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 курсу «Анализ алгоритмов»

на тему: «Задача коммивояжера»

Студент ИУ7-54Б (Группа) _____ (Подпись, дата) Булдаков М. (И. О. Фамилия)

Преподаватель _____ (Подпись, дата) Волкова Л. Л. (И. О. Фамилия)

2023 г.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Задача коммивояжера является одной из классических задач комбинаторной оптимизации, привлекающей внимание исследователей и практиков в области логистики, транспорта и информационных технологий. В контексте логистики, эта задача применяется для оптимизации маршрутов доставки товаров и грузов, что позволяет сократить затраты на перевозку, уменьшить время доставки и улучшить эффективность работы логистических компаний [1].

Цель данной лабораторной работы — рассмотреть алгоритмы решения задачи коммивояжера в случае построения карты перемещений для воздухоплавателей. Для достижения поставленной цели необходимо выполнить следующие задачи:

— описать алгоритмы решения задачи коммивояжера;

— спроектировать программное обеспечение, реализующее алгоритмы решения задачи коммивояжера;

— выбрать инструменты для реализации и замера процессорного времени выполнения реализаций решения задачи;

— проанализировать затраты реализаций алгоритмов по времени.

1 Аналитический раздел

В данном разделе будут описаны алгоритмы решения задачи коммивояжера.

1.1 Описание задачи

Задача коммивояжера — одна из самых известных задач комбинаторной оптимизации, заключающаяся в поиске наиболее выгодного маршрута, проходящего через указанные города и возвращающегося в начальный пункт.

В общем случае задача формулируется следующим образом: имеется N городов, для каждой пары которых известно расстояние между ними. Требуется найти такой маршрут, проходящий через каждый город по одному разу (и возвращающийся в исходный город), при этом сумма всех расстояний на этом маршруте должна быть наименьшей из возможных [1].

В данной работе будет рассматриваться вариант задачи с незамкнутым маршрутом, т. е. без одного последнего перехода. Стоимость пути между городами будет отличаться в различных направлениях, поэтому граф будет ориентированным.

1.2 Алгоритмы решения задачи

Алгоритм полного перебора

Задача может быть решена перебором всех вариантов объезда и выбором оптимального. Очевидно, что при полном переборе будет найден самый кратчайший маршрут, но при этом для перебора необходимо будет выполнить порядка $O(N!)$ операций, где N — количество городов, что является тяжелой задачей даже для современных ЭВМ при N порядка сотни.

Муравьиный алгоритм (без элитных муравьев)

Муравьиный алгоритм — алгоритм решения задачи коммивояжера, основанный на принципе поведения колонии муравьев [2].

Муравьи действуют, руководствуясь органами чувств. Каждый муравей оставляет на своем пути феромоны, чтобы другие могли ориентироваться. При большом количестве муравьев наибольшее количество феромона остается на наиболее посещаемом пути, посещаемость же может быть связана с длинами ребер. Муравьи используют непрямой обмен информацией через окружающую

среду посредством феромона.

Основная идея заключается в том, что выделяются две фазы: день и ночь. В фазу дня каждый муравей k строит один маршрут, вечером обновляется лучшая траектория. В фазу ночи обновляется матрица феромона.

Каждый муравей имеет 3 способности:

1) Зрение — муравей k, стоя в городе i, может оценить привлекательность ребра i — j;

2) Обоняние — муравей чует концентрацию феромона $\tau_{ij}(t)$ на ребре i—j в текущий день t;

3) Память — муравей запоминает список, посещенных за текущий день t городов — $J_k(t)$.

Привлекательность ребра i—j оценивается по формуле (1.1).

$$\eta_{ij} = \frac{1}{D_{ij}}, \tag{1.1}$$

где D_{ij} — метка ребра, D — матрица смежности.

Стоя в городе i, муравей k выбирает следующий город на основе вероятностного правила (1.2).

$$p_{ij,k} = \begin{cases} 0, j \in J_k, \\ \frac{\eta_{ij}^{\alpha} \cdot \tau_{ij}^{\beta}(t)}{\sum_{q \in J_k} \eta_{iq}^{\alpha} \cdot \tau_{iq}^{\beta}(t)}, \text{иначе}, \end{cases} \tag{1.2}$$

где α — коэффициент жадности, β — коэффициент стадности, причем $\alpha + \beta = 1$.

После завершения движения всех муравьев (ночью, перед наступлением следующего дня), феромон обновляется по формуле (1.3).

$$\tau_{ij}(t+1) = \tau_{ij}(t) \cdot (1 - \rho) + \Delta\tau_{ij}(t), \tag{1.3}$$

где $\rho \in [0, 1]$ — коэффициент испарения феромона.

$$\Delta\tau_{ij}(t) = \sum_{k=1}^{N} \Delta\tau_{ij,k}(t). \tag{1.4}$$

$$\Delta\tau_{ij}^{k}(t) = \begin{cases} 0, \text{муравей k в день t не ходил по ребру i—j}, \\ Q/L_k, \text{иначе}, \end{cases} \tag{1.5}$$

где Q — квота феромона одного муравья на день, Q выбирается соразмерной длине лучшего маршрута в графе.

Чтобы значение феромона не обнулилось и не повлекло обнуление вероятности перехода по ребру, после расчета значения $\tau_{ij}(t+1)$ необходимо выполнять проверку матрицы феромона и все значения, которые меньше заданного порога, заменить на порог [2].

В данном алгоритме отсутствует полный перебор, что означает меньшую трудоемкость, чем для алгоритма полного перебора, но при этом лучшее решение не гарантируется.

2 Конструкторский раздел

В этом разделе будет представлено описание используемых типов данных, а также схематические изображения алгоритмов решения задачи коммивояжера.

2.1 Требования к программному обеспечению

Программа должна поддерживать два режима работы: режим массового замера времени и режим решения задачи коммивояжера.

Режим массового замера времени должен обладать следующей функциональностью:

— генерировать графы различного размер для проведения замеров;

— осуществлять массовый замер, используя сгенерированные данные;

— результаты массового замера должны быть представлены в виде таблицы и графика.

К режиму решения задачи коммивояжера выдвигается следующий ряд требований:

— возможность работать графами, записанными в файл;

— возможность вводить матрицы смежности графов;

— наличие интерфейса для выбора действий;

— на выходе программы, стоимость и маршрут кратчайшей длины.

2.2 Описание используемых типов данных

При реализации алгоритмов будут использованы следующие структуры и типы данных:

— граф — множество вершин и ребер между ними, задается с помощью матрицы смежности;

— матрица — двумерный массив чисел.

2.3 Разработка алгоритмов

На рисунке 2.1 представлена схема алгоритма решения задачи коммивояжера полным перебором. На рисунке 2.2 представлена схема муравьиного алгоритма.

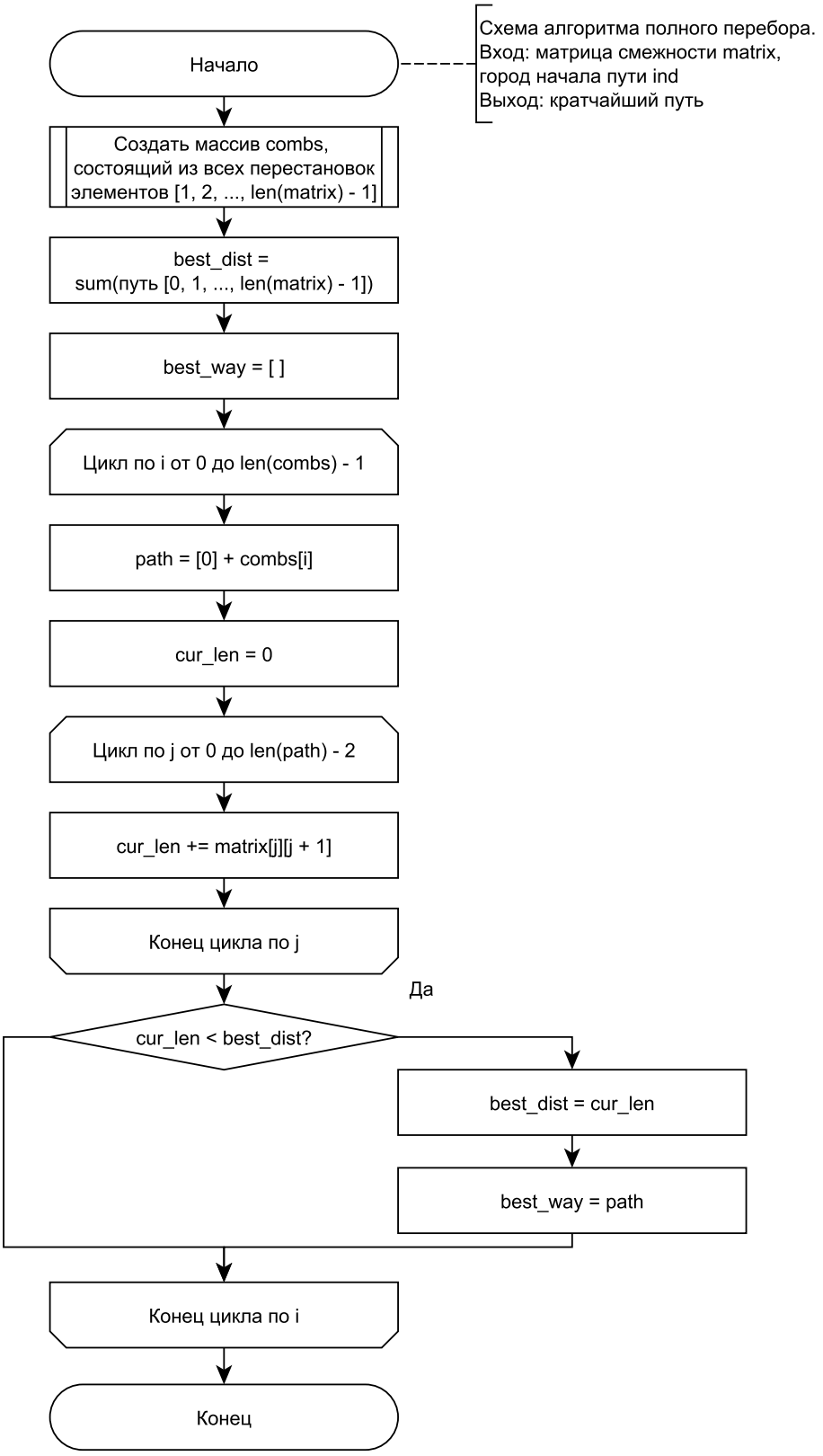

Рисунок 2.1 – Схема алгоритма решения задачи коммивояжера полным перебором

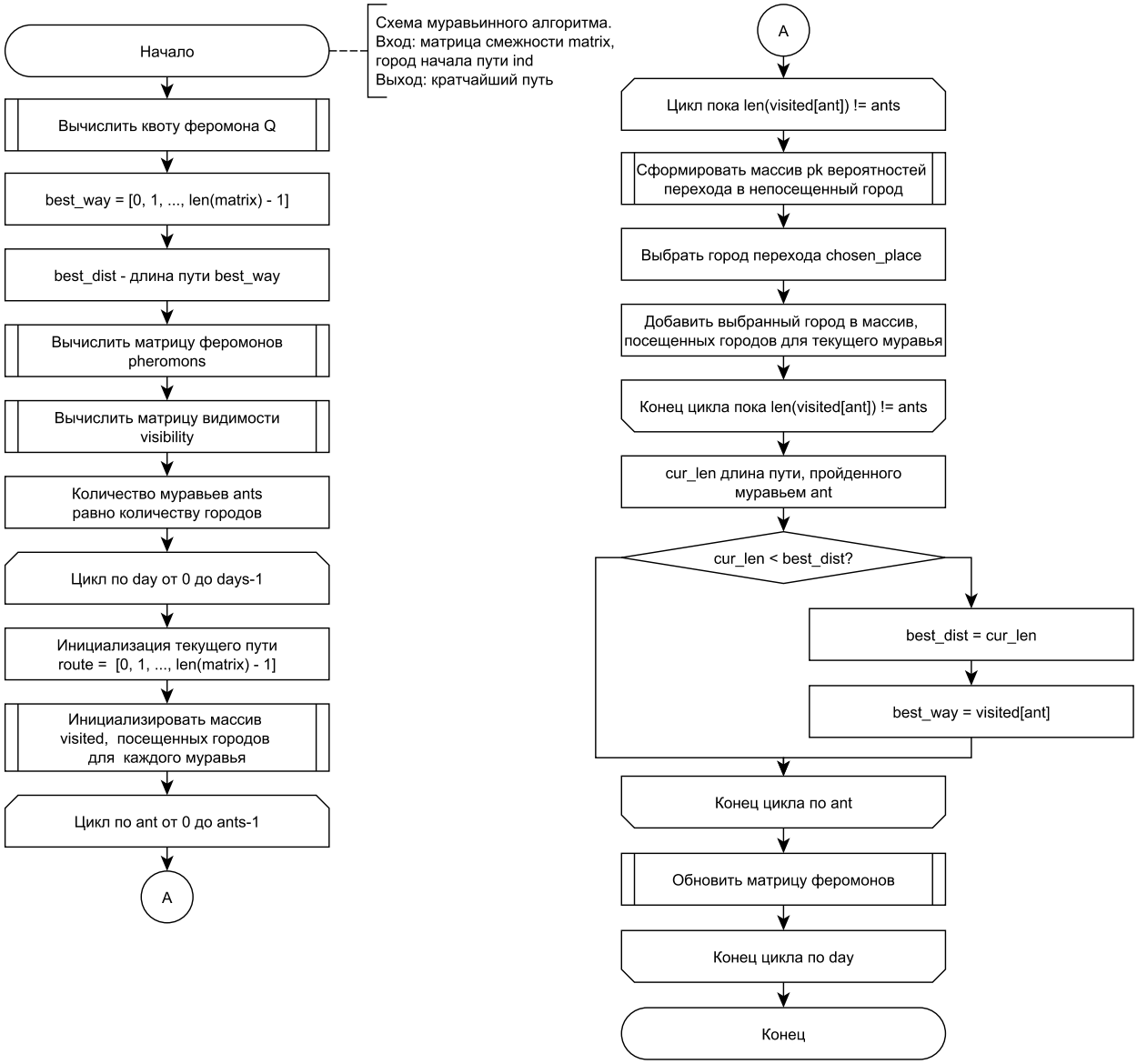

Рисунок 2.2 – Схема муравьиного алгоритма

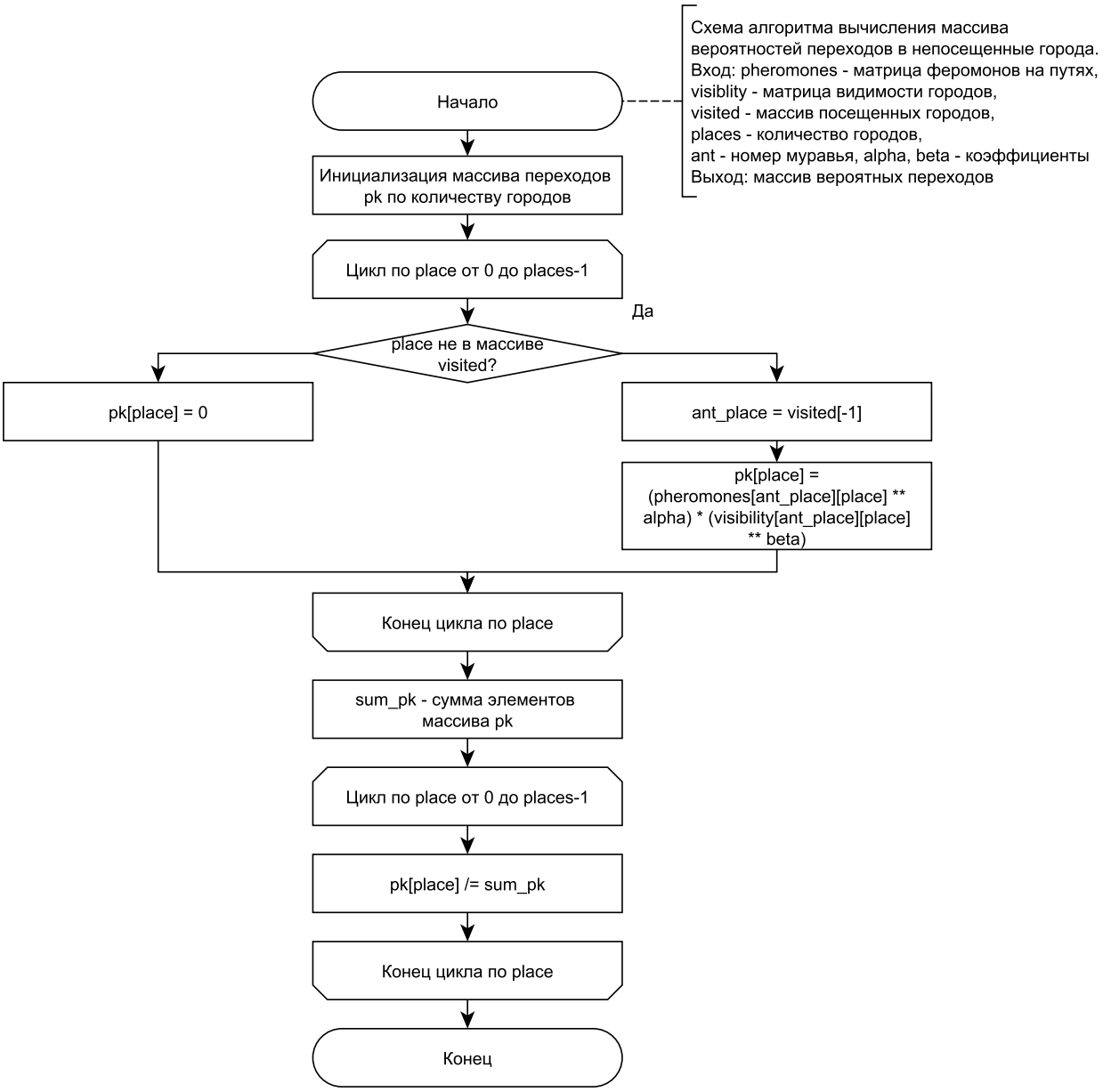

Рисунок 2.3 – Схема алгоритма вычисления массива вероятностей переходов в непосещенные города

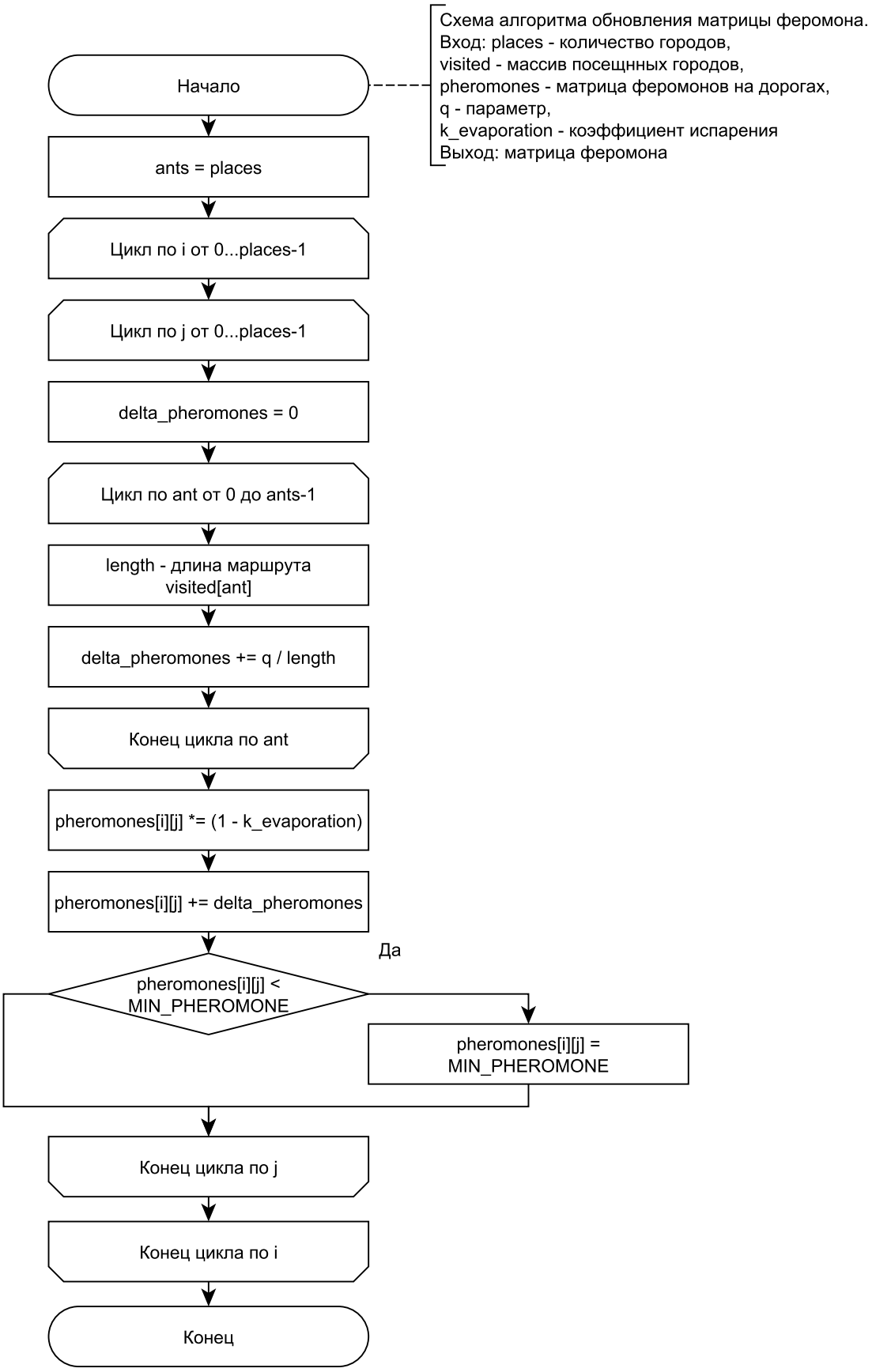

Рисунок 2.4 – Схема алгоритма обновления матрицы феромона

3 Технологический раздел

В данном разделе будут приведены требования к программному обеспечению, средства реализации, листинг кода.

3.1 Средства реализации

Для реализации данной работы был выбран язык *Python* [3]. Данный выбор обусловлен следующим:

— язык поддерживает все структуры данных, которые выбраны в результате проектирования;

— язык позволяет реализовать все алгоритмы, выбранные в результате проектирования;

— язык позволяет замерять процессорное время с помощью модуля *time*.

Процессорное время было замерено с помощью функции *process_time()* из модуля *time* [4].

3.2 Сведения о модулях программы

Данная программа разбита на следующие модули:

— *main.py* — файл, содержащий функцию *main*;

— *algorithms.py* — файл, содержащий код реализаций алгоритмов решения задачи коммивояжера;

— *utils.py* — содержит вспомогательные функции работы с графами;

— *compare_time.py* — файл, в котором содержатся функции для замера и вывода времени выполнения реализаций алгоритмов.

3.3 Реализация алгоритмов

В листинге 3.1 приведена реализация алгоритма решения задачи полным перебором. В листинге 3.2 приведена реализация муравьиного алгоритма. В листингах 3.3 – 3.5 приведены реализации вспомогательных подпрограмм.

Листинг 3.1 – Функция решения задачи комивояжера полным перебором

```
def brut(matrix, start_city):
    size = len(matrix)
    nodes = list(range(0, size))
    nodes.pop(start_city)

    min_dist = float("inf")
    best_way = []

    for comb in it.permutations(nodes):
        comb = [start_city] + list(comb)
        cur_dist = 0
        for i in range(len(comb) - 1):
            cur_dist += matrix[comb[i]][comb[i + 1]]

        if cur_dist < min_dist:
            min_dist = cur_dist
            best_way = comb

    return min_dist, best_way
```

Листинг 3.2 – Функция решения задачи комивояжера муравьиным алгоритмом

```
def ants(matrix, alpha, beta, k_evaporation, days, start_city):
    places = len(matrix)

    q = get_q(matrix)
    best_way = []
    min_dist = float("inf")
    pheromones = [[1 for i in range(places)] for j in
       range(places)]
    visibility = [
        [(1.0 / matrix[i][j] if (i != j) else 0) for j in
           range(len(matrix[i]))]
        for i in range(len(matrix))
    ]

    ants = places
    for day in range(days):
        route = np.arange(places)
        visited = [[start_city] for _ in range(ants)]
        for ant in range(ants):
            while len(visited[ant]) != ants:
                pk = find_ways(
                    pheromones, visibility, visited, places,
                       ant, alpha, beta
                )
                chosen_place = choose_place(pk)
                visited[ant].append(chosen_place - 1)

            cur_length = get_length(matrix, visited[ant])

            if cur_length < min_dist:
                min_dist = cur_length
                best_way = visited[ant]

        pheromones = update_pheromones(
            matrix, places, visited, pheromones, q, k_evaporation
        )

    return min_dist, best_way
```

Листинг 3.3 – Функция вычисления массива вероятностей переходов в непосещенные города

```
def find_ways(pheromones, visibility, visited, places, ant,
   alpha, beta):
    pk = [0] * places

    for place in range(places):
        if place not in visited[ant]:
            ant_place = visited[ant][-1]
            pk[place] = pow(pheromones[ant_place][place], alpha)
               * pow(
                visibility[ant_place][place], beta
            )
        else:
            pk[place] = 0

    sum_pk = sum(pk)

    for place in range(places):
        pk[place] /= sum_pk

    return pk
```

Листинг 3.4 – Функция обновляющая матрицу феромонов

```
MIN_PHEROMONE = 0.01


def update_pheromones(matrix, places, visited, pheromones, q,
    k_evaporation):
    ants = places

    for i in range(places):
        for j in range(places):
            delta = 0
            for ant in range(ants):
                length = get_length(matrix, visited[ant])
                delta += q / length

            pheromones[i][j] *= 1 - k_evaporation
            pheromones[i][j] += delta
            if pheromones[i][j] < MIN_PHEROMONE:
                pheromones[i][j] = MIN_PHEROMONE

    return pheromones
```

Листинг 3.5 – Вспомогательные функции

```
def choose_place(pk):
    posibility = random()
    choice = 0
    chosen_place = 0
    while (choice < posibility) and (chosen_place < len(pk)):
        choice += pk[chosen_place]
        chosen_place += 1

    return chosen_place


def get_q(matrix):
    q = 0
    count = 0
    for i in range(len(matrix)):
        for j in range(len(matrix[i])):
            if i != j:
                q += matrix[i][j]
                count += 1
    return q / count


def get_length(matrix, route):
    length = 0

    for way_len in range(1, len(route)):
        length += matrix[route[way_len - 1]][route[way_len]]

    return length
```

Вывод

Были разработаны спроектированные алгоритмы решения задачи комивояжера.

4 Исследовательский раздел

В данном разделе будут приведены: пример работы программы, постановка эксперимента и сравнительный анализ алгоритмов на основе полученных данных.

4.1 Демонстрация работы программы

На рисунке 4.1 представлена демонстрация работы разработанного программного обеспечения, а именно показаны результаты решения задачи коммивояжера для графа, заданного матрицей смежности $\begin{pmatrix} 0 & 6 & 1 & 9 & 9 \\ 1 & 0 & 6 & 2 & 7 \\ 1 & 4 & 0 & 6 & 5 \\ 8 & 5 & 5 & 0 & 8 \\ 5 & 1 & 4 & 8 & 0 \end{pmatrix}$.

```
          1 - Решить задачу для введенной матрицы полным перебором.
          2 - Решить задачу для введенной матрицы муравьиным алгоритмом.
          3 - Выполнить решение заготовленных матриц.
          4 - Провести замерный эксперимент.

          0 - Выйти.


Выберите пункт меню: 1
Введит количество городов: 5
Вводите матрицу
0 6 1 9 9
1 0 6 2 7
1 4 0 6 5
8 5 5 0 8
5 1 4 8 0
Введите начальный город: 0
Кратчайший путь [0, 2, 4, 1, 3] длинной 9

          1 - Решить задачу для введенной матрицы полным перебором.
          2 - Решить задачу для введенной матрицы муравьиным алгоритмом.
          3 - Выполнить решение заготовленных матриц.
          4 - Провести замерный эксперимент.

          0 - Выйти.


Выберите пункт меню: 2
Введит количество городов: 5
Вводите матрицу
0 6 1 9 9
1 0 6 2 7
1 4 0 6 5
8 5 5 0 8
5 1 4 8 0
Введите начальный город: 0
Введите количество дней: 10
Кратчайший путь [0, 2, 4, 1, 3] длинной 9
```

Рисунок 4.1 – Демонстрация работы программы при решении задачи коммивояжера

4.2 Технические характеристики

Технические характеристики устройства, на котором выполнялись замеры по времени, следующие:

— процессор: AMD Ryzen 5 4600H 3 ГГц [5];

— оперативная память: 16 ГБайт;

— операционная система: Windows 10 Pro 64-разрядная система версии 22H2 [6].

При замерах времени ноутбук был включен в сеть электропитания и был нагружен только системными приложениями.

4.3 Время выполнения реализаций алгоритмов

Результаты замеров времени выполнения реализаций алгоритмов решения задачи коммивояжера приведены в таблице 4.1. Замеры времени проводились на полносвязных графах одного размера и усреднялись для каждого набора одинаковых экспериментов. Для реализации муравьиного алгоритма количество дней равно 10.

Таблица 4.1 – Время работы реализации алгоритмов решения задачи коммивояжера (в с)

Количество городов	Полный перебор	Муравьиный
4	$1.563 \cdot 10^{-5}$	$1.422 \cdot 10^{-3}$
5	$4.688 \cdot 10^{-5}$	$3.125 \cdot 10^{-3}$
6	$2.344 \cdot 10^{-4}$	$5.750 \cdot 10^{-3}$
7	$1.594 \cdot 10^{-3}$	$1.005 \cdot 10^{-2}$
8	$1.231 \cdot 10^{-2}$	$1.664 \cdot 10^{-2}$
9	$1.105 \cdot 10^{-1}$	$2.506 \cdot 10^{-2}$

4.4 Параметризация муравьиного алгоритма

Автоматическая параметризация была проведена на двух классах данных — 4.4.1 и 4.4.2. Алгоритм будет запущен для набора значений $\alpha, \rho \in (0, 1)$ и $t_{max} \in \{5, 25, 50, 100\}$.

Итоговая таблица значений параметризации будет состоять из следующих колонок:

— α — коэффициент жадности;

— ρ — коэффициент испарения;

— t_{max} — количество дней жизни колонии муравьев;

— *Результат* — максимальная длина пути, полученная муравьиным алгоритмом за 10 запусков;

— *Ошибка* — разность между эталонным значением и результатом.

4.4.1 Класс данных 1

Класс данных 1 представляет собой матрицу (4.1) смежности размером 10 элементов со значениями от 1 до 100.

$$M_1 = \begin{pmatrix} 0 & 34 & 84 & 64 & 37 & 51 & 7 & 55 & 28 & 10 \\ 34 & 0 & 89 & 61 & 26 & 64 & 31 & 82 & 19 & 48 \\ 11 & 16 & 0 & 46 & 2 & 75 & 48 & 65 & 45 & 62 \\ 12 & 30 & 46 & 0 & 71 & 37 & 27 & 70 & 44 & 25 \\ 68 & 20 & 31 & 36 & 0 & 47 & 44 & 72 & 29 & 82 \\ 90 & 78 & 11 & 44 & 91 & 0 & 62 & 43 & 73 & 77 \\ 90 & 33 & 80 & 8 & 98 & 48 & 0 & 99 & 36 & 71 \\ 18 & 16 & 28 & 22 & 99 & 62 & 80 & 0 & 31 & 63 \\ 51 & 77 & 45 & 91 & 45 & 41 & 77 & 40 & 0 & 26 \\ 55 & 67 & 24 & 8 & 57 & 29 & 82 & 50 & 78 & 0 \end{pmatrix} \tag{4.1}$$

Результаты параметризации для первого класса данных содержатся в приложении А.

4.4.2 Класс данных 2

Класс данных 1 представляет собой матрицу (4.2) смежности размером 10 элементов со значениями от 100 до 10000.

$$M_2 = \begin{pmatrix} 0 & 2807 & 2494 & 4820 & 8257 & 8688 & 2784 & 7073 & 5246 & 5816 \\ 6688 & 0 & 3579 & 5313 & 8630 & 1530 & 6084 & 6745 & 7040 & 9483 \\ 3070 & 1462 & 0 & 5040 & 8379 & 1145 & 9036 & 4213 & 8686 & 9060 \\ 7869 & 6450 & 5517 & 0 & 2936 & 3890 & 8459 & 1535 & 264 & 6446 \\ 7588 & 1538 & 4715 & 957 & 0 & 3324 & 8420 & 705 & 780 & 2114 \\ 5313 & 7012 & 6748 & 3221 & 8283 & 0 & 738 & 2073 & 1966 & 9134 \\ 3297 & 2578 & 8005 & 2220 & 4799 & 8012 & 0 & 5366 & 6696 & 1371 \\ 2885 & 6139 & 3617 & 5044 & 4018 & 1991 & 5720 & 0 & 3252 & 3749 \\ 6359 & 5162 & 2316 & 2605 & 3085 & 8213 & 8597 & 8510 & 0 & 9288 \\ 4739 & 7850 & 3077 & 222 & 3818 & 8252 & 8204 & 8576 & 4921 & 0 \end{pmatrix} \quad (4.2)$$

Результаты параметризации для второго класса данных содержатся в приложении Б.

ПРИЛОЖЕНИЕ А

Таблица А.1 – Параметризация муравьиного алгоритма для первого класса данных

α	ρ	t_{max}	Результат	Ошибка
0.1	0.1	5	239	78
0.1	0.1	25	198	37
0.1	0.1	50	180	19
0.1	0.1	100	167	6
0.1	0.2	5	241	80
0.1	0.2	25	195	34
0.1	0.2	50	189	28
0.1	0.2	100	180	19
0.1	0.3	5	215	54
0.1	0.3	25	198	37
0.1	0.3	50	176	15
0.1	0.3	100	161	0
0.1	0.4	5	246	85
0.1	0.4	25	198	37
0.1	0.4	50	180	19
0.1	0.4	100	179	18
0.1	0.5	5	226	65
0.1	0.5	25	190	29
0.1	0.5	50	184	23
0.1	0.5	100	176	15
0.1	0.6	5	241	80
0.1	0.6	25	198	37
0.1	0.6	50	185	24
0.1	0.6	100	167	6
0.1	0.7	5	225	64
0.1	0.7	25	195	34
0.1	0.7	50	205	44

Таблица А.1 – Параметризация муравьиного алгоритма для первого класса данных

α	ρ	t_{max}	Результат	Ошибка
0.1	0.7	100	167	6
0.1	0.8	5	227	66
0.1	0.8	25	189	28
0.1	0.8	50	180	19
0.1	0.8	100	185	24
0.1	0.9	5	228	67
0.1	0.9	25	200	39
0.1	0.9	50	180	19
0.1	0.9	100	180	19
0.2	0.1	5	239	78
0.2	0.1	25	218	57
0.2	0.1	50	195	34
0.2	0.1	100	184	23
0.2	0.2	5	237	76
0.2	0.2	25	209	48
0.2	0.2	50	180	19
0.2	0.2	100	180	19
0.2	0.3	5	243	82
0.2	0.3	25	204	43
0.2	0.3	50	197	36
0.2	0.3	100	180	19
0.2	0.4	5	237	76
0.2	0.4	25	197	36
0.2	0.4	50	188	27
0.2	0.4	100	179	18
0.2	0.5	5	215	54
0.2	0.5	25	200	39
0.2	0.5	50	180	19
0.2	0.5	100	185	24
0.2	0.6	5	248	87

Таблица А.1 – Параметризация муравьиного алгоритма для первого класса данных

α	ρ	t_{max}	Результат	Ошибка
0.2	0.6	25	195	34
0.2	0.6	50	188	27
0.2	0.6	100	161	0
0.2	0.7	5	264	103
0.2	0.7	25	221	60
0.2	0.7	50	194	33
0.2	0.7	100	180	19
0.2	0.8	5	242	81
0.2	0.8	25	203	42
0.2	0.8	50	191	30
0.2	0.8	100	193	32
0.2	0.9	5	231	70
0.2	0.9	25	195	34
0.2	0.9	50	184	23
0.2	0.9	100	181	20
0.3	0.1	5	225	64
0.3	0.1	25	211	50
0.3	0.1	50	208	47
0.3	0.1	100	180	19
0.3	0.2	5	247	86
0.3	0.2	25	207	46
0.3	0.2	50	192	31
0.3	0.2	100	167	6
0.3	0.3	5	246	85
0.3	0.3	25	209	48
0.3	0.3	50	193	32
0.3	0.3	100	191	30
0.3	0.4	5	260	99
0.3	0.4	25	191	30
0.3	0.4	50	186	25

Таблица А.1 – Параметризация муравьиного алгоритма для первого класса данных

α	ρ	t_{max}	Результат	Ошибка
0.3	0.4	100	191	30
0.3	0.5	5	241	80
0.3	0.5	25	231	70
0.3	0.5	50	193	32
0.3	0.5	100	185	24
0.3	0.6	5	239	78
0.3	0.6	25	207	46
0.3	0.6	50	207	46
0.3	0.6	100	182	21
0.3	0.7	5	255	94
0.3	0.7	25	206	45
0.3	0.7	50	191	30
0.3	0.7	100	189	28
0.3	0.8	5	238	77
0.3	0.8	25	206	45
0.3	0.8	50	203	42
0.3	0.8	100	184	23
0.3	0.9	5	250	89
0.3	0.9	25	206	45
0.3	0.9	50	204	43
0.3	0.9	100	180	19
0.4	0.1	5	266	105
0.4	0.1	25	232	71
0.4	0.1	50	207	46
0.4	0.1	100	191	30
0.4	0.2	5	265	104
0.4	0.2	25	236	75
0.4	0.2	50	195	34
0.4	0.2	100	191	30
0.4	0.3	5	259	98

Таблица А.1 – Параметризация муравьиного алгоритма для первого класса данных

α	ρ	t_{max}	Результат	Ошибка
0.4	0.3	25	226	65
0.4	0.3	50	198	37
0.4	0.3	100	205	44
0.4	0.4	5	272	111
0.4	0.4	25	212	51
0.4	0.4	50	203	42
0.4	0.4	100	194	33
0.4	0.5	5	289	128
0.4	0.5	25	231	70
0.4	0.5	50	206	45
0.4	0.5	100	193	32
0.4	0.6	5	239	78
0.4	0.6	25	223	62
0.4	0.6	50	200	39
0.4	0.6	100	192	31
0.4	0.7	5	259	98
0.4	0.7	25	214	53
0.4	0.7	50	209	48
0.4	0.7	100	189	28
0.4	0.8	5	255	94
0.4	0.8	25	218	57
0.4	0.8	50	211	50
0.4	0.8	100	193	32
0.4	0.9	5	265	104
0.4	0.9	25	209	48
0.4	0.9	50	207	46
0.4	0.9	100	198	37
0.5	0.1	5	271	110
0.5	0.1	25	207	46
0.5	0.1	50	241	80

Таблица А.1 – Параметризация муравьиного алгоритма для первого класса данных

α	ρ	t_{max}	Результат	Ошибка
0.5	0.1	100	210	49
0.5	0.2	5	272	111
0.5	0.2	25	229	68
0.5	0.2	50	227	66
0.5	0.2	100	204	43
0.5	0.3	5	256	95
0.5	0.3	25	227	66
0.5	0.3	50	216	55
0.5	0.3	100	195	34
0.5	0.4	5	284	123
0.5	0.4	25	223	62
0.5	0.4	50	211	50
0.5	0.4	100	204	43
0.5	0.5	5	286	125
0.5	0.5	25	210	49
0.5	0.5	50	215	54
0.5	0.5	100	198	37
0.5	0.6	5	248	87
0.5	0.6	25	213	52
0.5	0.6	50	229	68
0.5	0.6	100	191	30
0.5	0.7	5	251	90
0.5	0.7	25	223	62
0.5	0.7	50	204	43
0.5	0.7	100	194	33
0.5	0.8	5	271	110
0.5	0.8	25	228	67
0.5	0.8	50	207	46
0.5	0.8	100	205	44
0.5	0.9	5	270	109

Таблица А.1 – Параметризация муравьиного алгоритма для первого класса данных

α	ρ	t_{max}	Результат	Ошибка
0.5	0.9	25	221	60
0.5	0.9	50	230	69
0.5	0.9	100	185	24
0.6	0.1	5	296	135
0.6	0.1	25	253	92
0.6	0.1	50	214	53
0.6	0.1	100	203	42
0.6	0.2	5	306	145
0.6	0.2	25	245	84
0.6	0.2	50	215	54
0.6	0.2	100	206	45
0.6	0.3	5	292	131
0.6	0.3	25	245	84
0.6	0.3	50	220	59
0.6	0.3	100	219	58
0.6	0.4	5	270	109
0.6	0.4	25	248	87
0.6	0.4	50	225	64
0.6	0.4	100	209	48
0.6	0.5	5	291	130
0.6	0.5	25	249	88
0.6	0.5	50	221	60
0.6	0.5	100	213	52
0.6	0.6	5	277	116
0.6	0.6	25	243	82
0.6	0.6	50	224	63
0.6	0.6	100	207	46
0.6	0.7	5	280	119
0.6	0.7	25	227	66
0.6	0.7	50	221	60

Таблица А.1 – Параметризация муравьиного алгоритма для первого класса данных

α	ρ	t_{max}	Результат	Ошибка
0.6	0.7	100	211	50
0.6	0.8	5	272	111
0.6	0.8	25	237	76
0.6	0.8	50	238	77
0.6	0.8	100	204	43
0.6	0.9	5	288	127
0.6	0.9	25	226	65
0.6	0.9	50	221	60
0.6	0.9	100	226	65
0.7	0.1	5	324	163
0.7	0.1	25	243	82
0.7	0.1	50	234	73
0.7	0.1	100	204	43
0.7	0.2	5	305	144
0.7	0.2	25	256	95
0.7	0.2	50	237	76
0.7	0.2	100	207	46
0.7	0.3	5	292	131
0.7	0.3	25	241	80
0.7	0.3	50	219	58
0.7	0.3	100	214	53
0.7	0.4	5	303	142
0.7	0.4	25	240	79
0.7	0.4	50	212	51
0.7	0.4	100	207	46
0.7	0.5	5	286	125
0.7	0.5	25	229	68
0.7	0.5	50	253	92
0.7	0.5	100	221	60
0.7	0.6	5	309	148

Таблица А.1 – Параметризация муравьиного алгоритма для первого класса данных

α	ρ	t_{max}	Результат	Ошибка
0.7	0.6	25	248	87
0.7	0.6	50	241	80
0.7	0.6	100	219	58
0.7	0.7	5	285	124
0.7	0.7	25	250	89
0.7	0.7	50	233	72
0.7	0.7	100	223	62
0.7	0.8	5	289	128
0.7	0.8	25	253	92
0.7	0.8	50	243	82
0.7	0.8	100	225	64
0.7	0.9	5	286	125
0.7	0.9	25	253	92
0.7	0.9	50	240	79
0.7	0.9	100	218	57
0.8	0.1	5	303	142
0.8	0.1	25	273	112
0.8	0.1	50	235	74
0.8	0.1	100	226	65
0.8	0.2	5	336	175
0.8	0.2	25	276	115
0.8	0.2	50	243	82
0.8	0.2	100	231	70
0.8	0.3	5	325	164
0.8	0.3	25	240	79
0.8	0.3	50	243	82
0.8	0.3	100	241	80
0.8	0.4	5	307	146
0.8	0.4	25	268	107
0.8	0.4	50	243	82

Таблица А.1 – Параметризация муравьиного алгоритма для первого класса данных

α	ρ	t_{max}	Результат	Ошибка
0.8	0.4	100	231	70
0.8	0.5	5	291	130
0.8	0.5	25	256	95
0.8	0.5	50	238	77
0.8	0.5	100	239	78
0.8	0.6	5	300	139
0.8	0.6	25	270	109
0.8	0.6	50	239	78
0.8	0.6	100	225	64
0.8	0.7	5	295	134
0.8	0.7	25	253	92
0.8	0.7	50	249	88
0.8	0.7	100	227	66
0.8	0.8	5	318	157
0.8	0.8	25	259	98
0.8	0.8	50	223	62
0.8	0.8	100	225	64
0.8	0.9	5	302	141
0.8	0.9	25	259	98
0.8	0.9	50	242	81
0.8	0.9	100	231	70
0.9	0.1	5	311	150
0.9	0.1	25	271	110
0.9	0.1	50	253	92
0.9	0.1	100	238	77
0.9	0.2	5	317	156
0.9	0.2	25	259	98
0.9	0.2	50	269	108
0.9	0.2	100	247	86
0.9	0.3	5	331	170

Таблица А.1 – Параметризация муравьиного алгоритма для первого класса данных

α	ρ	t_{max}	Результат	Ошибка
0.9	0.3	25	258	97
0.9	0.3	50	238	77
0.9	0.3	100	240	79
0.9	0.4	5	324	163
0.9	0.4	25	269	108
0.9	0.4	50	283	122
0.9	0.4	100	253	92
0.9	0.5	5	341	180
0.9	0.5	25	254	93
0.9	0.5	50	241	80
0.9	0.5	100	252	91
0.9	0.6	5	338	177
0.9	0.6	25	284	123
0.9	0.6	50	261	100
0.9	0.6	100	251	90
0.9	0.7	5	317	156
0.9	0.7	25	278	117
0.9	0.7	50	258	97
0.9	0.7	100	250	89
0.9	0.8	5	329	168
0.9	0.8	25	260	99
0.9	0.8	50	277	116
0.9	0.8	100	254	93
0.9	0.9	5	309	148
0.9	0.9	25	251	90
0.9	0.9	50	269	108
0.9	0.9	100	246	85

ПРИЛОЖЕНИЕ Б

Таблица Б.1 – Параметризация муравьиного алгоритма для второго класса данных

α	ρ	t_{max}	Результат	Ошибка
0.1	0.1	5	23060	11189
0.1	0.1	25	16040	4169
0.1	0.1	50	16163	4292
0.1	0.1	100	14339	2468
0.1	0.2	5	19499	7628
0.1	0.2	25	16575	4704
0.1	0.2	50	15731	3860
0.1	0.2	100	14339	2468
0.1	0.3	5	20654	8783
0.1	0.3	25	16913	5042
0.1	0.3	50	14710	2839
0.1	0.3	100	13916	2045
0.1	0.4	5	22479	10608
0.1	0.4	25	16040	4169
0.1	0.4	50	14339	2468
0.1	0.4	100	13916	2045
0.1	0.5	5	21307	9436
0.1	0.5	25	16040	4169
0.1	0.5	50	15317	3446
0.1	0.5	100	14339	2468
0.1	0.6	5	20951	9080
0.1	0.6	25	16163	4292
0.1	0.6	50	15731	3860
0.1	0.6	100	13916	2045
0.1	0.7	5	18937	7066
0.1	0.7	25	16163	4292
0.1	0.7	50	14339	2468

Таблица Б.1 – Параметризация муравьиного алгоритма для второго класса данных

α	ρ	t_{max}	Результат	Ошибка
0.1	0.7	100	14339	2468
0.1	0.8	5	20444	8573
0.1	0.8	25	15916	4045
0.1	0.8	50	15317	3446
0.1	0.8	100	13916	2045
0.1	0.9	5	20152	8281
0.1	0.9	25	17465	5594
0.1	0.9	50	15380	3509
0.1	0.9	100	13916	2045
0.2	0.1	5	23677	11806
0.2	0.1	25	17479	5608
0.2	0.1	50	15731	3860
0.2	0.1	100	15753	3882
0.2	0.2	5	20071	8200
0.2	0.2	25	17592	5721
0.2	0.2	50	17024	5153
0.2	0.2	100	14339	2468
0.2	0.3	5	19907	8036
0.2	0.3	25	17326	5455
0.2	0.3	50	16163	4292
0.2	0.3	100	14339	2468
0.2	0.4	5	19549	7678
0.2	0.4	25	17423	5552
0.2	0.4	50	15916	4045
0.2	0.4	100	15380	3509
0.2	0.5	5	22451	10580
0.2	0.5	25	17592	5721
0.2	0.5	50	15731	3860
0.2	0.5	100	15689	3818
0.2	0.6	5	21732	9861

Таблица Б.1 – Параметризация муравьиного алгоритма для второго класса данных

α	ρ	t_{max}	Результат	Ошибка
0.2	0.6	25	18372	6501
0.2	0.6	50	16040	4169
0.2	0.6	100	15731	3860
0.2	0.7	5	22159	10288
0.2	0.7	25	17479	5608
0.2	0.7	50	17024	5153
0.2	0.7	100	14339	2468
0.2	0.8	5	19962	8091
0.2	0.8	25	17479	5608
0.2	0.8	50	15731	3860
0.2	0.8	100	15731	3860
0.2	0.9	5	22628	10757
0.2	0.9	25	17326	5455
0.2	0.9	50	15040	3169
0.2	0.9	100	15731	3860
0.3	0.1	5	21826	9955
0.3	0.1	25	17423	5552
0.3	0.1	50	16018	4147
0.3	0.1	100	17024	5153
0.3	0.2	5	21013	9142
0.3	0.2	25	19766	7895
0.3	0.2	50	17326	5455
0.3	0.2	100	15689	3818
0.3	0.3	5	22344	10473
0.3	0.3	25	19671	7800
0.3	0.3	50	16163	4292
0.3	0.3	100	16163	4292
0.3	0.4	5	23085	11214
0.3	0.4	25	20220	8349
0.3	0.4	50	17602	5731

Таблица Б.1 – Параметризация муравьиного алгоритма для второго класса данных

α	ρ	t_{max}	Результат	Ошибка
0.3	0.4	100	16040	4169
0.3	0.5	5	21426	9555
0.3	0.5	25	17602	5731
0.3	0.5	50	16040	4169
0.3	0.5	100	15380	3509
0.3	0.6	5	21732	9861
0.3	0.6	25	18170	6299
0.3	0.6	50	16061	4190
0.3	0.6	100	15916	4045
0.3	0.7	5	22325	10454
0.3	0.7	25	17734	5863
0.3	0.7	50	15731	3860
0.3	0.7	100	15380	3509
0.3	0.8	5	22078	10207
0.3	0.8	25	18477	6606
0.3	0.8	50	17912	6041
0.3	0.8	100	15963	4092
0.3	0.9	5	21532	9661
0.3	0.9	25	17833	5962
0.3	0.9	50	15916	4045
0.3	0.9	100	15877	4006
0.4	0.1	5	22718	10847
0.4	0.1	25	18617	6746
0.4	0.1	50	17975	6104
0.4	0.1	100	16247	4376
0.4	0.2	5	21693	9822
0.4	0.2	25	19861	7990
0.4	0.2	50	18175	6304
0.4	0.2	100	15380	3509
0.4	0.3	5	26958	15087

Таблица Б.1 – Параметризация муравьиного алгоритма для второго класса данных

α	ρ	t_{max}	Результат	Ошибка
0.4	0.3	25	20654	8783
0.4	0.3	50	17439	5568
0.4	0.3	100	17024	5153
0.4	0.4	5	23759	11888
0.4	0.4	25	17912	6041
0.4	0.4	50	17602	5731
0.4	0.4	100	15731	3860
0.4	0.5	5	22915	11044
0.4	0.5	25	19108	7237
0.4	0.5	50	18129	6258
0.4	0.5	100	15731	3860
0.4	0.6	5	23885	12014
0.4	0.6	25	18882	7011
0.4	0.6	50	18332	6461
0.4	0.6	100	16247	4376
0.4	0.7	5	23476	11605
0.4	0.7	25	20771	8900
0.4	0.7	50	15963	4092
0.4	0.7	100	16040	4169
0.4	0.8	5	24443	12572
0.4	0.8	25	20056	8185
0.4	0.8	50	14339	2468
0.4	0.8	100	16163	4292
0.4	0.9	5	23307	11436
0.4	0.9	25	20946	9075
0.4	0.9	50	17833	5962
0.4	0.9	100	17439	5568
0.5	0.1	5	26416	14545
0.5	0.1	25	21798	9927
0.5	0.1	50	18617	6746

Таблица Б.1 – Параметризация муравьиного алгоритма для второго класса данных

α	ρ	t_{max}	Результат	Ошибка
0.5	0.1	100	15731	3860
0.5	0.2	5	26273	14402
0.5	0.2	25	18436	6565
0.5	0.2	50	18781	6910
0.5	0.2	100	17024	5153
0.5	0.3	5	23710	11839
0.5	0.3	25	19243	7372
0.5	0.3	50	18831	6960
0.5	0.3	100	16040	4169
0.5	0.4	5	25685	13814
0.5	0.4	25	21806	9935
0.5	0.4	50	18601	6730
0.5	0.4	100	16388	4517
0.5	0.5	5	25926	14055
0.5	0.5	25	20993	9122
0.5	0.5	50	18146	6275
0.5	0.5	100	16247	4376
0.5	0.6	5	24608	12737
0.5	0.6	25	19499	7628
0.5	0.6	50	17852	5981
0.5	0.6	100	16913	5042
0.5	0.7	5	24573	12702
0.5	0.7	25	20829	8958
0.5	0.7	50	20300	8429
0.5	0.7	100	16061	4190
0.5	0.8	5	25350	13479
0.5	0.8	25	20681	8810
0.5	0.8	50	18978	7107
0.5	0.8	100	17572	5701
0.5	0.9	5	24722	12851

Таблица Б.1 – Параметризация муравьиного алгоритма для второго класса данных

α	ρ	t_{max}	Результат	Ошибка
0.5	0.9	25	20234	8363
0.5	0.9	50	19440	7569
0.5	0.9	100	18175	6304
0.6	0.1	5	26265	14394
0.6	0.1	25	22779	10908
0.6	0.1	50	20220	8349
0.6	0.1	100	19890	8019
0.6	0.2	5	25954	14083
0.6	0.2	25	19300	7429
0.6	0.2	50	18937	7066
0.6	0.2	100	17506	5635
0.6	0.3	5	26706	14835
0.6	0.3	25	21897	10026
0.6	0.3	50	20029	8158
0.6	0.3	100	17513	5642
0.6	0.4	5	26318	14447
0.6	0.4	25	21922	10051
0.6	0.4	50	19737	7866
0.6	0.4	100	18025	6154
0.6	0.5	5	24792	12921
0.6	0.5	25	19313	7442
0.6	0.5	50	18781	6910
0.6	0.5	100	18175	6304
0.6	0.6	5	26379	14508
0.6	0.6	25	21448	9577
0.6	0.6	50	20576	8705
0.6	0.6	100	18022	6151
0.6	0.7	5	30844	18973
0.6	0.7	25	22038	10167
0.6	0.7	50	21481	9610

Таблица Б.1 – Параметризация муравьиного алгоритма для второго класса данных

α	ρ	t_{max}	Результат	Ошибка
0.6	0.7	100	18071	6200
0.6	0.8	5	24128	12257
0.6	0.8	25	19362	7491
0.6	0.8	50	19708	7837
0.6	0.8	100	18170	6299
0.6	0.9	5	29514	17643
0.6	0.9	25	20655	8784
0.6	0.9	50	19700	7829
0.6	0.9	100	19053	7182
0.7	0.1	5	30467	18596
0.7	0.1	25	24449	12578
0.7	0.1	50	19275	7404
0.7	0.1	100	18978	7107
0.7	0.2	5	30315	18444
0.7	0.2	25	21013	9142
0.7	0.2	50	19275	7404
0.7	0.2	100	18477	6606
0.7	0.3	5	28810	16939
0.7	0.3	25	22250	10379
0.7	0.3	50	20755	8884
0.7	0.3	100	19255	7384
0.7	0.4	5	27823	15952
0.7	0.4	25	23007	11136
0.7	0.4	50	20748	8877
0.7	0.4	100	19440	7569
0.7	0.5	5	26219	14348
0.7	0.5	25	22284	10413
0.7	0.5	50	23177	11306
0.7	0.5	100	21824	9953
0.7	0.6	5	28162	16291

Таблица Б.1 – Параметризация муравьиного алгоритма для второго класса данных

α	ρ	t_{max}	Результат	Ошибка
0.7	0.6	25	23072	11201
0.7	0.6	50	23458	11587
0.7	0.6	100	20609	8738
0.7	0.7	5	28500	16629
0.7	0.7	25	22605	10734
0.7	0.7	50	20071	8200
0.7	0.7	100	18876	7005
0.7	0.8	5	27174	15303
0.7	0.8	25	22906	11035
0.7	0.8	50	21775	9904
0.7	0.8	100	19275	7404
0.7	0.9	5	28773	16902
0.7	0.9	25	24753	12882
0.7	0.9	50	20714	8843
0.7	0.9	100	21283	9412
0.8	0.1	5	29337	17466
0.8	0.1	25	25776	13905
0.8	0.1	50	22198	10327
0.8	0.1	100	20152	8281
0.8	0.2	5	31448	19577
0.8	0.2	25	23546	11675
0.8	0.2	50	23875	12004
0.8	0.2	100	22078	10207
0.8	0.3	5	31561	19690
0.8	0.3	25	23821	11950
0.8	0.3	50	22284	10413
0.8	0.3	100	19148	7277
0.8	0.4	5	30177	18306
0.8	0.4	25	26653	14782
0.8	0.4	50	20263	8392

Таблица Б.1 – Параметризация муравьиного алгоритма для второго класса данных

α	ρ	t_{max}	Результат	Ошибка
0.8	0.4	100	19243	7372
0.8	0.5	5	32430	20559
0.8	0.5	25	23700	11829
0.8	0.5	50	23195	11324
0.8	0.5	100	18916	7045
0.8	0.6	5	28654	16783
0.8	0.6	25	26648	14777
0.8	0.6	50	23077	11206
0.8	0.6	100	22719	10848
0.8	0.7	5	28161	16290
0.8	0.7	25	24380	12509
0.8	0.7	50	20472	8601
0.8	0.7	100	20194	8323
0.8	0.8	5	29927	18056
0.8	0.8	25	24780	12909
0.8	0.8	50	22430	10559
0.8	0.8	100	21759	9888
0.8	0.9	5	29428	17557
0.8	0.9	25	24577	12706
0.8	0.9	50	24123	12252
0.8	0.9	100	20444	8573
0.9	0.1	5	29572	17701
0.9	0.1	25	23786	11915
0.9	0.1	50	24566	12695
0.9	0.1	100	22779	10908
0.9	0.2	5	29490	17619
0.9	0.2	25	24786	12915
0.9	0.2	50	24062	12191
0.9	0.2	100	20921	9050
0.9	0.3	5	29786	17915

Таблица Б.1 – Параметризация муравьиного алгоритма для второго класса данных

α	ρ	t_{max}	Результат	Ошибка
0.9	0.3	25	25894	14023
0.9	0.3	50	25369	13498
0.9	0.3	100	21784	9913
0.9	0.4	5	33109	21238
0.9	0.4	25	24192	12321
0.9	0.4	50	23535	11664
0.9	0.4	100	22734	10863
0.9	0.5	5	31154	19283
0.9	0.5	25	24566	12695
0.9	0.5	50	24128	12257
0.9	0.5	100	22136	10265
0.9	0.6	5	32073	20202
0.9	0.6	25	26462	14591
0.9	0.6	50	23498	11627
0.9	0.6	100	23583	11712
0.9	0.7	5	27670	15799
0.9	0.7	25	27466	15595
0.9	0.7	50	23198	11327
0.9	0.7	100	23720	11849
0.9	0.8	5	32014	20143
0.9	0.8	25	26411	14540
0.9	0.8	50	24825	12954
0.9	0.8	100	20361	8490
0.9	0.9	5	32716	20845
0.9	0.9	25	26788	14917
0.9	0.9	50	21940	10069
0.9	0.9	100	23291	11420

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. *Володина Е.В. С. Е.* Практическое применение алгоритма решения задачи коммивояжера // ИВД. — 2015. — № 2.
2. *Colorni A.*, *Dorigo M.*, *Maniezzo V.* Distributed Optimization by Ant Colonies // Proceedings of the First European Conference on Artificial Life. — 1991.
3. The official home of the Python Programming Language [Электронный ресурс]. — Режим доступа: `https://www.python.org/` (дата обращения: 19.09.2023).
4. time — Time access and conversions [Электронный ресурс]. — Режим доступа: `https://docs.python.org/3/library/time.html` (дата обращения: 19.09.2023).
5. Amd [Электронный ресурс]. — Режим доступа: `https://www.amd.com/en.html` (дата обращения: 28.09.2023).
6. Windows 10 Pro 22h2 64-bit [Электронный ресурс]. — Режим доступа: `https://www.microsoft.com/ru-ru/software-download/windows10` (дата обращения: 28.09.2023).