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»

Отчет по лабораторной работе № 6 по дисциплине "Анализ алгоритмов"

Тема Муравьиный алгоритм

Студент Хамзина Р. Р.

Группа ИУ7-53Б

Оценка (баллы)

Преподаватель Волкова Л. Л.

Москва — 2021 г.

Содержание

Введение **3**

1 Аналитическая часть **5**

1.1 Задача коммивояжера 5

1.1.1 Математическая модель 5

1.2 Полный перебор 7

1.3 Муравьиный алгоритм 7

1.4 Вывод 9

2 Конструкторская часть **11**

2.1 Разработка алгоритмов 11

2.2 Описание используемых типов и структур данных 18

2.3 Структура разрабатываемого ПО 18

2.4 Классы эквивалентности при тестировании 19

2.5 Вывод 19

3 Технологическая часть **21**

3.1 Средства реализации 21

3.2 Сведения о модулях программы 21

3.3 Листинги кода 21

3.4 Функциональные тесты 26

3.5 Вывод 27

4 Исследовательская часть **28**

4.1 Технические характеристики 28

4.2 Демонстрация работы программы 29

4.3 Время выполнения алгоритмов 29

4.4 Параметризация 31

4.4.1 Класс данных 1 32

4.4.2 Класс данных 2 34

4.5 Вывод 35

Заключение **37**

Список литературы **38**

Приложение 1 **39**

Приложение 2 **59**

Введение

Современные средства навигации, организация логистики, конвейерного производства, анализ эффективности финансовых инструментов строятся на алгоритмах решения задачи поиска оптимального решения по выбранному параметру в сложной системе. Данную задачу высокой вычислительной сложности называют задачей коммивояжера [1].

Задачи высокой вычислительной сложности могут быть решены при помощи полного перебора вариантов и эвристических алгоритмов [2]. Смысл понятия "эвристический алгоритм" состоит в том, что в этом случае алгоритм не вытекает из строгих положений теории, а в значительной степени основан на интуиции и опыте. Такие методы могут давать удовлетворительные результаты при вероятностных параметрах. Алгоритмы, основанные на использовании эвристических алгоритмов, не всегда приводят к оптимальным решениям. Однако для их применения на практике достаточно, чтобы ошибка прогнозирования не превышала допустимого значения, а этого можно добиться, например, подбором более информативных параметров.

Целью данной лабораторной является сравнительный анализ метода полного перебора и эвристического метода на базе муравьиного алгоритма. Для достижения поставленной цели требуется выполнить следующие задачи:

- изучить задачу коммивояжера;
- рассмотреть методы ее решения: полный перебор вариантов и муравьиный алгоритм;
- привести схемы изучаемых алгоритмов;
- описать используемые типы и структуры данных;
- описать структуру разрабатываемого программного обеспечения;
- определить средства программной реализации выбранных алгоритмов;
- реализовать разработанные алгоритмы;

- провести функциональное тестирование программного обеспечения;
- провести сравнительный анализ по времени реализованных алгоритмов;
- провести параметризацию муравьиного алгоритма;
- подготовить отчет о выполненной лабораторной работе.

1 Аналитическая часть

В данном разделе будет рассмотрена задача коммивояжера и будут описаны алгоритмы её решения.

1.1 Задача коммивояжера

Цель задачи коммивояжера [3] заключается в нахождении самого выгодного маршрута (кратчайшего, самого быстрого, наиболее дешевого), проходящего через все заданные точки (пункты, города) по одному разу.

Условия задачи должны содержать критерий выгодности маршрута (должен ли он быть максимально коротким, быстрым, дешевым или все вместе), а также исходные данные в виде матрицы затрат (расстояния, стоимости, времени) при перемещении между рассматриваемыми пунктами.

1.1.1 Математическая модель

Исходные условия можно представить в виде взвешенного графа - конечного множества вершин и множества ребер, соединяющих вершины. Вершины символизируют города, ребра - пути между городами, вес ребра - стоимость пути.

Рассмотрим взевешенный граф на рисунке 1.1. Самый выгодный маршрут для данного графа равен 14 ($1- > 3- > 2- > 4- > 1$).

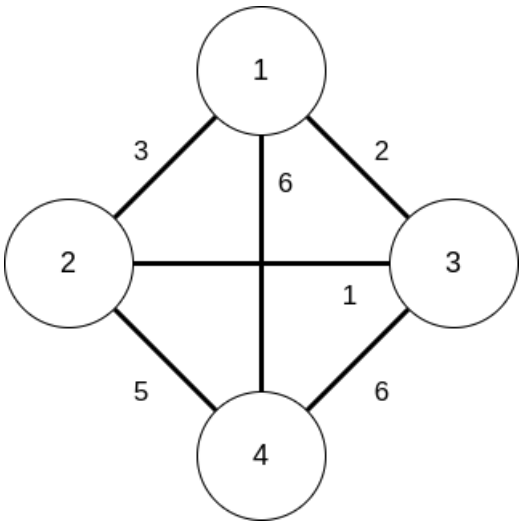

Рисунок 1.1 – Взвешенный граф

По симметричности задача коммивояжера бывает:

- симметричная - все пары ребер, соединяющие одни и те же вершины, имеют одинаковый вес (граф неориентированный);
- ассиметричная - вес пар ребер соединяющих одни и те же города, может различаться (ориентированный граф).

По замкнутости маршрута задача бывает:

- замкнутая - нахождение кратчайшего пути, проходящего через все вершины по одному разу с последующим возвратом в точку старта;
- незамкнутая — нахождение кратчайшего пути, проходящего через все вершины по одному разу и без обязательного возврата в исходную точку.

Далее будут рассмотрены алгоритмы решения симметричной замкнутой задачи коммивояжера.

Граф можно представить при помощи матрицы — таблицы, где строкам соответствуют города отправления, столбцам — города прибытия, а в ячейках указывается стоимость пути. Для графа на рисунке 1.1 матрица стоимостей будет следующая:

$$A = \begin{pmatrix} 0 & 3 & 2 & 6 \\ 3 & 0 & 1 & 5 \\ 2 & 1 & 0 & 6 \\ 6 & 5 & 6 & 0 \end{pmatrix} \tag{1.1}$$

1.2 Полный перебор

Рассмотрим n городов и матрицу расстояний между ними. Найдем самый короткий маршрут посещения всех городов ровно по одному разу, без возвращения в первый город:

- число вариантов выбрать первый город равно n;
- число вариантов выбрать второй город равно $n - 1$;
- с каждым выбором следующего города число вариантов уменьшается на 1;
- число всех вариантов маршрутра равно $n!$;
- минимальный по сумме значений матрицы расстояний вариант маршрута - искомый.

В связи со сложностью $n!$ полный перебор вариантов занимает существенное время, а при большом количестве городов становится технически невозможным.

1.3 Муравьиный алгоритм

Идея муравьиного алгоритма [4] — моделирование поведения муравьев, связанное с их способностью быстро находить кратчайший путь и адаптироваться к изменяющимся условиям, находя новый кратчайший путь.

Муравьи действуют согласно правилам:

- муравей запоминает посещенные города, причем каждый город может быть посещен только один раз. Обозначим через $J_{i,k}$ список городов, которые посетил муравей k, находящийся в городе i;

- муравей обладает видимостью η_{ij} - эвристическим желанием посетить город j, если муравей находится в городе i, причем

$$\eta_{ij} = 1/D_{ij}, \tag{1.2}$$

  где D_{ij} — стоимость пути из города i в город j;

- муравей может улавливать след феромона - специального химического вещества. Число феромона на пути из города i в город j - τ_{ij}.

Муравей выполняет следующую последовательность действий, пока не посетит все города:

- выбирает следующий город назначения, основываясь на вероятностно-пропорциональном правиле (1.3), в котором учитываются видимость и число феромона:

$$P_{ij,k} = \begin{cases} \frac{\tau_{ij}^{\alpha}\eta_{ij}^{\beta}}{\sum_{l=1}^{m}\tau_{il}^{\alpha}\eta_{il}^{\beta}}, \text{если город j необходимо посетить;} \\ 0, \text{иначе,} \end{cases} \tag{1.3}$$

  где α - параметр влияния феромона, β - параметр влияния видимости пути, τ_{ij} - число феромона на ребре (ij), η_{ij} - эвристическое желание посетить город j, если муравей находится в городе i. Выбор города является вероятностным, данное правило определяет ширину зоны города j, в общую зону всех городов $J_{i,k}$ бросается случайное число, которое и определяет выбор муравья;

- муравей проходит путь (ij) и оставляет на нем феромон.

Информация о числе феромона на пути используется другими муравьями для выбора пути. Те муравьи, которые случайно выберут кратчайший путь, будут быстрее его проходить, и за несколько передвижений он будет более обогащен феромоном. Следующие муравьи будут предпочитать именно этот путь, продолжая обогащать его феромоном.

После прохождения маршрутов всеми муравьями значение феромона на путях обновляется в соответствии со следующим правилом (1.4):

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \Delta\tau_{ij}, \tag{1.4}$$

где ρ - коэффициент испарения. Чтобы найденное локальное решение не было единственным, моделируется испарение феромона.

При этом

$$\Delta\tau_{ij} = \sum_{k=1}^{m} \tau_{ij,k}, \tag{1.5}$$

где m - число муравьев,

$$\Delta\tau_{ij,k} = \begin{cases} Q/L_k, \text{если k-ый муравей прошел путь (i,j);} \\ 0, \text{иначе.} \end{cases} \tag{1.6}$$

1.4 Вывод

Была изучена задача поиска оптимального маршрута, проходящего через все заданные вершины по одному разу. Были рассмотрены подходы к решению замкнутой симметричной задачи коммивояжера.

Программе, реализующей данные алгоритмы, на вход должна подаваться матрица стоимостей, которая задает взвешенный неориентированный граф. Выходными данными такой программы должны быть оптимальный маршрут, проходящий через все заданные вершины по одному разу с последующим возвратом в исходную точку, и его стоимость. Программа должна работать в рамках следующих ограничений:

- стоимости путей должны быть целыми числами;
- число городов должно быть больше 1;
- число дней должно быть больше 0;
- параметры муравьиного алгоритма должны быть вещественными числами, большими 0;
- матрица должна задавать неориентированный граф;

- должно быть выдано сообщение об ошибке при некорректном вводе параметров.

Пользователь должен иметь возможность выбора метода решения - полным перебором или муравьиным аогоритмом, и вывода результата на экран. Кроме того должна быть возможность проведения параметризации муравьиного алгоритма. Также должны быть реализованы сравнение алгоритмов по времени работы с выводом результатов на экран и получение графического представления результатов сравнения. Данные действия пользователь должен выполнять при помощи меню.

2 Конструкторская часть

В данном разделе будут представлены схемы алгоритмов: полного перебора и муравьиного алгоритма. Будут описаны типы и структуры данных, используемые для реализации, а также структура разрабатываемого программного обеспечения. Кроме того, будут выделены классы эквивалентности для тестирования.

2.1 Разработка алгоритмов

На рисунке 2.1 приведена схема алгоритма решения задачи коммивояжера полным перебором. Схемы муравьиного алгоритма приведена на рисунке 2.2, вспомогательные функции данного алгоритма показаны на рисунках 2.4-2.6.

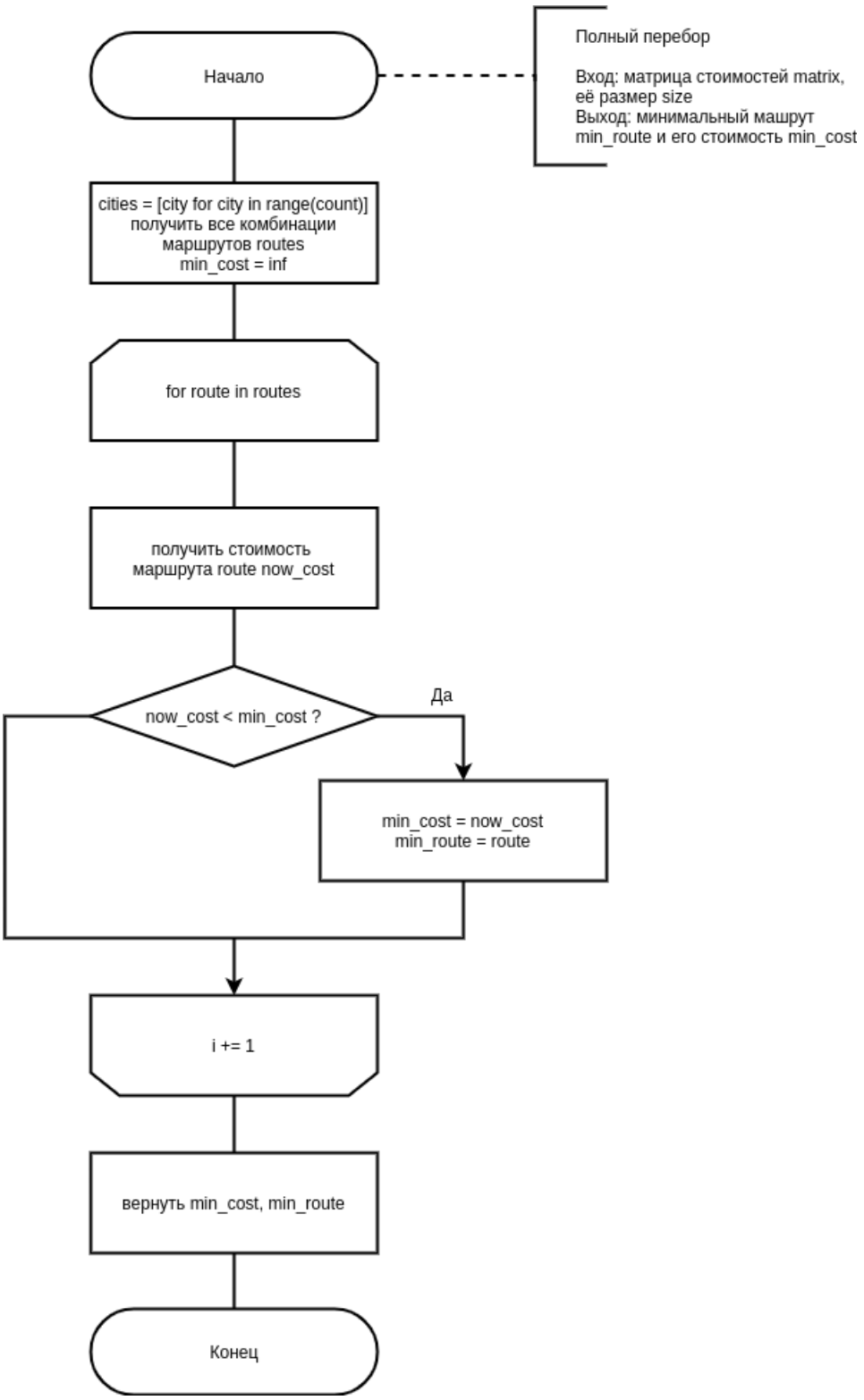

Рисунок 2.1 – Полный перебор

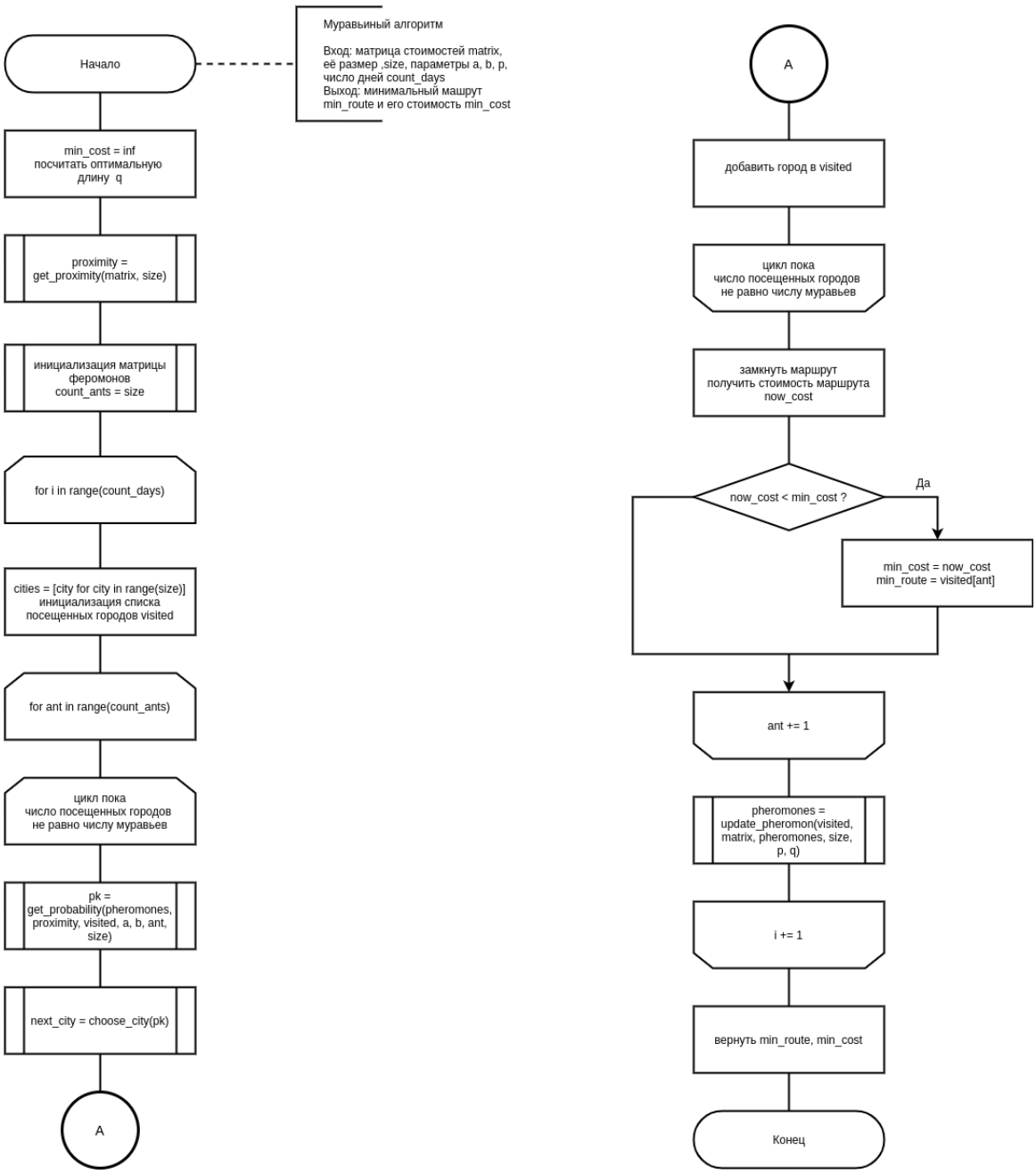

Рисунок 2.2 – Муравьиный алгоритм

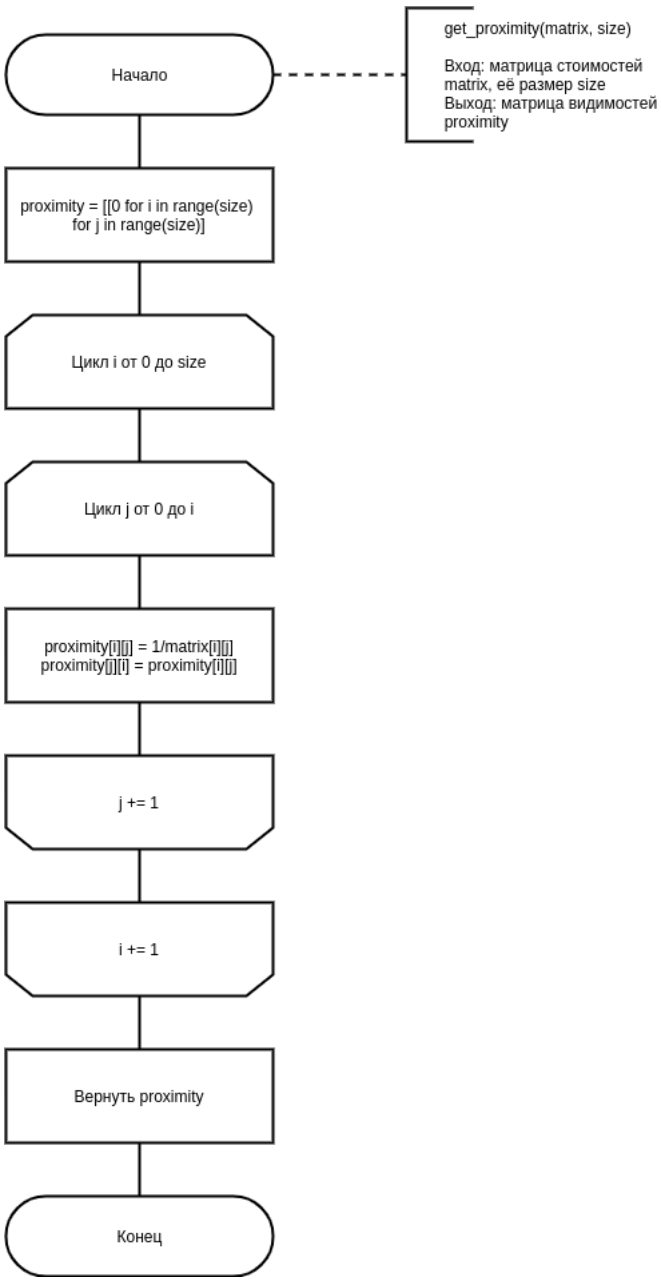

Рисунок 2.3 – Алгоритм вычисления матрицы видимостей

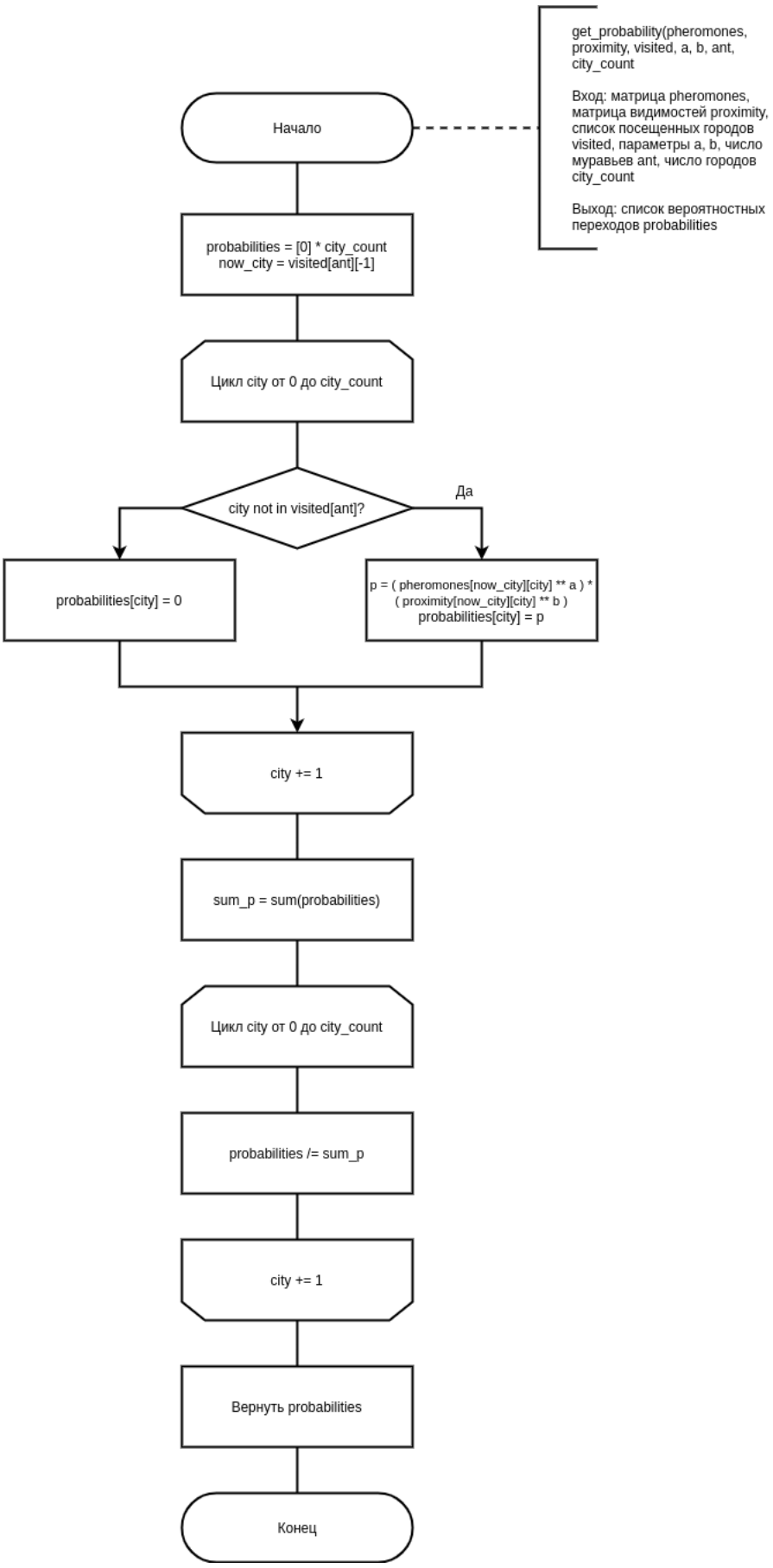

Рисунок 2.4 – Алгоритм вычисления списка вероятностных переходов

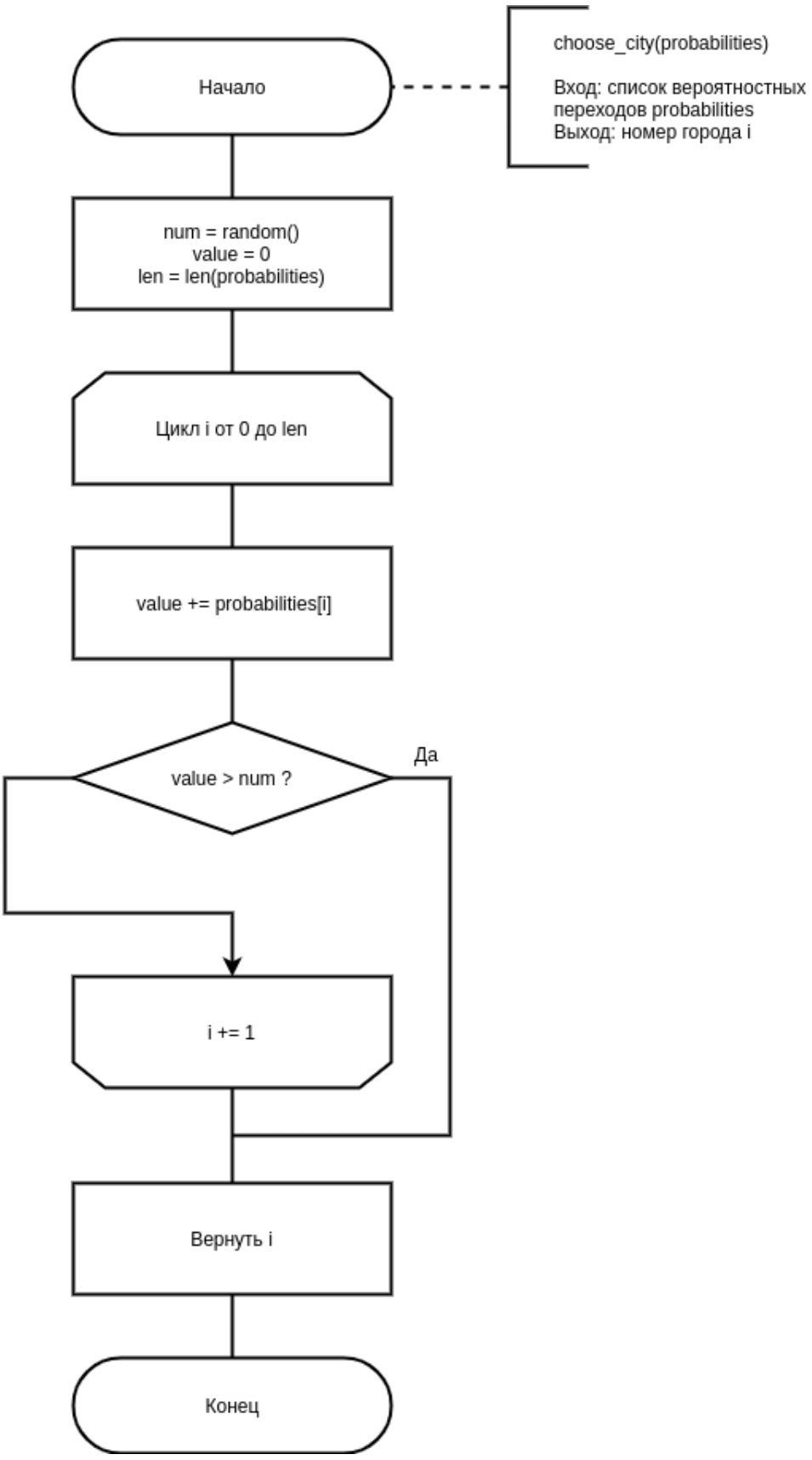

Рисунок 2.5 – Алгоритм выбора следующего города случайным образом

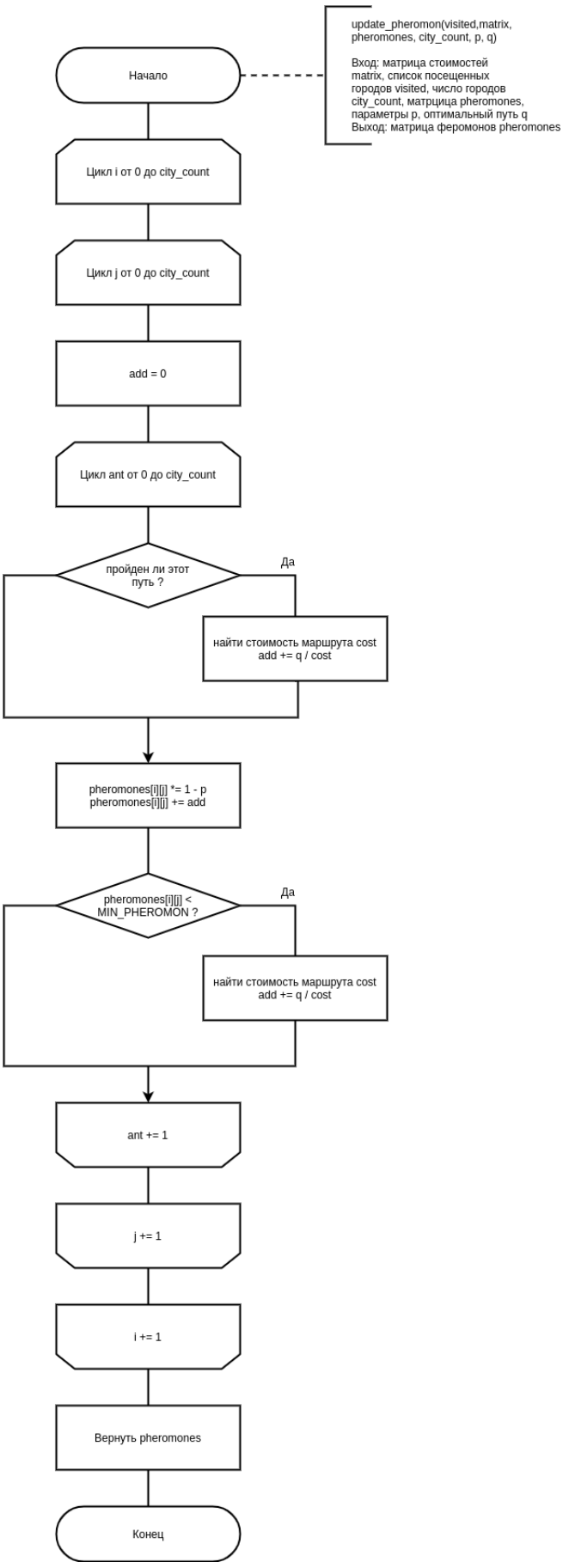

Рисунок 2.6 – Алгоритм обновления феромона

2.2 Описание используемых типов и структур данных

Для задания числа городов и количества дней поиска будет использован тип данных int, для задания параметров α, β и коэффициента ρ - тип данных $float$.

Структура данных - матрица - представляет собой двумерный список значений типа int.

2.3 Структура разрабатываемого ПО

При реализации разрабатываемого программного обеспечения будет использоваться метод структурного программирования. Для взаимодействия с пользователем будет выделена функция $process()$, из которой будут вызываться методы решения задачи коммивояжера, функция параметризации и функции сравнительного анализа. Методы поиска оптимального маршрута будут реализованы в следующих функциях, выходные параметры которых - искомый маршрут и его стоимость:

- функция полного перебора маршрутов, у которой на входе - матрица стоимостей и число городов;
- функция муравьиного алгоритма, входными параметрами которой являются матрица стоимостей, число городов, параметры α, β, коэффициент ρ, число дней поиска.

Для работы с матрицей стоимостей будут разработаны следущие функции:

- создание матрицы случайных чисел, входным параметром функции является размер, выходным - матрица;
- ввод матрицы, входной параметр - размер, выходной параметр - введенная матрица;

- процедура вывода матрицы, входным параметром которой является матрица;
- чтение матрицы из файла, у функции на входе - имя файла, а на выходе - матрица и ее размер.

Для параметризации будет выделена функция, выходными параметрами которой являются файлы с результатами для двух классов данных.

Для сравнительного анализа будут реализованы:

- функция замеров времени, выходным парметром которой является массив временных значений;
- функция графического представления замеров времени, у которой на входе - массив временных значений, на выходе - его графическое представление.

2.4 Классы эквивалентности при тестировании

Для тестирования разрабатываемой программы будут выделены следующие классы эквивалентности:

- неверный ввод параметров α, β, коэффициента ρ и числа дней - не число или число, меньшее или равное 0;
- неверный ввод числа дней - не число или число, меньшее 1;
- неверный ввод числа городов - не число или число, меньшее 2;
- корректный ввод всех параметров.

2.5 Вывод

Были представлены схемы поиска оптимального пути, проходящего через все города по одному разу. Были указаны типы и структуры данных,

используемые для реализации, и описана структура разрабатываемого программного обеспечения. Также были выделены классы эквивалентности для тестирования ПО.

3 Технологическая часть

В данном разделе будут указаны средства реализации, будут представлены листинги кода, а также функциональные тесты.

3.1 Средства реализации

Реализация данной лабораторной работы выполнялась при помощи языка программирования Python [5]. Выбор ЯП обусловлен простотой синтаксиса, большим числом библиотек и эффективностью визуализации данных.

Замеры времени проводились при помощи функции process_time из библиотеки time [6].

3.2 Сведения о модулях программы

Программа состоит из следующих модулей:

- main.py - главный файл программы, предоставляющий пользователю меню для выполнения основных функций;
- matrix.py - файл, содержащий функции работы с матрицами;
- salesman.py - файл, содержащий метды решения задачи коммивояжера;
- time_test.py - файл, содержащий функции замеров времени работы указанных алгоритмов;
- graph_result.py - файл, содержащий функции визуализации временных характеристик описанных алгоритмов.

3.3 Листинги кода

Полный перебор приведен в листинге 3.1, муравьиный алгоритм - в листнге 3.2. Алгоритмы, используемые в муравьином алгоритмы представ-

лены в листингах 3.3-3.5.

Листинг 3.1 – Полный перебор

```
def get_cost(matrix, route):
    now_cost = 0

    for num in range(len(route) - 1):
        start_city = route[num]
        end_city = route[num + 1]

        now_cost += matrix[start_city][end_city]

    return now_cost


def get_all_routes(cities):
    routes = []

    for route in itertools.permutations(cities, len(cities)):
        route = list(route)
        route.append(route[0])
        routes.append(route)

    return routes


def brute_force(count, matrix):
    cities = [city for city in range(count)]
    routes = get_all_routes(cities)

    min_cost = inf

    for route in routes:
        now_cost = get_cost(matrix, route)

        if now_cost < min_cost:
            min_cost = now_cost
            min_rout = route

    return min_rout, min_cost
```

Листинг 3.2 – Муравьиный алгоритм

```
def ant_algorithm(matrix, size, a, b, p, count_days):
    min_cost = inf

    q = get_q(matrix, size)

    proximity = get_proximity(matrix, size)
    pheromones = [[START_PHEROMON for i in range(size)] for j in
       range(size)]
    count_ants = size

    for _ in range(count_days):
        cities = [city for city in range(size)]
        visited = get_visited_cities(cities, count_ants)

        for ant in range(count_ants):
            while len(visited[ant]) != count_ants:
                pk = get_probability(pheromones, proximity,
                                     visited, a, b, ant, size)
                next_city = choose_city(pk)
                visited[ant].append(next_city)
            visited[ant].append(visited[ant][0])

            now_cost = get_cost(matrix, visited[ant])

            if now_cost < min_cost:
                min_cost = now_cost
                min_route = visited[ant]

        pheromones = update_pheromon(visited, matrix, pheromones,
           size, p, q)

    return min_route, min_cost
```

Листинг 3.3 – Алгоритм получения матрицы видимостей городов

```
def get_proximity(matrix, size):
    proximity = [[0 for i in range(size)] for j in range(size)]

    for i in range(size):
        for j in range(i):
            proximity[i][j] = 1 / matrix[i][j]
            proximity[j][i] = proximity[i][j]

    return proximity
```

Листинг 3.4 – Алгоритм нахождения вероятностных переходов

```
def get_probability(pheromones, proximity, visited, a, b, ant,
    city_count):
    probabilities = [0] * city_count

    now_city = visited[ant][-1]

    for city in range(city_count):
        if city not in visited[ant]:

            p = ( pheromones[now_city][city] ** a ) * \
                ( proximity[now_city][city] ** b )

            probabilities[city] = p
        else:
            probabilities[city] = 0

    sum_p = sum(probabilities)

    for city in range(city_count):
        probabilities[city] /= sum_p

    return probabilities
```

Листинг 3.5 – Алгоритм обновления феромона на путях

```
def get_q(matrix, size):
    q = 0
    count = 0

    for i in range(size):
        for j in range(i + 1, size):
            q += matrix[i][j] * 2
            count += 2

    return q / count


def check_route(i, j, route):
    if i in route:
        index_i = route.index(i)

        if j == route[index_i + 1]:
            return True

    return False


def update_pheromon(visited, matrix, pheromones, city_count, p,
    q):
    for i in range(city_count):
        for j in range(city_count):
            add = 0

            for ant in range(city_count):
                if check_route(i, j, visited[ant]):
                    cost = get_cost(matrix, visited[ant])
                    add += q / cost

            pheromones[i][j] *= (1 - p)
            pheromones[i][j] += add

            if pheromones[i][j] < MIN_PHEROMON:
                pheromones[i][j] = MIN_PHEROMON

    return pheromones
```

Листинг 3.6 – Алгоритм выбора города случайным образом

```
def choose_city(probabilities):
    num = random()
    value = 0

    for i in range(len(probabilities)):
        value += probabilities[i]

        if value > num:
            return i
```

3.4 Функциональные тесты

В таблице 3.1 приведены функциональные тесты для функций, реализующих полный перебор, в таблице 3.2 - для функций, реализующих муравьиный алгоритм. Все тесты пройдены успешно.

Таблица 3.1 – Функциональные тесты (полный перебор)

Матрица	Число городов	Ожидаемый результат
$\begin{pmatrix} 0 & 5 & 2 & 5 \\ 5 & 0 & 1 & 1 \\ 2 & 1 & 0 & 1 \\ 5 & 1 & 1 & 0 \end{pmatrix}$	4	[1 2 4 3 1], 9
-	-2	Ошибка

Таблица 3.2 – Функциональные тесты (муравьиный алгоритм)

Матрица	Число городов	α	ρ	Число дней	Результат
$\begin{pmatrix} 0 & 5 & 2 & 5 \\ 5 & 0 & 1 & 1 \\ 2 & 1 & 0 & 1 \\ 5 & 1 & 1 & 0 \end{pmatrix}$	4	0.1	0.3	200	[1 3 2 4 1], 9
-	-2	0.1	0.3	200	Ошибка
$\begin{pmatrix} 0 & 5 & 2 & 5 \\ 5 & 0 & 1 & 1 \\ 2 & 1 & 0 & 1 \\ 5 & 1 & 1 & 0 \end{pmatrix}$	4	-2	0.3	200	Ошибка
$\begin{pmatrix} 0 & 5 & 2 & 5 \\ 5 & 0 & 1 & 1 \\ 2 & 1 & 0 & 1 \\ 5 & 1 & 1 & 0 \end{pmatrix}$	4	0.1	-5	200	Ошибка
$\begin{pmatrix} 0 & 5 & 2 & 5 \\ 5 & 0 & 1 & 1 \\ 2 & 1 & 0 & 1 \\ 5 & 1 & 1 & 0 \end{pmatrix}$	4	0.1	0.3	0	Ошибка

3.5 Вывод

Были реализованы функции алгоритмов решения задачи коммивояжера. Было проведено функциональное тестирование указанных функций.

4 Исследовательская часть

В данном разделе будут приведены примеры работы программы, и будет проведен сравнительный анализ реализованных алгоритмов умножения матриц по затраченному процессорному времени.

4.1 Технические характеристики

Тестирование проводилось на устройстве со следующими техническими характеристиками:

- операционная система: Ubuntu 20.04.1 Linux x86_64 [7];
- память : 8 GiB;
- процессор: AMD® Ryzen™ 3 3200u @ 2.6 GHz [8].

Тестирование проводилось на ноутбуке, включенном в сеть электропитания. Во время тестирования ноутбук был нагружен только встроенными приложениями окружения, а также непосредственно системой тестирования.

4.2 Демонстрация работы программы

На рисунке 4.1 приведен пример работы программы.

```
МЕНЮ:
1. Полный перебор
2. Муравьиный алгоритм
3. Оба алгоритма
4. Построить графики
5. Замерить время
6. Параметризация
0. Выход
Выбор: 3

Введите число городов: 6

Выберите способ ввода матрицы стоимостей:
    1. Ручной ввод
    2. Случайная матрица
    3. Загрузить из файла
Выбор: 2

Введите нижнюю границу стоимостей: 1
Введите верхнюю границу стоимостей: 9

Исходная матрица стоимостей:
0 8 5 4 9 4
8 0 9 6 7 1
5 9 0 6 9 6
4 6 6 0 4 1
9 7 9 4 0 1
4 1 6 1 1 0


=== Полный перебор ===
Самый короткий маршрут:
1 -> 3 -> 2 -> 6 -> 5 -> 4 -> 1
Его стоимость :  24

Введите параметр α: 0.5
Введите коэффициент испарения: 0.3
Введите число дней: 200


=== Муравьиный алгоритм ===
Самый короткий маршрут:
1 -> 4 -> 5 -> 6 -> 2 -> 3 -> 1
Его стоимость :  24
```

Рисунок 4.1 – Пример работы программы

4.3 Время выполнения алгоритмов

Функция process_time из библиотеки time ЯП Python возвращает процессорное время в секундах - значение типа float.

Для замера времени необходимо получить значение времени до начала выполнения алгоритма, затем после её окончания. Чтобы получить результат, необходимо вычесть из второго значения первое.

Замеры проводились для матриц стоимостей, заполненных случайным образом, размером от 2 до 10. Результаты измерения времени приведены в таблице 4.1 (в с), а их графическое представление - на рисунке 4.2.

Таблица 4.1 – Результаты замеров времени

Число городов	Полный перебор	Муравьиный алгоритм
2	0.000327	0.010621
3	0.000039	0.015454
4	0.000121	0.031155
5	0.000706	0.049366
6	0.003409	0.078147
7	0.024055	0.106820
8	0.223502	0.149278
9	2.248946	0.227690
10	22.530937	0.299644

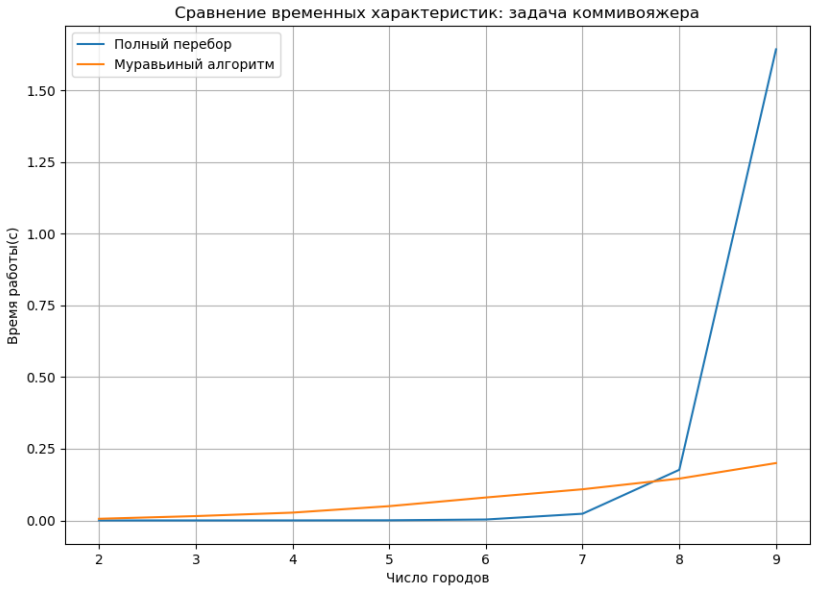

Рисунок 4.2 – Сравнение по времени алгоритмов задачи коммивояжера

4.4 Параметризация

Целью проведения параметризации является определение таких комбинаций параметров, при которых муравьиный алгоритм даёт наилучшие результаты.

В результате автоматической параметризации будет получена таблицы со следующими столбцами:

- коэффициент видимости α - изменяющийся параметр;
- коэффициент испарения феромона ρ - изменяющийся параметр;
- число дней $days$ - изменяющийся параметр;
- эталонный результат $ideal$;
- разность полученного при данных параметрах значения и эталонного $mistake$.

4.4.1 Класс данных 1

Класс данных 1 представляет собой матрицу стоимостей в диапозоне от 1 до 5 для 8 городов:

$$K_1 = \begin{pmatrix} 0 & 4 & 3 & 2 & 4 & 5 & 5 & 3 \\ 4 & 0 & 4 & 5 & 5 & 1 & 4 & 5 \\ 3 & 4 & 0 & 5 & 4 & 1 & 2 & 1 \\ 2 & 5 & 5 & 0 & 1 & 3 & 1 & 5 \\ 4 & 5 & 4 & 1 & 0 & 2 & 5 & 4 \\ 5 & 1 & 1 & 3 & 2 & 0 & 4 & 5 \\ 5 & 4 & 2 & 1 & 5 & 4 & 0 & 2 \\ 3 & 5 & 1 & 5 & 4 & 5 & 2 & 0 \end{pmatrix} \tag{4.1}$$

Для данного класса данных приведена таблица с выборкой параметров, которые наилучшим образом решают поставленную задачу.

Таблица 4.2 – Параметры для класса данных 1

α	ρ	Days	Result	Mistake
0.1	0.9	100	15	0
0.1	0.9	200	15	0
0.1	0.9	300	15	0
0.1	0.9	400	15	0
0.1	0.9	500	15	0
0.2	0.8	100	15	0
0.2	0.8	200	15	0
0.2	0.8	300	15	0
0.2	0.8	400	15	0
0.2	0.8	500	15	0
0.3	0.7	100	15	0
0.3	0.7	200	15	0
0.3	0.7	300	15	0
0.3	0.7	400	15	0
0.3	0.7	500	15	0

0.4	0.6	100	15	0
0.4	0.6	200	15	0
0.4	0.6	300	15	0
0.4	0.6	400	15	0
0.4	0.6	500	15	0
0.5	0.5	100	15	0
0.5	0.5	200	15	0
0.5	0.5	300	15	0
0.5	0.5	400	15	0
0.5	0.5	500	15	0
0.6	0.4	100	15	0
0.6	0.4	200	15	0
0.6	0.4	300	15	0
0.6	0.4	400	15	0
0.6	0.4	500	15	0
0.7	0.3	100	15	0
0.7	0.3	200	15	0
0.7	0.3	300	15	0
0.7	0.3	400	15	0
0.7	0.3	500	15	0
0.8	0.2	100	15	0
0.8	0.2	200	15	0
0.8	0.2	300	15	0
0.8	0.2	400	15	0
0.8	0.2	500	15	0
0.9	0.1	100	15	0
0.9	0.1	200	15	0
0.9	0.1	300	15	0
0.9	0.1	400	15	0
0.9	0.1	500	15	0

4.4.2 Класс данных 2

Класс данных 2 представляет собой матрицу стоимостей в диапозоне от 5000 до 9000 для 8 городов:

$$K_1 = \begin{pmatrix} 0 & 5466 & 8308 & 8068 & 7284 & 5635 & 6055 & 8129 \\ 5466 & 0 & 8205 & 7384 & 6794 & 6048 & 6174 & 6306 \\ 8308 & 8205 & 0 & 5485 & 7872 & 7981 & 7868 & 6912 \\ 8068 & 7384 & 5485 & 0 & 7002 & 6683 & 7544 & 8278 \\ 7284 & 6794 & 7872 & 7002 & 0 & 5159 & 8240 & 5663 \\ 5635 & 6048 & 7981 & 6683 & 5159 & 0 & 8801 & 8844 \\ 6055 & 6174 & 7868 & 7544 & 8240 & 8801 & 0 & 5493 \\ 8129 & 6306 & 6912 & 8278 & 5663 & 8844 & 5493 & 0 \end{pmatrix} \tag{4.2}$$

Для данного класса данных приведена таблица с выборкой параметров, которые наилучшим образом решают поставленную задачу.

Таблица 4.3 – Параметры для класса данных 2

α	ρ	days	ideal	mistake
0.1	0.7	300	47326	0
0.1	0.7	400	47326	0
0.1	0.7	500	47326	0
0.2	0.5	300	47326	0
0.2	0.5	400	47326	0
0.2	0.5	500	47326	0
0.3	0.3	300	47326	0
0.3	0.3	400	47326	0
0.3	0.3	500	47326	0
0.4	0.1	300	47326	0
0.4	0.1	400	47326	0
0.4	0.1	500	47326	0
0.5	0.6	300	47326	0

0.5	0.6	400	47326	0
0.5	0.6	500	47326	0
0.6	0.8	300	47326	0
0.6	0.8	400	47326	0
0.6	0.8	500	47326	0
0.7	0.2	300	47326	0
0.7	0.2	400	47326	0
0.7	0.2	500	47326	0
0.8	0.6	300	47326	0
0.8	0.6	400	47326	0
0.8	0.6	500	47326	0
0.9	0.9	300	47326	0
0.9	0.9	400	47326	0
0.9	0.9	500	47326	0

4.5 Вывод

В результате эксперимента было получено, что для 2 городов полный перебор работает быстрее муравьиного алгоритма в 32 раза. Начиная с 8 городов, муравьиный алгоритм работает быстрее полного перебора: в 75 раз быстрее для 10 городов. Таким образом, муравьиный алгоритм необходимо использовать при большом числе городов - от 8 и более.

Также в результате проведения параметризации было установлено, что для первого класса данных лучшие результаты муравьиный алгоритм дает на следующих значениях параметров:

- $\alpha = 0.1$, $\rho = 0.1\text{-}0.5, 0.7\text{-}0.9$;
- $\alpha = 0.2$, $\rho = 0.1\text{-}0.7, 0.9$;
- $\alpha = 0.3$, $\rho = 0.2\text{-}0.6, 0.9$;
- $\alpha = 0.4$, $\rho = 0.5\text{-}0.9$;

- $\alpha = 0.5$, $\rho = 0.1$, 0.5-0.9;
- $\alpha = 0.7$, $\rho = 0.4$-0.8.

Можно сделать вывод о том, что данные значения параметров следует использовать для матриц стоимостей в диапазоне от 1 до 5.

Для второго класса данных лучшие результаты муравьиный алгоритм дает на следующих значениях параметров:

- $\alpha = 0.1$, $\rho = 0.1$, 0.4, 0.7;
- $\alpha = 0.2$, $\rho = 0.3$, 0.7, 0.9;
- $\alpha = 0.3$, $\rho = 0.2$, 0.6, 0.9;
- $\alpha = 0.4$, $\rho = 0.7$, 0.8;
- $\alpha = 0.5$, $\rho = 0.2$, 0.6-0.9;
- $\alpha = 0.5$, $\rho = 0.3$, 0.7-0.9;
- $\alpha = 0.6$, $\rho = 0.2$, 0.6-0.9;
- $\alpha = 0.8$, $\rho = 0.1$, 0.6;
- $\alpha = 0.9$, $\rho = 0.9$.

Можно сделать вывод о том, что данные значения параметров следует использовать для матриц стоимостей в диапазоне от 5000 до 9000.

Из результатов параметризации видно, что погрешность результата уменьшается при большом числе дней и меньшем значении коэффициента видимости.

Заключение

В результате исследования было получено, что эвристический метод на базе муравьиного алгоритма следует использовать при большом количество городов - от 8 и более, так как для 2 городов мураьвиный алгоритм работает медленнее полного перебора в 32 раза, а для 10 городов - быстрее в 75 раз.

Лучшие значения муравьиный алгоритм показывает при меньших значениях коэффициента видимости и при большом числе дней.

Цель, поставленная перед началом работы, была достигнута. В ходе лабораторной работы были решены следующие задачи:

- были изучена задача коммивояжера и методы ее решения;
- были разработаны полный перебор и муравьиный алгоритм;
- был проведен сравнительный анализ реализованных алгоритмов;
- был подготовлен отчет о выполненной лабораторной работе.

Список литературы

[1] Что такое «задача коммивояжёра» [Электронный ресурс]. Режим доступа: https://thecode.media/komm/ (дата обращения: 5.12.2021).

[2] Эвристический алгоритм классификации и его нейросетевая интерпретация [Электронный ресурс]. Режим доступа: https://cyberleninka.ru/article/n/evristicheskiy-algoritm-klassifikatsii-i-ego-neyrosetevaya-interpre (дата обращения: 5.12.2021).

[3] Задача коммивояжера - метод ветвей и границ. Режим доступа: http://galyautdinov.ru/post/zadacha-kommivoyazhera (дата обращения: 5.12.2021).

[4] М.В. Ульянов Ресурсно-эффективные компютерные алгоритмы. Разработка и анализ. 2007.

[5] Welcome to Python [Электронный ресурс]. Режим доступа: https://www.python.org (дата обращения: 18.10.2021).

[6] time — Time access and conversions [Электронный ресурс]. Режим доступа: https://docs.python.org/3/library/time.html#functions (дата обращения: 18.10.2021).

[7] Ubuntu 20.04 LTS (Focal Fossa) Beta [Электронный ресурс]. Режим доступа: http://old-releases.ubuntu.com/releases/20.04.1/ (дата обращения: 18.10.2021).

[8] Мобильный процессор AMD Ryzen™ 3 3200U с графикой Radeon™ Vega 3 [Электронный ресурс]. Режим доступа: https://www.amd.com/ru/products/apu/amd-ryzen-3-3200u (дата обращения: 18.10.2021).

Приложение 1

Таблица 4.4 – Параметризация для класса данных 1

α	ρ	days	ideal	mistake
0.1	0.1	25	15	0
0.1	0.1	50	15	0
0.1	0.1	100	15	0
0.1	0.1	200	15	0
0.1	0.1	300	15	0
0.1	0.1	400	15	0
0.1	0.1	500	15	0
0.1	0.2	10	15	0
0.1	0.2	25	15	0
0.1	0.2	50	15	0
0.1	0.2	100	15	0
0.1	0.2	200	15	0
0.1	0.2	300	15	0
0.1	0.2	400	15	0
0.1	0.2	500	15	0
0.1	0.3	10	15	0
0.1	0.3	25	15	0
0.1	0.3	50	15	0
0.1	0.3	100	15	0
0.1	0.3	200	15	0
0.1	0.3	300	15	0
0.1	0.3	400	15	0
0.1	0.3	500	15	0
0.1	0.4	10	15	0
0.1	0.4	25	15	0
0.1	0.4	50	15	0
0.1	0.4	100	15	0
0.1	0.4	200	15	0

0.1	0.4	300	15	0
0.1	0.4	400	15	0
0.1	0.4	500	15	0
0.1	0.5	10	15	0
0.1	0.5	25	15	0
0.1	0.5	50	15	0
0.1	0.5	100	15	0
0.1	0.5	200	15	0
0.1	0.5	300	15	0
0.1	0.5	400	15	0
0.1	0.5	500	15	0
0.1	0.6	10	15	2
0.1	0.6	25	15	0
0.1	0.6	50	15	0
0.1	0.6	100	15	0
0.1	0.6	200	15	0
0.1	0.6	300	15	0
0.1	0.6	400	15	0
0.1	0.6	500	15	0
0.1	0.7	10	15	0
0.1	0.7	25	15	0
0.1	0.7	50	15	0
0.1	0.7	100	15	0
0.1	0.7	200	15	0
0.1	0.7	300	15	0
0.1	0.7	400	15	0
0.1	0.7	500	15	0
0.1	0.8	10	15	0
0.1	0.8	25	15	0
0.1	0.8	50	15	0
0.1	0.8	100	15	0
0.1	0.8	200	15	0
0.1	0.8	300	15	0

0.1	0.8	400	15	0
0.1	0.8	500	15	0
0.1	0.9	10	15	0
0.1	0.9	25	15	0
0.1	0.9	50	15	0
0.1	0.9	100	15	0
0.1	0.9	200	15	0
0.1	0.9	300	15	0
0.1	0.9	400	15	0
0.1	0.9	500	15	0
0.2	0.1	10	15	0
0.2	0.1	25	15	0
0.2	0.1	50	15	0
0.2	0.1	100	15	0
0.2	0.1	200	15	0
0.2	0.1	300	15	0
0.2	0.1	400	15	0
0.2	0.1	500	15	0
0.2	0.2	10	15	0
0.2	0.2	25	15	0
0.2	0.2	50	15	0
0.2	0.2	100	15	0
0.2	0.2	200	15	0
0.2	0.2	300	15	0
0.2	0.2	400	15	0
0.2	0.2	500	15	0
0.2	0.3	10	15	0
0.2	0.3	25	15	0
0.2	0.3	50	15	0
0.2	0.3	100	15	0
0.2	0.3	200	15	0
0.2	0.3	300	15	0
0.2	0.3	400	15	0

0.2	0.3	500	15	0
0.2	0.4	10	15	0
0.2	0.4	25	15	0
0.2	0.4	50	15	0
0.2	0.4	100	15	0
0.2	0.4	200	15	0
0.2	0.4	300	15	0
0.2	0.4	400	15	0
0.2	0.4	500	15	0
0.2	0.5	10	15	0
0.2	0.5	25	15	0
0.2	0.5	50	15	0
0.2	0.5	100	15	0
0.2	0.5	200	15	0
0.2	0.5	300	15	0
0.2	0.5	400	15	0
0.2	0.5	500	15	0
0.2	0.6	10	15	0
0.2	0.6	25	15	0
0.2	0.6	50	15	0
0.2	0.6	100	15	0
0.2	0.6	200	15	0
0.2	0.6	300	15	0
0.2	0.6	400	15	0
0.2	0.6	500	15	0
0.2	0.7	10	15	0
0.2	0.7	25	15	0
0.2	0.7	50	15	0
0.2	0.7	100	15	0
0.2	0.7	200	15	0
0.2	0.7	300	15	0
0.2	0.7	400	15	0
0.2	0.7	500	15	0

0.2	0.8	10	15	2
0.2	0.8	25	15	0
0.2	0.8	50	15	0
0.2	0.8	100	15	0
0.2	0.8	200	15	0
0.2	0.8	300	15	0
0.2	0.8	400	15	0
0.2	0.8	500	15	0
0.2	0.9	10	15	0
0.2	0.9	25	15	0
0.2	0.9	50	15	0
0.2	0.9	100	15	0
0.2	0.9	200	15	0
0.2	0.9	300	15	0
0.2	0.9	400	15	0
0.2	0.9	500	15	0
0.3	0.1	10	15	1
0.3	0.1	25	15	0
0.3	0.1	50	15	0
0.3	0.1	100	15	0
0.3	0.1	200	15	0
0.3	0.1	300	15	0
0.3	0.1	400	15	0
0.3	0.1	500	15	0
0.3	0.2	10	15	0
0.3	0.2	25	15	0
0.3	0.2	50	15	0
0.3	0.2	100	15	0
0.3	0.2	200	15	0
0.3	0.2	300	15	0
0.3	0.2	400	15	0
0.3	0.2	500	15	0
0.3	0.3	10	15	0

0.3	0.3	25	15	0
0.3	0.3	50	15	0
0.3	0.3	100	15	0
0.3	0.3	200	15	0
0.3	0.3	300	15	0
0.3	0.3	400	15	0
0.3	0.3	500	15	0
0.3	0.4	10	15	0
0.3	0.4	25	15	0
0.3	0.4	50	15	0
0.3	0.4	100	15	0
0.3	0.4	200	15	0
0.3	0.4	300	15	0
0.3	0.4	400	15	0
0.3	0.4	500	15	0
0.3	0.5	10	15	0
0.3	0.5	25	15	0
0.3	0.5	50	15	0
0.3	0.5	100	15	0
0.3	0.5	200	15	0
0.3	0.5	300	15	0
0.3	0.5	400	15	0
0.3	0.5	500	15	0
0.3	0.6	10	15	0
0.3	0.6	25	15	0
0.3	0.6	50	15	0
0.3	0.6	100	15	0
0.3	0.6	200	15	0
0.3	0.6	300	15	0
0.3	0.6	400	15	0
0.3	0.6	500	15	0
0.3	0.7	10	15	2
0.3	0.7	25	15	2

0.3	0.7	50	15	0
0.3	0.7	100	15	0
0.3	0.7	200	15	0
0.3	0.7	300	15	0
0.3	0.7	400	15	0
0.3	0.7	500	15	0
0.3	0.8	10	15	2
0.3	0.8	25	15	0
0.3	0.8	50	15	0
0.3	0.8	100	15	0
0.3	0.8	200	15	0
0.3	0.8	300	15	0
0.3	0.8	400	15	0
0.3	0.8	500	15	0
0.3	0.9	10	15	0
0.3	0.9	25	15	0
0.3	0.9	50	15	0
0.3	0.9	100	15	0
0.3	0.9	200	15	0
0.3	0.9	300	15	0
0.3	0.9	400	15	0
0.3	0.9	500	15	0
0.4	0.1	10	15	3
0.4	0.1	25	15	0
0.4	0.1	50	15	0
0.4	0.1	100	15	0
0.4	0.1	200	15	0
0.4	0.1	300	15	0
0.4	0.1	400	15	0
0.4	0.1	500	15	0
0.4	0.2	10	15	2
0.4	0.2	25	15	0
0.4	0.2	50	15	0

0.4	0.2	100	15	0
0.4	0.2	200	15	0
0.4	0.2	300	15	0
0.4	0.2	400	15	0
0.4	0.2	500	15	0
0.4	0.3	10	15	2
0.4	0.3	25	15	0
0.4	0.3	50	15	0
0.4	0.3	100	15	0
0.4	0.3	200	15	0
0.4	0.3	300	15	0
0.4	0.3	400	15	0
0.4	0.3	500	15	0
0.4	0.4	10	15	0
0.4	0.4	25	15	0
0.4	0.4	50	15	0
0.4	0.4	100	15	0
0.4	0.4	200	15	0
0.4	0.4	300	15	0
0.4	0.4	400	15	0
0.4	0.4	500	15	0
0.4	0.5	10	15	0
0.4	0.5	25	15	0
0.4	0.5	50	15	0
0.4	0.5	100	15	0
0.4	0.5	200	15	0
0.4	0.5	300	15	0
0.4	0.5	400	15	0
0.4	0.5	500	15	0
0.4	0.6	10	15	0
0.4	0.6	25	15	0
0.4	0.6	50	15	0
0.4	0.6	100	15	0

0.4	0.6	200	15	0
0.4	0.6	300	15	0
0.4	0.6	400	15	0
0.4	0.6	500	15	0
0.4	0.7	10	15	2
0.4	0.7	25	15	0
0.4	0.7	50	15	0
0.4	0.7	100	15	0
0.4	0.7	200	15	0
0.4	0.7	300	15	0
0.4	0.7	400	15	0
0.4	0.7	500	15	0
0.4	0.8	10	15	0
0.4	0.8	25	15	0
0.4	0.8	50	15	0
0.4	0.8	100	15	0
0.4	0.8	200	15	0
0.4	0.8	300	15	0
0.4	0.8	400	15	0
0.4	0.8	500	15	0
0.4	0.9	10	15	0
0.4	0.9	25	15	0
0.4	0.9	50	15	0
0.4	0.9	100	15	0
0.4	0.9	200	15	0
0.4	0.9	300	15	0
0.4	0.9	400	15	0
0.4	0.9	500	15	0
0.5	0.1	10	15	0
0.5	0.1	25	15	0
0.5	0.1	50	15	0
0.5	0.1	100	15	0
0.5	0.1	200	15	0

0.5	0.1	300	15	0
0.5	0.1	400	15	0
0.5	0.1	500	15	0
0.5	0.2	10	15	1
0.5	0.2	25	15	0
0.5	0.2	50	15	0
0.5	0.2	100	15	0
0.5	0.2	200	15	0
0.5	0.2	300	15	0
0.5	0.2	400	15	0
0.5	0.2	500	15	0
0.5	0.3	10	15	2
0.5	0.3	25	15	2
0.5	0.3	50	15	0
0.5	0.3	100	15	0
0.5	0.3	200	15	0
0.5	0.3	300	15	0
0.5	0.3	400	15	0
0.5	0.3	500	15	0
0.5	0.4	10	15	2
0.5	0.4	25	15	0
0.5	0.4	50	15	0
0.5	0.4	100	15	0
0.5	0.4	200	15	0
0.5	0.4	300	15	0
0.5	0.4	400	15	0
0.5	0.4	500	15	0
0.5	0.5	10	15	0
0.5	0.5	25	15	0
0.5	0.5	50	15	0
0.5	0.5	100	15	0
0.5	0.5	200	15	0
0.5	0.5	300	15	0

0.5	0.5	400	15	0
0.5	0.5	500	15	0
0.5	0.6	10	15	0
0.5	0.6	25	15	0
0.5	0.6	50	15	0
0.5	0.6	100	15	0
0.5	0.6	200	15	0
0.5	0.6	300	15	0
0.5	0.6	400	15	0
0.5	0.6	500	15	0
0.5	0.7	10	15	0
0.5	0.7	25	15	0
0.5	0.7	50	15	0
0.5	0.7	100	15	0
0.5	0.7	200	15	0
0.5	0.7	300	15	0
0.5	0.7	400	15	0
0.5	0.7	500	15	0
0.5	0.8	10	15	0
0.5	0.8	25	15	0
0.5	0.8	50	15	0
0.5	0.8	100	15	0
0.5	0.8	200	15	0
0.5	0.8	300	15	0
0.5	0.8	400	15	0
0.5	0.8	500	15	0
0.5	0.9	10	15	0
0.5	0.9	25	15	0
0.5	0.9	50	15	0
0.5	0.9	100	15	0
0.5	0.9	200	15	0
0.5	0.9	300	15	0
0.5	0.9	400	15	0

0.5	0.9	500	15	0
0.6	0.1	10	15	3
0.6	0.1	25	15	0
0.6	0.1	50	15	0
0.6	0.1	100	15	0
0.6	0.1	200	15	0
0.6	0.1	300	15	0
0.6	0.1	400	15	0
0.6	0.1	500	15	0
0.6	0.2	10	15	1
0.6	0.2	25	15	0
0.6	0.2	50	15	1
0.6	0.2	100	15	0
0.6	0.2	200	15	0
0.6	0.2	300	15	0
0.6	0.2	400	15	0
0.6	0.2	500	15	0
0.6	0.3	10	15	3
0.6	0.3	25	15	0
0.6	0.3	50	15	0
0.6	0.3	100	15	0
0.6	0.3	200	15	0
0.6	0.3	300	15	0
0.6	0.3	400	15	0
0.6	0.3	500	15	0
0.6	0.4	10	15	0
0.6	0.4	25	15	0
0.6	0.4	50	15	0
0.6	0.4	100	15	0
0.6	0.4	200	15	0
0.6	0.4	300	15	0
0.6	0.4	400	15	0
0.6	0.4	500	15	0

0.6	0.5	10	15	1
0.6	0.5	25	15	0
0.6	0.5	50	15	0
0.6	0.5	100	15	0
0.6	0.5	200	15	0
0.6	0.5	300	15	0
0.6	0.5	400	15	0
0.6	0.5	500	15	0
0.6	0.6	10	15	1
0.6	0.6	25	15	0
0.6	0.6	50	15	0
0.6	0.6	100	15	0
0.6	0.6	200	15	0
0.6	0.6	300	15	0
0.6	0.6	400	15	0
0.6	0.6	500	15	0
0.6	0.7	10	15	0
0.6	0.7	25	15	0
0.6	0.7	50	15	0
0.6	0.7	100	15	0
0.6	0.7	200	15	0
0.6	0.7	300	15	0
0.6	0.7	400	15	0
0.6	0.7	500	15	0
0.6	0.8	10	15	1
0.6	0.8	25	15	0
0.6	0.8	50	15	0
0.6	0.8	100	15	0
0.6	0.8	200	15	0
0.6	0.8	300	15	0
0.6	0.8	400	15	0
0.6	0.8	500	15	0
0.6	0.9	10	15	0

0.6	0.9	25	15	0
0.6	0.9	50	15	0
0.6	0.9	100	15	0
0.6	0.9	200	15	0
0.6	0.9	300	15	0
0.6	0.9	400	15	0
0.6	0.9	500	15	0
0.7	0.1	10	15	2
0.7	0.1	25	15	0
0.7	0.1	50	15	0
0.7	0.1	100	15	0
0.7	0.1	200	15	0
0.7	0.1	300	15	0
0.7	0.1	400	15	0
0.7	0.1	500	15	0
0.7	0.2	10	15	2
0.7	0.2	25	15	0
0.7	0.2	50	15	0
0.7	0.2	100	15	0
0.7	0.2	200	15	0
0.7	0.2	300	15	0
0.7	0.2	400	15	0
0.7	0.2	500	15	0
0.7	0.3	10	15	3
0.7	0.3	25	15	1
0.7	0.3	50	15	0
0.7	0.3	100	15	0
0.7	0.3	200	15	0
0.7	0.3	300	15	0
0.7	0.3	400	15	0
0.7	0.3	500	15	0
0.7	0.4	10	15	0
0.7	0.4	25	15	0

0.7	0.4	50	15	0
0.7	0.4	100	15	0
0.7	0.4	200	15	0
0.7	0.4	300	15	0
0.7	0.4	400	15	0
0.7	0.4	500	15	0
0.7	0.5	10	15	0
0.7	0.5	25	15	0
0.7	0.5	50	15	0
0.7	0.5	100	15	0
0.7	0.5	200	15	0
0.7	0.5	300	15	0
0.7	0.5	400	15	0
0.7	0.5	500	15	0
0.7	0.6	10	15	0
0.7	0.6	25	15	0
0.7	0.6	50	15	0
0.7	0.6	100	15	0
0.7	0.6	200	15	0
0.7	0.6	300	15	0
0.7	0.6	400	15	0
0.7	0.6	500	15	0
0.7	0.7	10	15	0
0.7	0.7	25	15	0
0.7	0.7	50	15	0
0.7	0.7	100	15	0
0.7	0.7	200	15	0
0.7	0.7	300	15	0
0.7	0.7	400	15	0
0.7	0.7	500	15	0
0.7	0.8	10	15	0
0.7	0.8	25	15	0
0.7	0.8	50	15	0

0.7	0.8	100	15	0
0.7	0.8	200	15	0
0.7	0.8	300	15	0
0.7	0.8	400	15	0
0.7	0.8	500	15	0
0.7	0.9	10	15	2
0.7	0.9	25	15	0
0.7	0.9	50	15	0
0.7	0.9	100	15	0
0.7	0.9	200	15	0
0.7	0.9	300	15	0
0.7	0.9	400	15	0
0.7	0.9	500	15	0
0.8	0.1	10	15	4
0.8	0.1	25	15	2
0.8	0.1	50	15	0
0.8	0.1	100	15	0
0.8	0.1	200	15	0
0.8	0.1	300	15	0
0.8	0.1	400	15	0
0.8	0.1	500	15	0
0.8	0.2	10	15	1
0.8	0.2	25	15	0
0.8	0.2	50	15	0
0.8	0.2	100	15	0
0.8	0.2	200	15	0
0.8	0.2	300	15	0
0.8	0.2	400	15	0
0.8	0.2	500	15	0
0.8	0.3	10	15	2
0.8	0.3	25	15	0
0.8	0.3	50	15	0
0.8	0.3	100	15	0

0.8	0.3	200	15	0
0.8	0.3	300	15	0
0.8	0.3	400	15	0
0.8	0.3	500	15	0
0.8	0.4	10	15	3
0.8	0.4	25	15	1
0.8	0.4	50	15	0
0.8	0.4	100	15	0
0.8	0.4	200	15	0
0.8	0.4	300	15	0
0.8	0.4	400	15	0
0.8	0.4	500	15	0
0.8	0.5	10	15	4
0.8	0.5	25	15	1
0.8	0.5	50	15	0
0.8	0.5	100	15	0
0.8	0.5	200	15	0
0.8	0.5	300	15	0
0.8	0.5	400	15	0
0.8	0.5	500	15	0
0.8	0.6	10	15	1
0.8	0.6	25	15	0
0.8	0.6	50	15	0
0.8	0.6	100	15	0
0.8	0.6	200	15	0
0.8	0.6	300	15	0
0.8	0.6	400	15	0
0.8	0.6	500	15	0
0.8	0.7	10	15	0
0.8	0.7	25	15	1
0.8	0.7	50	15	0
0.8	0.7	100	15	0
0.8	0.7	200	15	0

0.8	0.7	300	15	0
0.8	0.7	400	15	0
0.8	0.7	500	15	0
0.8	0.8	10	15	0
0.8	0.8	25	15	1
0.8	0.8	50	15	0
0.8	0.8	100	15	0
0.8	0.8	200	15	0
0.8	0.8	300	15	0
0.8	0.8	400	15	0
0.8	0.8	500	15	0
0.8	0.9	10	15	0
0.8	0.9	25	15	0
0.8	0.9	50	15	1
0.8	0.9	100	15	0
0.8	0.9	200	15	0
0.8	0.9	300	15	0
0.8	0.9	400	15	0
0.8	0.9	500	15	0
0.9	0.1	10	15	5
0.9	0.1	25	15	2
0.9	0.1	50	15	0
0.9	0.1	100	15	0
0.9	0.1	200	15	0
0.9	0.1	300	15	0
0.9	0.1	400	15	0
0.9	0.1	500	15	0
0.9	0.2	10	15	1
0.9	0.2	25	15	1
0.9	0.2	50	15	1
0.9	0.2	100	15	0
0.9	0.2	200	15	0
0.9	0.2	300	15	0

0.9	0.2	400	15	0
0.9	0.2	500	15	0
0.9	0.3	10	15	2
0.9	0.3	25	15	0
0.9	0.3	50	15	0
0.9	0.3	100	15	0
0.9	0.3	200	15	0
0.9	0.3	300	15	0
0.9	0.3	400	15	0
0.9	0.3	500	15	0
0.9	0.4	10	15	2
0.9	0.4	25	15	2
0.9	0.4	50	15	0
0.9	0.4	100	15	0
0.9	0.4	200	15	0
0.9	0.4	300	15	0
0.9	0.4	400	15	0
0.9	0.4	500	15	0
0.9	0.5	10	15	3
0.9	0.5	25	15	0
0.9	0.5	50	15	0
0.9	0.5	100	15	0
0.9	0.5	200	15	0
0.9	0.5	300	15	0
0.9	0.5	400	15	0
0.9	0.5	500	15	0
0.9	0.6	10	15	2
0.9	0.6	25	15	0
0.9	0.6	50	15	0
0.9	0.6	100	15	0
0.9	0.6	200	15	0
0.9	0.6	300	15	0
0.9	0.6	400	15	0

0.9	0.6	500	15	0
0.9	0.7	10	15	2
0.9	0.7	25	15	2
0.9	0.7	50	15	1
0.9	0.7	100	15	0
0.9	0.7	200	15	0
0.9	0.7	300	15	0
0.9	0.7	400	15	0
0.9	0.7	500	15	0
0.9	0.8	10	15	0
0.9	0.8	25	15	0
0.9	0.8	50	15	1
0.9	0.8	100	15	0
0.9	0.8	200	15	0
0.9	0.8	300	15	0
0.9	0.8	400	15	0
0.9	0.8	500	15	0
0.9	0.9	10	15	2
0.9	0.9	25	15	2
0.9	0.9	50	15	0
0.9	0.9	100	15	0
0.9	0.9	200	15	0
0.9	0.9	300	15	0
0.9	0.9	400	15	0
0.9	0.9	500	15	0

Приложение 2

Таблица 4.5 – Параметризация для класса данных 2

α	ρ	days	ideal	mistake
0.1	0.1	25	47326	1141
0.1	0.1	50	47326	0
0.1	0.1	100	47326	0
0.1	0.1	200	47326	0
0.1	0.1	300	47326	0
0.1	0.1	400	47326	0
0.1	0.1	500	47326	0
0.1	0.2	10	47326	1573
0.1	0.2	25	47326	712
0.1	0.2	50	47326	294
0.1	0.2	100	47326	0
0.1	0.2	200	47326	712
0.1	0.2	300	47326	0
0.1	0.2	400	47326	0
0.1	0.2	500	47326	0
0.1	0.3	10	47326	2957
0.1	0.3	25	47326	294
0.1	0.3	50	47326	294
0.1	0.3	100	47326	294
0.1	0.3	200	47326	0
0.1	0.3	300	47326	0
0.1	0.3	400	47326	0
0.1	0.3	500	47326	0
0.1	0.4	10	47326	721
0.1	0.4	25	47326	721
0.1	0.4	50	47326	0
0.1	0.4	100	47326	0
0.1	0.4	200	47326	0

0.1	0.4	300	47326	0
0.1	0.4	400	47326	0
0.1	0.4	500	47326	0
0.1	0.5	10	47326	2418
0.1	0.5	25	47326	1360
0.1	0.5	50	47326	0
0.1	0.5	100	47326	294
0.1	0.5	200	47326	0
0.1	0.5	300	47326	0
0.1	0.5	400	47326	0
0.1	0.5	500	47326	0
0.1	0.6	10	47326	1975
0.1	0.6	25	47326	883
0.1	0.6	50	47326	0
0.1	0.6	100	47326	0
0.1	0.6	200	47326	712
0.1	0.6	300	47326	0
0.1	0.6	400	47326	0
0.1	0.6	500	47326	0
0.1	0.7	10	47326	1330
0.1	0.7	25	47326	294
0.1	0.7	50	47326	294
0.1	0.7	100	47326	0
0.1	0.7	200	47326	0
0.1	0.7	300	47326	0
0.1	0.7	400	47326	0
0.1	0.7	500	47326	0
0.1	0.8	10	47326	721
0.1	0.8	25	47326	294
0.1	0.8	50	47326	294
0.1	0.8	100	47326	294
0.1	0.8	200	47326	0
0.1	0.8	300	47326	294

0.1	0.8	400	47326	0
0.1	0.8	500	47326	294
0.1	0.9	10	47326	1360
0.1	0.9	25	47326	712
0.1	0.9	50	47326	0
0.1	0.9	100	47326	827
0.1	0.9	200	47326	0
0.1	0.9	300	47326	0
0.1	0.9	400	47326	0
0.1	0.9	500	47326	0
0.2	0.1	10	47326	1145
0.2	0.1	25	47326	1145
0.2	0.1	50	47326	0
0.2	0.1	100	47326	712
0.2	0.1	200	47326	0
0.2	0.1	300	47326	0
0.2	0.1	400	47326	0
0.2	0.1	500	47326	0
0.2	0.2	10	47326	1105
0.2	0.2	25	47326	1088
0.2	0.2	50	47326	1088
0.2	0.2	100	47326	294
0.2	0.2	200	47326	0
0.2	0.2	300	47326	0
0.2	0.2	400	47326	0
0.2	0.2	500	47326	0
0.2	0.3	10	47326	1360
0.2	0.3	25	47326	1088
0.2	0.3	50	47326	0
0.2	0.3	100	47326	0
0.2	0.3	200	47326	0
0.2	0.3	300	47326	0
0.2	0.3	400	47326	0

0.2	0.3	500	47326	0
0.2	0.4	10	47326	1591
0.2	0.4	25	47326	712
0.2	0.4	50	47326	827
0.2	0.4	100	47326	712
0.2	0.4	200	47326	0
0.2	0.4	300	47326	0
0.2	0.4	400	47326	294
0.2	0.4	500	47326	0
0.2	0.5	10	47326	2269
0.2	0.5	25	47326	0
0.2	0.5	50	47326	0
0.2	0.5	100	47326	721
0.2	0.5	200	47326	0
0.2	0.5	300	47326	0
0.2	0.5	400	47326	0
0.2	0.5	500	47326	0
0.2	0.6	10	47326	1145
0.2	0.6	25	47326	1359
0.2	0.6	50	47326	294
0.2	0.6	100	47326	294
0.2	0.6	200	47326	0
0.2	0.6	300	47326	294
0.2	0.6	400	47326	0
0.2	0.6	500	47326	0
0.2	0.7	10	47326	1591
0.2	0.7	25	47326	1088
0.2	0.7	50	47326	712
0.2	0.7	100	47326	0
0.2	0.7	200	47326	0
0.2	0.7	300	47326	0
0.2	0.7	400	47326	0
0.2	0.7	500	47326	0

0.2	0.8	10	47326	712
0.2	0.8	25	47326	1088
0.2	0.8	50	47326	721
0.2	0.8	100	47326	294
0.2	0.8	200	47326	712
0.2	0.8	300	47326	0
0.2	0.8	400	47326	0
0.2	0.8	500	47326	0
0.2	0.9	10	47326	1591
0.2	0.9	25	47326	2229
0.2	0.9	50	47326	294
0.2	0.9	100	47326	0
0.2	0.9	200	47326	0
0.2	0.9	300	47326	0
0.2	0.9	400	47326	0
0.2	0.9	500	47326	0
0.3	0.1	10	47326	883
0.3	0.1	25	47326	1088
0.3	0.1	50	47326	294
0.3	0.1	100	47326	294
0.3	0.1	200	47326	0
0.3	0.1	300	47326	0
0.3	0.1	400	47326	0
0.3	0.1	500	47326	0
0.3	0.2	10	47326	1105
0.3	0.2	25	47326	0
0.3	0.2	50	47326	827
0.3	0.2	100	47326	0
0.3	0.2	200	47326	0
0.3	0.2	300	47326	0
0.3	0.2	400	47326	0
0.3	0.2	500	47326	0
0.3	0.3	10	47326	712

0.3	0.3	25	47326	1634
0.3	0.3	50	47326	721
0.3	0.3	100	47326	294
0.3	0.3	200	47326	0
0.3	0.3	300	47326	0
0.3	0.3	400	47326	0
0.3	0.3	500	47326	0
0.3	0.4	10	47326	1840
0.3	0.4	25	47326	1105
0.3	0.4	50	47326	0
0.3	0.4	100	47326	712
0.3	0.4	200	47326	0
0.3	0.4	300	47326	0
0.3	0.4	400	47326	0
0.3	0.4	500	47326	0
0.3	0.5	10	47326	1439
0.3	0.5	25	47326	883
0.3	0.5	50	47326	0
0.3	0.5	100	47326	712
0.3	0.5	200	47326	0
0.3	0.5	300	47326	0
0.3	0.5	400	47326	0
0.3	0.5	500	47326	0
0.3	0.6	10	47326	1006
0.3	0.6	25	47326	1359
0.3	0.6	50	47326	721
0.3	0.6	100	47326	0
0.3	0.6	200	47326	0
0.3	0.6	300	47326	0
0.3	0.6	400	47326	0
0.3	0.6	500	47326	0
0.3	0.7	10	47326	2098
0.3	0.7	25	47326	1141

0.3	0.7	50	47326	827
0.3	0.7	100	47326	0
0.3	0.7	200	47326	294
0.3	0.7	300	47326	0
0.3	0.7	400	47326	0
0.3	0.7	500	47326	0
0.3	0.8	10	47326	1360
0.3	0.8	25	47326	1359
0.3	0.8	50	47326	1141
0.3	0.8	100	47326	294
0.3	0.8	200	47326	0
0.3	0.8	300	47326	0
0.3	0.8	400	47326	0
0.3	0.8	500	47326	0
0.3	0.9	10	47326	2402
0.3	0.9	25	47326	294
0.3	0.9	50	47326	712
0.3	0.9	100	47326	0
0.3	0.9	200	47326	0
0.3	0.9	300	47326	0
0.3	0.9	400	47326	0
0.3	0.9	500	47326	0
0.4	0.1	10	47326	2260
0.4	0.1	25	47326	1088
0.4	0.1	50	47326	827
0.4	0.1	100	47326	0
0.4	0.1	200	47326	0
0.4	0.1	300	47326	0
0.4	0.1	400	47326	0
0.4	0.1	500	47326	0
0.4	0.2	10	47326	2229
0.4	0.2	25	47326	294
0.4	0.2	50	47326	827

0.4	0.2	100	47326	294
0.4	0.2	200	47326	0
0.4	0.2	300	47326	294
0.4	0.2	400	47326	0
0.4	0.2	500	47326	0
0.4	0.3	10	47326	294
0.4	0.3	25	47326	1439
0.4	0.3	50	47326	0
0.4	0.3	100	47326	0
0.4	0.3	200	47326	0
0.4	0.3	300	47326	294
0.4	0.3	400	47326	0
0.4	0.3	500	47326	0
0.4	0.4	10	47326	3728
0.4	0.4	25	47326	0
0.4	0.4	50	47326	1105
0.4	0.4	100	47326	294
0.4	0.4	200	47326	0
0.4	0.4	300	47326	0
0.4	0.4	400	47326	0
0.4	0.4	500	47326	0
0.4	0.5	10	47326	2229
0.4	0.5	25	47326	827
0.4	0.5	50	47326	712
0.4	0.5	100	47326	0
0.4	0.5	200	47326	712
0.4	0.5	300	47326	0
0.4	0.5	400	47326	0
0.4	0.5	500	47326	0
0.4	0.6	10	47326	1145
0.4	0.6	25	47326	0
0.4	0.6	50	47326	712
0.4	0.6	100	47326	0

0.4	0.6	200	47326	0
0.4	0.6	300	47326	0
0.4	0.6	400	47326	0
0.4	0.6	500	47326	0
0.4	0.7	10	47326	1840
0.4	0.7	25	47326	294
0.4	0.7	50	47326	0
0.4	0.7	100	47326	0
0.4	0.7	200	47326	0
0.4	0.7	300	47326	0
0.4	0.7	400	47326	0
0.4	0.7	500	47326	0
0.4	0.8	10	47326	1634
0.4	0.8	25	47326	1088
0.4	0.8	50	47326	0
0.4	0.8	100	47326	0
0.4	0.8	200	47326	0
0.4	0.8	300	47326	0
0.4	0.8	400	47326	0
0.4	0.8	500	47326	0
0.4	0.9	10	47326	827
0.4	0.9	25	47326	827
0.4	0.9	50	47326	721
0.4	0.9	100	47326	0
0.4	0.9	200	47326	0
0.4	0.9	300	47326	0
0.4	0.9	400	47326	0
0.4	0.9	500	47326	0
0.5	0.1	10	47326	294
0.5	0.1	25	47326	0
0.5	0.1	50	47326	712
0.5	0.1	100	47326	294
0.5	0.1	200	47326	0

0.5	0.1	300	47326	0
0.5	0.1	400	47326	0
0.5	0.1	500	47326	0
0.5	0.2	10	47326	2556
0.5	0.2	25	47326	0
0.5	0.2	50	47326	0
0.5	0.2	100	47326	0
0.5	0.2	200	47326	0
0.5	0.2	300	47326	0
0.5	0.2	400	47326	0
0.5	0.2	500	47326	0
0.5	0.3	10	47326	294
0.5	0.3	25	47326	1761
0.5	0.3	50	47326	294
0.5	0.3	100	47326	294
0.5	0.3	200	47326	0
0.5	0.3	300	47326	0
0.5	0.3	400	47326	0
0.5	0.3	500	47326	0
0.5	0.4	10	47326	721
0.5	0.4	25	47326	0
0.5	0.4	50	47326	294
0.5	0.4	100	47326	0
0.5	0.4	200	47326	294
0.5	0.4	300	47326	0
0.5	0.4	400	47326	0
0.5	0.4	500	47326	0
0.5	0.5	10	47326	294
0.5	0.5	25	47326	1105
0.5	0.5	50	47326	721
0.5	0.5	100	47326	294
0.5	0.5	200	47326	294
0.5	0.5	300	47326	0

0.5	0.5	400	47326	0
0.5	0.5	500	47326	0
0.5	0.6	10	47326	883
0.5	0.6	25	47326	1088
0.5	0.6	50	47326	712
0.5	0.6	100	47326	0
0.5	0.6	200	47326	0
0.5	0.6	300	47326	0
0.5	0.6	400	47326	0
0.5	0.6	500	47326	0
0.5	0.7	10	47326	2402
0.5	0.7	25	47326	294
0.5	0.7	50	47326	883
0.5	0.7	100	47326	0
0.5	0.7	200	47326	0
0.5	0.7	300	47326	0
0.5	0.7	400	47326	0
0.5	0.7	500	47326	0
0.5	0.8	10	47326	1402
0.5	0.8	25	47326	0
0.5	0.8	50	47326	712
0.5	0.8	100	47326	0
0.5	0.8	200	47326	0
0.5	0.8	300	47326	0
0.5	0.8	400	47326	0
0.5	0.8	500	47326	0
0.5	0.9	10	47326	1088
0.5	0.9	25	47326	712
0.5	0.9	50	47326	827
0.5	0.9	100	47326	0
0.5	0.9	200	47326	0
0.5	0.9	300	47326	0
0.5	0.9	400	47326	0

0.5	0.9	500	47326	0
0.6	0.1	10	47326	2418
0.6	0.1	25	47326	1402
0.6	0.1	50	47326	294
0.6	0.1	100	47326	0
0.6	0.1	200	47326	0
0.6	0.1	300	47326	0
0.6	0.1	400	47326	0
0.6	0.1	500	47326	0
0.6	0.2	10	47326	2238
0.6	0.2	25	47326	827
0.6	0.2	50	47326	1330
0.6	0.2	100	47326	712
0.6	0.2	200	47326	0
0.6	0.2	300	47326	294
0.6	0.2	400	47326	0
0.6	0.2	500	47326	0
0.6	0.3	10	47326	1105
0.6	0.3	25	47326	827
0.6	0.3	50	47326	883
0.6	0.3	100	47326	0
0.6	0.3	200	47326	0
0.6	0.3	300	47326	0
0.6	0.3	400	47326	0
0.6	0.3	500	47326	0
0.6	0.4	10	47326	1591
0.6	0.4	25	47326	1006
0.6	0.4	50	47326	0
0.6	0.4	100	47326	0
0.6	0.4	200	47326	0
0.6	0.4	300	47326	294
0.6	0.4	400	47326	0
0.6	0.4	500	47326	0

0.6	0.5	10	47326	294
0.6	0.5	25	47326	1006
0.6	0.5	50	47326	0
0.6	0.5	100	47326	712
0.6	0.5	200	47326	294
0.6	0.5	300	47326	0
0.6	0.5	400	47326	0
0.6	0.5	500	47326	0
0.6	0.6	10	47326	2063
0.6	0.6	25	47326	1006
0.6	0.6	50	47326	294
0.6	0.6	100	47326	0
0.6	0.6	200	47326	721
0.6	0.6	300	47326	0
0.6	0.6	400	47326	0
0.6	0.6	500	47326	0
0.6	0.7	10	47326	712
0.6	0.7	25	47326	1105
0.6	0.7	50	47326	712
0.6	0.7	100	47326	0
0.6	0.7	200	47326	0
0.6	0.7	300	47326	0
0.6	0.7	400	47326	0
0.6	0.7	500	47326	0
0.6	0.8	10	47326	721
0.6	0.8	25	47326	1145
0.6	0.8	50	47326	721
0.6	0.8	100	47326	0
0.6	0.8	200	47326	0
0.6	0.8	300	47326	0
0.6	0.8	400	47326	0
0.6	0.8	500	47326	0
0.6	0.9	10	47326	2726

0.6	0.9	25	47326	1840
0.6	0.9	50	47326	0
0.6	0.9	100	47326	0
0.6	0.9	200	47326	0
0.6	0.9	300	47326	0
0.6	0.9	400	47326	0
0.6	0.9	500	47326	0
0.7	0.1	10	47326	4282
0.7	0.1	25	47326	1088
0.7	0.1	50	47326	1141
0.7	0.1	100	47326	712
0.7	0.1	200	47326	0
0.7	0.1	300	47326	0
0.7	0.1	400	47326	0
0.7	0.1	500	47326	0
0.7	0.2	10	47326	2482
0.7	0.2	25	47326	0
0.7	0.2	50	47326	1145
0.7	0.2	100	47326	712
0.7	0.2	200	47326	721
0.7	0.2	300	47326	0
0.7	0.2	400	47326	0
0.7	0.2	500	47326	0
0.7	0.3	10	47326	1359
0.7	0.3	25	47326	712
0.7	0.3	50	47326	294
0.7	0.3	100	47326	712
0.7	0.3	200	47326	294
0.7	0.3	300	47326	712
0.7	0.3	400	47326	0
0.7	0.3	500	47326	0
0.7	0.4	10	47326	1359
0.7	0.4	25	47326	0

0.7	0.4	50	47326	294
0.7	0.4	100	47326	294
0.7	0.4	200	47326	0
0.7	0.4	300	47326	827
0.7	0.4	400	47326	0
0.7	0.4	500	47326	0
0.7	0.5	10	47326	2309
0.7	0.5	25	47326	1105
0.7	0.5	50	47326	883
0.7	0.5	100	47326	294
0.7	0.5	200	47326	0
0.7	0.5	300	47326	0
0.7	0.5	400	47326	0
0.7	0.5	500	47326	0
0.7	0.6	10	47326	294
0.7	0.6	25	47326	1359
0.7	0.6	50	47326	294
0.7	0.6	100	47326	294
0.7	0.6	200	47326	0
0.7	0.6	300	47326	0
0.7	0.6	400	47326	0
0.7	0.6	500	47326	0
0.7	0.7	10	47326	2014
0.7	0.7	25	47326	1006
0.7	0.7	50	47326	712
0.7	0.7	100	47326	0
0.7	0.7	200	47326	712
0.7	0.7	300	47326	0
0.7	0.7	400	47326	0
0.7	0.7	500	47326	0
0.7	0.8	10	47326	1840
0.7	0.8	25	47326	1359
0.7	0.8	50	47326	883

0.7	0.8	100	47326	0
0.7	0.8	200	47326	721
0.7	0.8	300	47326	0
0.7	0.8	400	47326	294
0.7	0.8	500	47326	0
0.7	0.9	10	47326	1006
0.7	0.9	25	47326	1591
0.7	0.9	50	47326	0
0.7	0.9	100	47326	0
0.7	0.9	200	47326	0
0.7	0.9	300	47326	294
0.7	0.9	400	47326	0
0.7	0.9	500	47326	0
0.8	0.1	10	47326	0
0.8	0.1	25	47326	1330
0.8	0.1	50	47326	1105
0.8	0.1	100	47326	0
0.8	0.1	200	47326	0
0.8	0.1	300	47326	0
0.8	0.1	400	47326	0
0.8	0.1	500	47326	0
0.8	0.2	10	47326	2269
0.8	0.2	25	47326	0
0.8	0.2	50	47326	294
0.8	0.2	100	47326	712
0.8	0.2	200	47326	0
0.8	0.2	300	47326	0
0.8	0.2	400	47326	0
0.8	0.2	500	47326	294
0.8	0.3	10	47326	1006
0.8	0.3	25	47326	2418
0.8	0.3	50	47326	1088
0.8	0.3	100	47326	294

0.8	0.3	200	47326	0
0.8	0.3	300	47326	0
0.8	0.3	400	47326	0
0.8	0.3	500	47326	0
0.8	0.4	10	47326	294
0.8	0.4	25	47326	1591
0.8	0.4	50	47326	2163
0.8	0.4	100	47326	294
0.8	0.4	200	47326	0
0.8	0.4	300	47326	294
0.8	0.4	400	47326	0
0.8	0.4	500	47326	0
0.8	0.5	10	47326	1330
0.8	0.5	25	47326	2229
0.8	0.5	50	47326	827
0.8	0.5	100	47326	712
0.8	0.5	200	47326	0
0.8	0.5	300	47326	0
0.8	0.5	400	47326	0
0.8	0.5	500	47326	0
0.8	0.6	10	47326	1867
0.8	0.6	25	47326	294
0.8	0.6	50	47326	721
0.8	0.6	100	47326	0
0.8	0.6	200	47326	0
0.8	0.6	300	47326	0
0.8	0.6	400	47326	0
0.8	0.6	500	47326	0
0.8	0.7	10	47326	1359
0.8	0.7	25	47326	1105
0.8	0.7	50	47326	1006
0.8	0.7	100	47326	294
0.8	0.7	200	47326	0

0.8	0.7	300	47326	0
0.8	0.7	400	47326	0
0.8	0.7	500	47326	0
0.8	0.8	10	47326	712
0.8	0.8	25	47326	827
0.8	0.8	50	47326	1141
0.8	0.8	100	47326	294
0.8	0.8	200	47326	0
0.8	0.8	300	47326	0
0.8	0.8	400	47326	712
0.8	0.8	500	47326	294
0.8	0.9	10	47326	1867
0.8	0.9	25	47326	1324
0.8	0.9	50	47326	721
0.8	0.9	100	47326	827
0.8	0.9	200	47326	294
0.8	0.9	300	47326	0
0.8	0.9	400	47326	0
0.8	0.9	500	47326	0
0.9	0.1	10	47326	1761
0.9	0.1	25	47326	712
0.9	0.1	50	47326	721
0.9	0.1	100	47326	0
0.9	0.1	200	47326	0
0.9	0.1	300	47326	0
0.9	0.1	400	47326	0
0.9	0.1	500	47326	0
0.9	0.2	10	47326	2547
0.9	0.2	25	47326	1360
0.9	0.2	50	47326	1105
0.9	0.2	100	47326	294
0.9	0.2	200	47326	294
0.9	0.2	300	47326	0

0.9	0.2	400	47326	294
0.9	0.2	500	47326	0
0.9	0.3	10	47326	2735
0.9	0.3	25	47326	712
0.9	0.3	50	47326	1141
0.9	0.3	100	47326	294
0.9	0.3	200	47326	294
0.9	0.3	300	47326	0
0.9	0.3	400	47326	0
0.9	0.3	500	47326	0
0.9	0.4	10	47326	2664
0.9	0.4	25	47326	2707
0.9	0.4	50	47326	294
0.9	0.4	100	47326	0
0.9	0.4	200	47326	712
0.9	0.4	300	47326	0
0.9	0.4	400	47326	0
0.9	0.4	500	47326	0
0.9	0.5	10	47326	721
0.9	0.5	25	47326	0
0.9	0.5	50	47326	1360
0.9	0.5	100	47326	0
0.9	0.5	200	47326	0
0.9	0.5	300	47326	0
0.9	0.5	400	47326	0
0.9	0.5	500	47326	294
0.9	0.6	10	47326	294
0.9	0.6	25	47326	0
0.9	0.6	50	47326	0
0.9	0.6	100	47326	712
0.9	0.6	200	47326	0
0.9	0.6	300	47326	0
0.9	0.6	400	47326	0

0.9	0.6	500	47326	0
0.9	0.7	10	47326	2238
0.9	0.7	25	47326	0
0.9	0.7	50	47326	712
0.9	0.7	100	47326	721
0.9	0.7	200	47326	0
0.9	0.7	300	47326	0
0.9	0.7	400	47326	0
0.9	0.7	500	47326	0
0.9	0.8	10	47326	2309
0.9	0.8	25	47326	2957
0.9	0.8	50	47326	0
0.9	0.8	100	47326	294
0.9	0.8	200	47326	0
0.9	0.8	300	47326	0
0.9	0.8	400	47326	0
0.9	0.8	500	47326	0
0.9	0.9	10	47326	1105
0.9	0.9	25	47326	3134
0.9	0.9	50	47326	0
0.9	0.9	100	47326	0
0.9	0.9	200	47326	0
0.9	0.9	300	47326	0
0.9	0.9	400	47326	0
0.9	0.9	500	47326	0