$ ):".format(maxTime, dt))
print("Всего сообщений: {}, отказов: {}, попало в очередь: {}".format(totalMessages,
unProcessedMessages, queueCount[0]))
print("Среднее время обработки сообщения: ", averageTime)
print("\nОтносительная пропускная способность Q = ", (totalMessages - unProcessedMessages) /
totalMessages)
print("Абсолютная пропускная способность A = ", (totalMessages - unProcessedMessages) / maxTime)
print("Вероятность отказа при обработке P_отк = ", unProcessedMessages / totalMessages)
print("Среднее число занятых каналов при этом составило ~k = ", busyChannels)
print("Вероятность образования очереди: ", queueCount[0]/totalMessages)
print("Среднее количество сообщений в очереди: ", averageQueueCount)
print("Среднее время ожидания в очереди: ", averageQueueTime[0])
```