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»

О Т Ч Е Т

по лабораторной работе № 5

Название: «Определение вероятности отказа»

Дисциплина: Моделирование

Студент ИУ7-76Б (Группа) А. А. Петрова (И.О. Фамилия)

Преподаватель И. В. Рудаков (И.О. Фамилия)

2022 г.

Задание

В информационный центр приходят клиенты через интервалы времени 10 ± 2 минуты. Если все три имеющихся оператора заняты, клиенту отказывают в обслуживании. Операторы имеют разную производительность и могут обеспечивать обслуживание среднего запроса пользователя за 20 ± 5, 40 ± 10 и 40 ± 20 минут. Клиенты стремятся занять свободного оператора с максимальной производительностью. Полученные запросы сдаются в приемный накопитель, откуда они выбираются для обработки. На первый компьютер – запросы от первого и второго оператора, на второй – от третьего оператора. Время обработки на первом и втором компьютере равны соответственно 15 и 30 минутам. Смоделировать процесс обработки 300 запросов и определить вероятность отказа. За единицу системного времени выбрать 0,01 минуты. Построить структурную схему модели и модель в терминах СМО.

Математическая формализация

На рисунке ниже представлена структурная схема моделируемой системы.

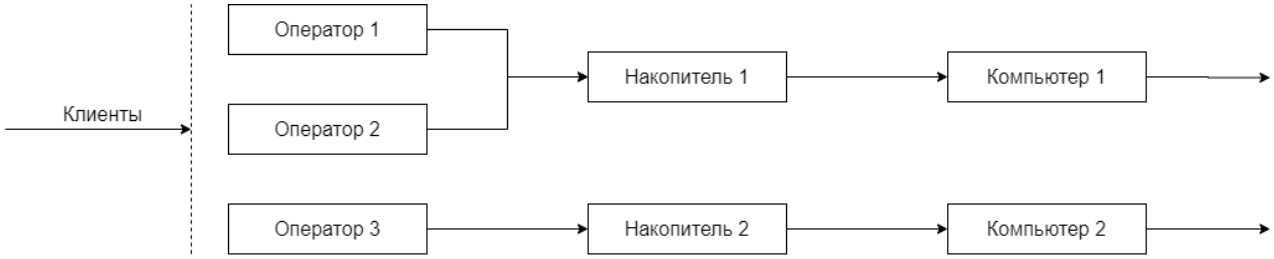

Рисунок 1: структурная схема модели

В процессе взаимодействия клиента с информационным центром возможны два режима:

1. Режим нормального обслуживания (клиент выбирает одного из свободных операторов, отдавая предпочтение тому, у которого меньше номер).
2. Режим отказа в обслуживании клиента (все операторы заняты).

Эндогенные переменные: время обработки задания i-ым оператором и время решения задания на j-ом компьютере.

Экзогенные переменные: число обслуженных клиентов N_0 и число клиентов, получивших отказ, N_1.

Вероятность отказа:

$$P_{\text{отк}} = \frac{N_1}{N_0 + N_1}$$

Ниже представлена схема модели в терминах СМО.

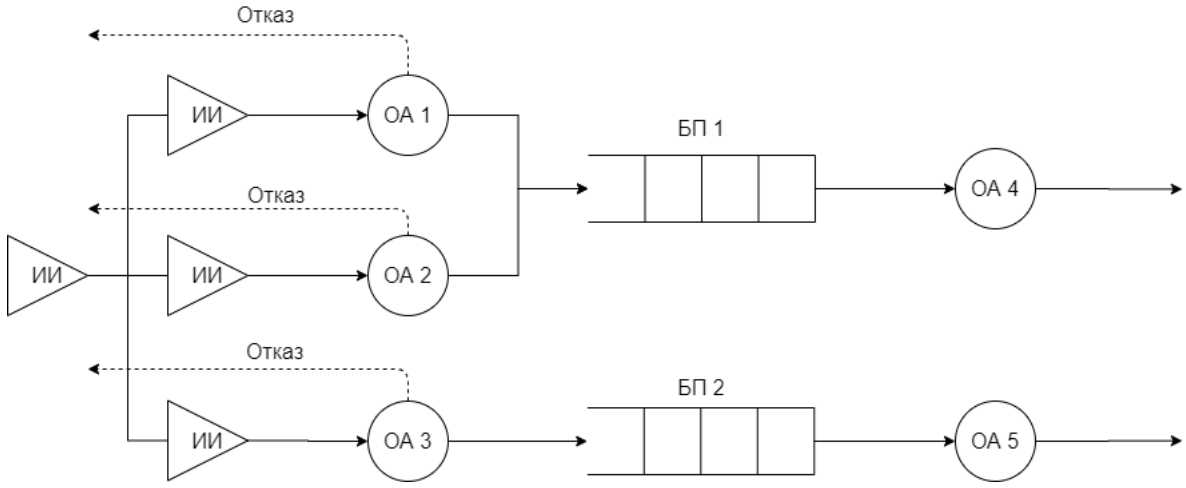

Рисунок 2: модель в терминах СМО

Где ИИ – это источник информации (заявка), БП – буферная память (приемный накопитель), а ОА – обслуживающий аппарат (операторы и компьютеры).

Реализация

В листингах ниже представлена реализация генератора заявок, оператора и компьютера.

Листинг 1: генератор заявок

```
class TimeProcessor:
    def __init__(self, time_distribution):
        self.time_distribution = time_distribution
        self.remaining_time = 0

    def update_time(self):
        if self.remaining_time > 0:
            self.remaining_time -= TIME_DELTA

        if self.remaining_time <= 1e-5:
            self.remaining_time = self.time_distribution.get_time()
            return Request()

        return None
```

Листинг 2: оператор

```
class Operator:
    def __init__(self, recipient, time_distribution):
        self.time_distribution = time_distribution
        self.recipient = recipient
        self.remaining_time = 0
        self.is_busy = False
        self.processing_request = None

    def update_time(self):
```

```
        self.remaining_time -= TIME_DELTA
        if self.is_busy and self.remaining_time <= 1e-5:
            self.finish_process_request()

    def start_process_new_request(self, request):
        self.is_busy = True
        self.processing_request = request
        self.remaining_time = self.time_distribution.get_time()

    def finish_process_request(self):
        self.recipient.append(self.processing_request)
        self.is_busy = False
        self.processing_request = None
```

Листинг 3: компьютер

```
class Processor:
    def __init__(self, requests_storage, time_distribution):
        self.requests_storage = requests_storage
        self.time_distribution = time_distribution
        self.is_busy = False
        self.processing_request = None
        self.remaining_time = 0

    def update_time(self):
        if self.remaining_time != 0:
            self.remaining_time -= TIME_DELTA

        if self.is_busy and self.remaining_time <= 1e-5:
            self.is_busy = False
            self.processing_request = None
            return FINISH_PROCESS_REQUEST

        if not self.is_busy and len(self.requests_storage) != 0:
            self.processing_request = self.requests_storage.pop(0)
            self.remaining_time = self.time_distribution.get_time()
            self.is_busy = True
```

Результаты работы

Ниже представлены результаты работы программы для 300 заявок.

```
Время работы программы (в секундах):  0.9215
Общее количество запросов:  300
Количество обработанных запросов:  232
Количество запросов, которым было отказано в обработке:  68
Вероятность отказа:  0.2267
```

Рисунок 3: работа программы при 300 заявках