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: «Методы решения задачи коммивояжера»

Студент	ИУ7-52Б (Группа)	(Подпись, дата)	А. П. Лемешев (И. О. Фамилия)
Преподаватель		(Подпись, дата)	Ю. В. Строганов (И. О. Фамилия)
Преподаватель		(Подпись, дата)	Л. Л. Волкова (И. О. Фамилия)

Москва — 2023 г.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Задача коммивояжера находится в центре внимания с 1960 года. Суть ее состоит в том, чтобы найти кратчайший круговой маршрут, включающий посещение определенного числа n вершин, причем начальная и конечная вершины являются одинаковыми, и каждая последующая вершина входит в этот круговой маршрут один раз [1].

Эта задача имеет большое количество практических приложений, особенно в сфере логистики (например, утилизация бытовых отходов, доставка товаров со склада, раздача хлебобулочных изделий из пекарен в отдельные магазины, планирование маршрута школьного автобуса, планирование услуг в компаниях, службы доставки, сверление отверстий под печатные платы, компьютерные системы, управление промышленными роботами, оптимизация схем, проектирование сетей и многое другое) [1].

В последние два десятилетия при оптимизации сложных систем исследователи все чаще применяют природные механизмы поиска наилучших решений. Эти механизмы обеспечивают эффективную адаптацию флоры и фауны к окружающей среде на протяжении миллионов лет. Муравьиные алгоритмы серьезно исследуются европейскими учеными с середины 90-х годов. На сегодня уже получены хорошие результаты муравьиной оптимизации таких сложных комбинаторных задач, как: задачи коммивояжера, задачи оптимизации маршрутов грузовиков, задачи раскраски графа, квадратичной задачи о назначениях, оптимизации сетевых графиков, задачи календарного планирования и других [2].

1 Аналитическая часть

1.1 Цели и задачи

Цель работы: изучить задачу коммивояжера и реализовать алгоритм полного перебора и муравьиный алгоритм для ее решения.

Задачи лабораторной работы:

1) исследовать задачу коммивояжера;
2) реализовать алгоритм полного перебора для решения задачи коммивояжера;
3) реализовать муравьиный алгоритм для решения задачи коммивояжера;
4) провести параметризацию муравьиного алгоритма на нескольких классах данных;
5) провести сравнительный анализ времени работы двух алгоритмов для решения задачи коммивояжера на основе экспериментальных данных.

1.2 Задача коммивояжера

Коммивояжёр — разъездной представитель торговой фирмы, который по поручению фирмы ищет покупателей ее товаров, предлагая им образцы, рекламируя товар, распространяя каталоги товаров.

В классической постановке коммивояжёр должен объехать N городов по замкнутому маршруту, посетив каждый из них лишь однажды, таким образом, чтобы полная длина его маршрута была минимальной. Если решать задачу коммивояжёра с помощью полного перебора — перебором всех замкнутых путей, связывающих города, то придется проверить все возможные маршруты, то есть простой метод перебора всех вариантов чрезвычайно неэффективный при большом N. Эффективными же признаются решения, гарантирующие получение ответа за время, ограниченное полиномом от размерности задачи. С помощью муравьиных алгоритмов находятся субоптимальные решения, локальные минимумы целевой функции, приближающиеся к абсолютному минимуму.

1.3 Алгоритм полного перебора

Самое простое решение — попробовать все перестановки множества вершин и посмотреть, какая из них возвращает в результате наименьшую длину пути. Очевидно, что время работы данного алгоритма — $O(n!)$. Это факториал количества вершин, поэтому данное решение становится непрактичным даже для небольшого числа вершин. С другой стороны, благодаря полному перебору алгоритм гарантирует получение пользователем корректного решения задачи коммивояжера.

1.4 Муравьиный алгоритм

Многократность взаимодействия мураьвев реализуется итерационным поиском маршрута коммивояжера одновременно несколькими муравьями. При этом каждый муравей рассматривается как отдельный, независимый коммивояжер, решающий свою задачу. За одну итерацию алгоритма каждый муравей совершает полный маршрут коммивояжера [2].

Положительная обратная связь реализуется как имитация поведения муравьев типа «оставление следов — перемещение по следам». Чем больше следов оставлено на тропе — ребре графа в задаче коммивояжера, тем больше муравьев будет передвигаться по ней. При этом на тропе появляются новые следы, привлекающие дополнительных муравьев. Для задачи коммивояжера положительная обратная связь реализуется следующим стохастическим правилом: вероятность включения ребра графа в маршрут муравья пропорциональна количеству феромона на нем [2].

Применение такого вероятностного правила обеспечивает реализацию и другой составляющей самоорганизации — случайности. Количество откладываемого муравьем феромона на ребре графа обратно пропорционально длине маршрута. Чем короче маршрут, тем больше феромона будет отложено на соответствующих ребрах графа и тем больше муравьев будет использовать их при синтезе своих маршрутов. Отложенный на ребрах феромон выступает как усилитель, он позволяет хорошим маршрутам сохраняться в глобальной памяти муравейника. Эти маршруты могут быть улучшены на последующих итерациях алгоритма [2].

Использование только положительной обратной связи приводит к преждевременной сходимости решений — к случаю, когда все муравьи двигаются одним и тем же субоптимальным маршрутом. Для избежания этого используется отрицательная обратная связь — испарение феромона. Время испарения не должно быть слишком большим, так как при этом возникает опасность сходимости популяции маршрутов к одному субоптимальному решению. С другой стороны, время испарения не должно быть и слишком малым, так как это приводит к быстрому «забыванию», потере памяти колонии и, следовательно, к некооперативному поведению муравьев. В поведении муравьев кооперативность является очень важной: множество идентичных муравьев одновременно исследуют разные точки пространства решений и передают свой опыт через изменения ячеек глобальной памяти муравейника [2].

Для каждого муравья переход из города i в город j зависит от трех составляющих: памяти муравья (tabu list), видимости и виртуального следа феромона [2].

Tabu list (память муравья) — это список посещенных муравьем городов, заходить в которые еще раз нельзя. Используя этот список, муравей гарантированно не попадет в один и тот же город дважды. Ясно, что tabu list возрастает при совершении маршрута и обнуляется в начале каждой итерации алгоритма. Обозначим через $J_{i,k}$ список городов, которые еще необходимо посетить муравью k, находящемуся в городе i. Понятно, что $J_{i,k}$ является дополнением к tabu list [2].

Видимость — величина, обратная расстоянию: $\eta_{ij} = frac1D_{ij}$, где D_{ij} — расстояние между городами i и j. Видимость — это локальная статическая информация, выражающая эвристическое желание посетить город j из города i — чем ближе город, тем больше желание посетить его. Использование только видимости, конечно, является недостаточным для нахождения оптимального маршрута [2].

Виртуальный след феромона на ребре (i, j) представляет подтвержденное муравьиным опытом желание посетить город j из города i. В отличие от видимости след феромона является более глобальной и динамичной информацией — она изменяется после каждой итерации алгоритма, отражая приобретенный муравьями опыт. Количество виртуального феромона на ребре (i, j) на итерации t обозначим через $\tau_{ij}(t)$

[2].

Важную роль в муравьиных алгоритмах играет вероятностно-пропорциональное правило, определяющее вероятность перехода k-го муравья из города i в город j на t-й итерации:

$$P_{ij,k}(t) = \begin{cases} \frac{\tau_{ij}^{\alpha}(t)\cdot\eta_{ij}^{\beta}}{\sum\limits_{l\in J_{i,k}} \tau_{il}^{\alpha}(t)\cdot\eta_{il}^{\beta}}, & \text{если } j \in J_{i,k}, \\ 0, & \text{иначе}, \end{cases} \tag{1.1}$$

где α и β — два регулируемых параметра, задающие веса следа феромона и видимости при выборе маршрута. При $\alpha = 0$ будет выбран ближайший город, что соответствует жадному алгоритму в классической теории оптимизации. Если $\beta = 0$, тогда работает лишь феромонное усиление, что влечет за собой быстрое вырождение маршрутов к одному субоптимальному решению [2].

После завершения маршрута каждый муравей k откладывает на ребре (i, j) такое количество феромона:

$$\Delta\tau_{ij,k}(t) = \begin{cases} \frac{Q}{L_k(t)}, & \text{если } (i,j) \in T_k(t), \\ 0, & \text{иначе}, \end{cases} \tag{1.2}$$

где $T_k(t)$ — маршрут, пройденный муравьем k на итерации t; $L_k(t)$ — длина этого маршрута; Q — регулируемый параметр, значение которого выбирают одного порядка с длиной оптимального маршрута [2].

Для исследования всего пространства решений необходимо обеспечить испарение феромона — уменьшение во времени количества отложенного на предыдущих итерациях феромона. Обозначим коэффициент испарения феромона через $p \in [0, 1]$. Тогда правило обновления феромона примет вид:

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

где $\Delta\tau_{ij}(t) = \sum\limits_{k=1}^{m} \Delta\tau_{ij,k}(t)$, m — количество муравьев в колонии [2].

В начале оптимизации количество феромона принимается равным небольшому положительному числу τ_0. Общее количество муравьев в колонии остается постоянным на протяжении выполнения алгоритма. Многочисленная колония приводит к быстрому усилению субоптимальных маршрутов, а когда муравьев мало, возникает опасность потери кооперативности поведения через ограниченное взаимодействие и быстрое испарение феромона. Обычно число муравьев назначают равным количеству городов — каждый муравей начинает маршрут со своего города [2].

Вывод

В текущем разделе была рассмотрена задача коммивояжера, а также алгоритм полного перебора и муравьиный алгоритм для ее решения.

2 Конструкторская часть

Разработка алгоритма

На рисунке 2.1 показана схема алгоритма полного перебора для решения задачи коммивояжера. На рисунке 2.2 показана схема муравьиного алгоритма для решения задачи коммивояжера.

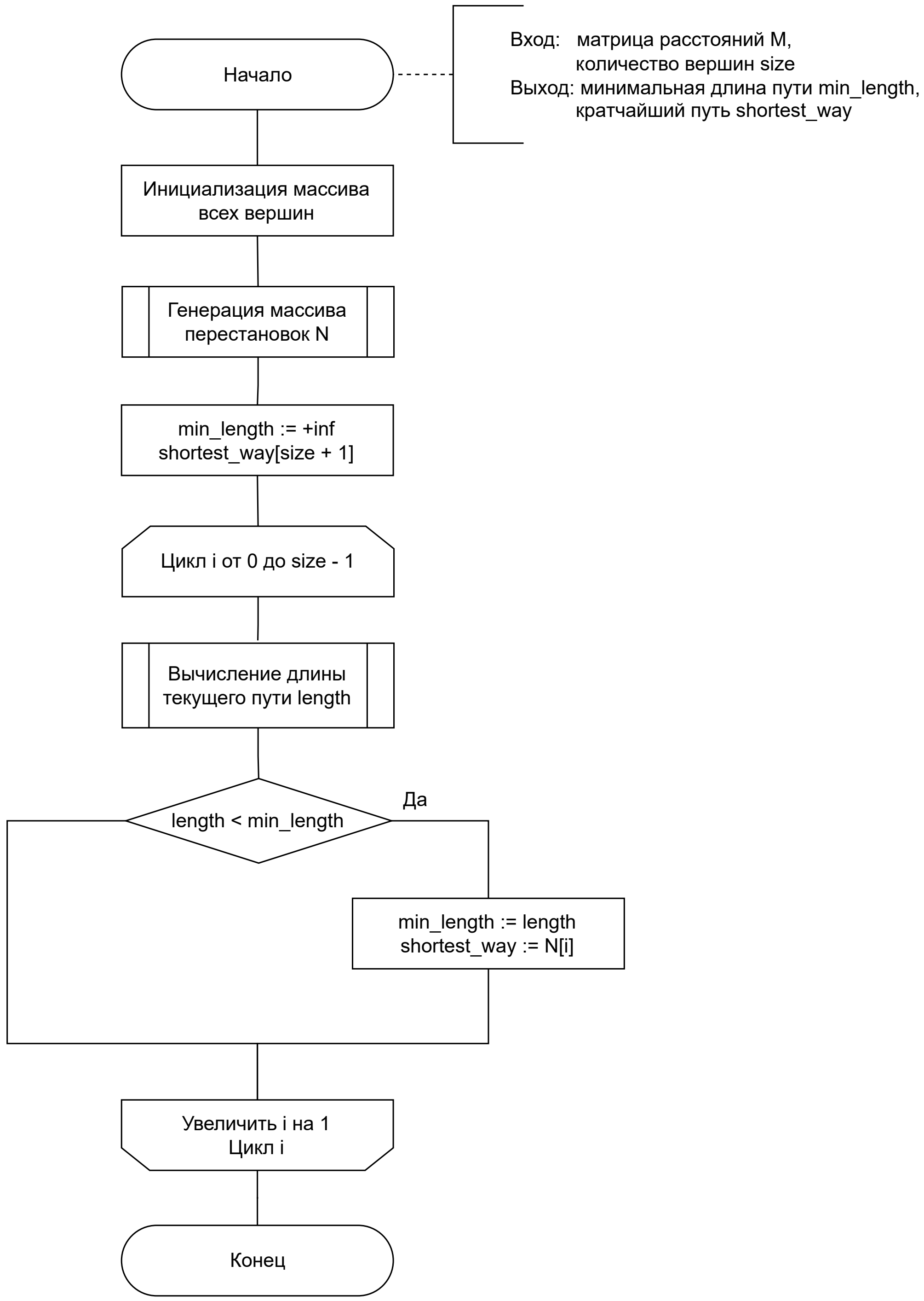

Рисунок 2.1 – Схема алгоритма полного перебора

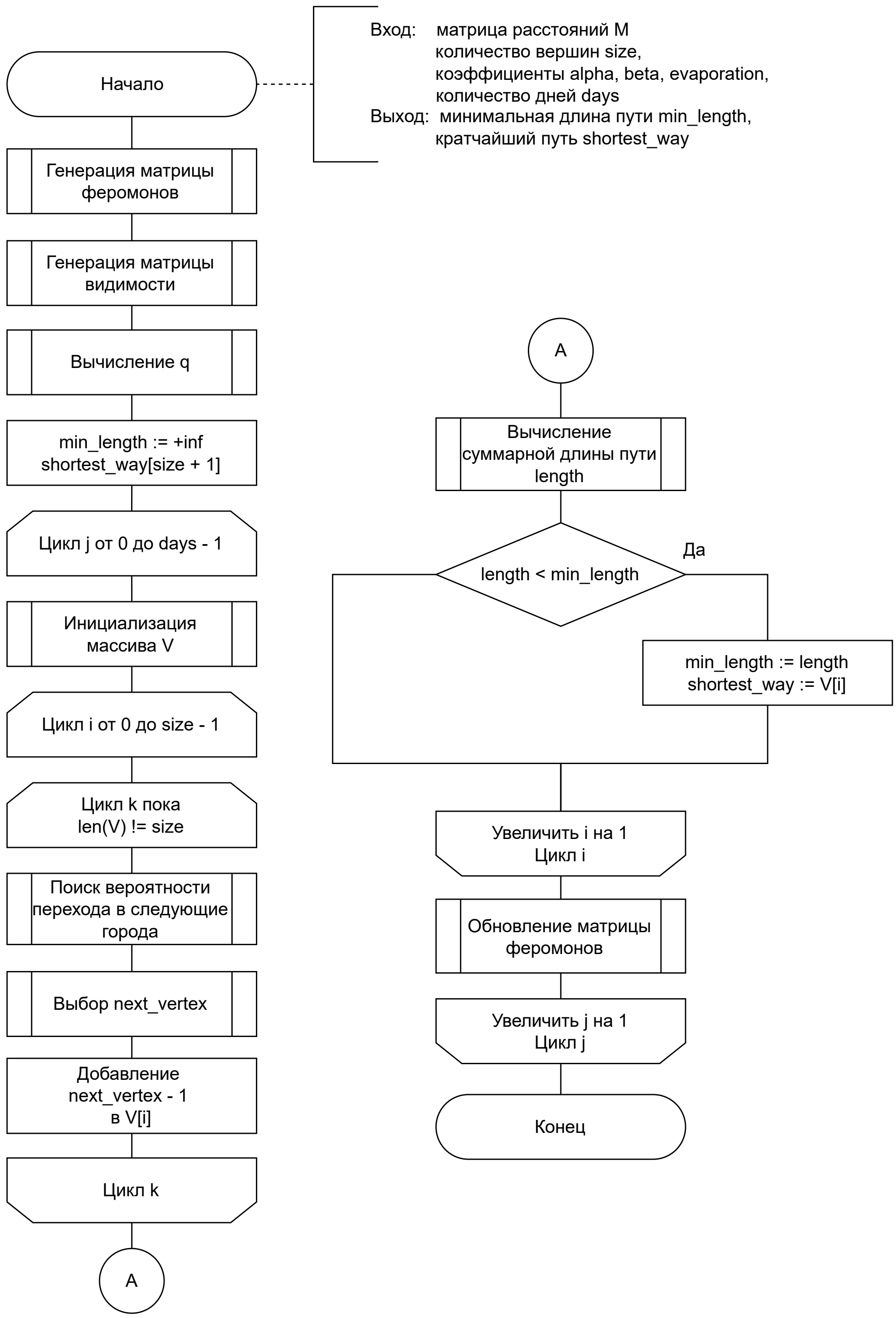

Рисунок 2.2 – Схема муравьиного алгоритма

Вывод

На основе теоретических данных, полученных в аналитическом разделе, были построены схемы алгоритма полного перебора и муравьиного алгоритма для решения задачи коммивояжера.

3 Технологическая часть

3.1 Требования к программному обеспечению

Программа должна отвечать следующим требованиям:

— программа получает на вход файл с матрицей смежности;

— программа выдает в результате кратчайший путь и массив вершин, по которым он строится;

— для вычисления кратчайшего пути используются алгоритма полного перебора и муравьиного алгоритма;

— программа должна измерять реальное время.

Программа должна обрабатывать ошибки (например, отсутствие файла с матрицей расстояний) и корректно завершать работу с выводом информации об ошибке на экран.

3.2 Средства реализации

В качестве языка программирования для реализации данной лабораторной работы был выбран *Python* ввиду следующих причин:

— мое желанием расширить свои знания в области применения данного языка;

— возможность измерять реальное время выполнения алгоритма;

— в языке реализован модуль *numpy*, который позволяет работать с матрицами.

Таким образом, с помощью языка *Python* можно реализовать программное обеспечение, которое соответствует перечисленным выше требованиям.

3.3 Реализация алгоритма

В листинге 3.2 показана реализация алгоритма полного перебора для решения задачи коммивояжера. В листинге 3.1 показана реализация муравьиного алгоритма для решения задачи коммивояжера.

Листинг 3.1 – Реализация муравьиного алгоритма

```
def aco(matrix, size, alpha, beta, evaporation, days):
    pmatrix = create_matrix(size)
    vmatrix  = get_vmatrix(matrix, size)
    q = get_q(matrix, size)
    shortest_way = []
    min_size = float("inf")
    for j in range(days):
        vis_arr = get_visited_vertices(np.arange(size), size)
        for i in range(size):
            while size(vis_arr[i]) != size:
                array = get_probability(pmatrix, vmatrix,
                   vis_arr, size, i, alpha, beta)
                next_place = get_next_vertex(array)
                vis_arr[i].append(next_place - 1)
            vis_arr[i].append(vis_arr[i][0])
            size = tsp.get_size(matrix, size, vis_arr[i])
            if size < min_size:
                min_size = size
                shortest_way = vis_arr[i]
        pmatrix = update_pmatrix(matrix, size, vis_arr,
           pmatrix, q, evaporation)
    for i in range(size(shortest_way)):
        shortest_way[i] += 1
    return min_size, shortest_way
```

Листинг 3.2 – Реализация алгоритма полного перебора

```python
def tsp(matrix, size):
    cities = np.arange(size)
    cities_combs = []
    for combination in it.permutations(cities):
        cities_combs.append(list(combination))
    shortest_way = []
    min_size = float("inf")
    for i in range(size(cities_combs)):
        cities_combs[i].append(cities_combs[i][0])
        size = get_size(matrix, size, cities_combs[i])
        if size < min_size:
            min_size = size
            shortest_way = cities_combs[i]
    for i in range(size(shortest_way)):
        shortest_way[i] += 1
    return min_size, shortest_way
```

3.4 Тестовые данные

В таблице 3.1 приведены тестовые данные для двух функций, реализующих алгоритмы для решения задачи коммивояжера (поиска гамильтонова цикла). Результата записаны в следующем формате: значение кратчайшего пути; кратчайший путь.

Тесты выполнялись по методологии черного ящика (модульное тестирование). Все тесты пройдены успешно.

Таблица 3.1 – Результаты тестирования

Матрица расстояний	Полный перебор	Муравьиный алгоритм
$\begin{pmatrix} 0 \end{pmatrix}$	0: [1, 1]	0: [1, 1]
$\begin{pmatrix} 0 & 10 \\ 10 & 0 \end{pmatrix}$	10: [1, 2, 1]	10: [1, 2, 1]
$\begin{pmatrix} 0 & 10 & 15 & 20 \\ 10 & 0 & 35 & 25 \\ 15 & 35 & 0 & 30 \\ 20 & 25 & 30 & 0 \end{pmatrix}$	80: [1, 2, 4, 3, 1]	80: [1, 2, 4, 3, 1]

Вывод

В текущем разделе был написан исходный код алгоритма полного перебора и муравьиного алгоритма для решения задачи коммивояжера. Описаны тесты и приведены результаты тестирования.

4 Исследовательская часть

4.1 Технические характеристики

Тестирование выполнялось на устройстве со следующими техническими характеристиками.

1. Операционная система: Ubuntu Linux 64-bit.
2. Память: 8 ГБ.
3. Процессор: AMD Ryzen 5 3550H.

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

4.2 Время работы алгоритмов

Время работы функций замерено с помощью функции *process_time_ns()* модуля *time*, которая возвращает количество наносекунд суммы системного и пользовательского процессорного времени текущего процесса.

В таблице 4.1 приведено время работы в миллисекундах функций, реализующих алгоритмы для решения задачи коммивояжера, в зависимости от количества вершин. На рисунке 4.1 изображена зависимость времени работы в миллисекундах функций, реализующих два алгоритма для решения задачи коммивояжера, от количества вершин.

Таблица 4.1 – Время работы алгоритмов

Количество вершин	**Время работы, нс**	
	Полный перебор	**Муравьиный алгоритм**
5	10	93
6	67	134
7	189	234
8	981	342
9	3678	452

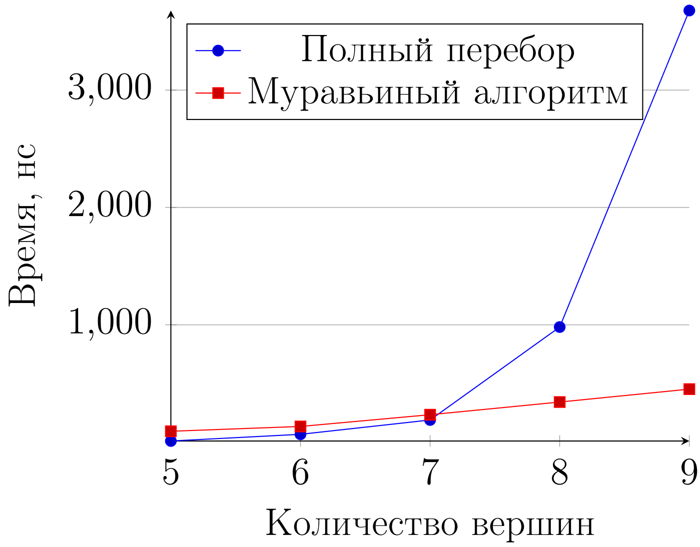

Рисунок 4.1 – Зависимость времени работы от алгоритма

4.3 Параметризация

Параметризация муравьиного алгоритма была проведена на двух классах данных. В качестве входных данных были сгенерированы две матрицы размером 10×10. Муравьиный алгоритм был запущен для всех значений $\alpha \in (0, 1)$ и $p \in (0, 1)$ с шагом 0.1. В качестве эталона был выбран результат работы алгоритма полного перебора для решения задачи коммивояжера.

4.3.1 Класс данных №1

Все расстояния между вершинами для класса данных №1 находятся на отрезке $[1; 5]$. Матрица расстояний для класса данных №1:

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

В таблице 4.2 показаны значения, полученные в результате проведения параметризации муравьиного алгоритма для класса данных №1.

Таблица 4.2 – Параметры для класса данных №1

α	β	p	Количество дней	Результат	Ошибка
0.1	0.9	0.1	50	20	1
0.1	0.9	0.1	100	20	1
0.1	0.9	0.1	200	20	1
0.1	0.9	0.2	50	20	1
0.1	0.9	0.2	100	20	0
0.1	0.9	0.2	200	20	1
0.1	0.9	0.3	50	20	1
0.1	0.9	0.3	100	20	1
0.1	0.9	0.3	200	20	1
0.1	0.9	0.4	50	20	1
0.1	0.9	0.4	100	20	1
0.1	0.9	0.4	200	20	1
0.1	0.9	0.5	50	20	1
0.1	0.9	0.5	100	20	1

Продолжение таблицы 4.2

α	β	p	Количество дней	Результат	Ошибка
0.1	0.9	0.5	200	20	1
0.1	0.9	0.6	50	20	1
0.1	0.9	0.6	100	20	1
0.1	0.9	0.6	200	20	0
0.1	0.9	0.7	50	20	1
0.1	0.9	0.7	100	20	1
0.1	0.9	0.7	200	20	0
0.1	0.9	0.8	50	20	1
0.1	0.9	0.8	100	20	1
0.1	0.9	0.8	200	20	1
0.1	0.9	0.9	50	20	1
0.1	0.9	0.9	100	20	0
0.1	0.9	0.9	200	20	1
0.2	0.8	0.1	50	20	1
0.2	0.8	0.1	100	20	0

Продолжение таблицы 4.2

α	β	p	Количество дней	Результат	Ошибка
0.2	0.8	0.1	200	20	0
0.2	0.8	0.2	50	20	1
0.2	0.8	0.2	100	20	1
0.2	0.8	0.2	200	20	1
0.2	0.8	0.3	50	20	1
0.2	0.8	0.3	100	20	0
0.2	0.8	0.3	200	20	1
0.2	0.8	0.4	50	20	1
0.2	0.8	0.4	100	20	1
0.2	0.8	0.4	200	20	1
0.2	0.8	0.5	50	20	2
0.2	0.8	0.5	100	20	0
0.2	0.8	0.5	200	20	0
0.2	0.8	0.6	50	20	0
0.2	0.8	0.6	100	20	1

Продолжение таблицы 4.2

α	β	p	Количество дней	Результат	Ошибка
0.2	0.8	0.6	200	20	0
0.2	0.8	0.7	50	20	1
0.2	0.8	0.7	100	20	0
0.2	0.8	0.7	200	20	1
0.2	0.8	0.8	50	20	1
0.2	0.8	0.8	100	20	1
0.2	0.8	0.8	200	20	1
0.2	0.8	0.9	50	20	1
0.2	0.8	0.9	100	20	1
0.2	0.8	0.9	200	20	0
0.3	0.7	0.1	50	20	2
0.3	0.7	0.1	100	20	1
0.3	0.7	0.1	200	20	0
0.3	0.7	0.2	50	20	2
0.3	0.7	0.2	100	20	1

Продолжение таблицы 4.2

α	β	p	Количество дней	Результат	Ошибка
0.3	0.7	0.2	200	20	1
0.3	0.7	0.3	50	20	1
0.3	0.7	0.3	100	20	1
0.3	0.7	0.3	200	20	1
0.3	0.7	0.4	50	20	1
0.3	0.7	0.4	100	20	1
0.3	0.7	0.4	200	20	1
0.3	0.7	0.5	50	20	1
0.3	0.7	0.5	100	20	1
0.3	0.7	0.5	200	20	1
0.3	0.7	0.6	50	20	1
0.3	0.7	0.6	100	20	1
0.3	0.7	0.6	200	20	1
0.3	0.7	0.7	50	20	1
0.3	0.7	0.7	100	20	1

Продолжение таблицы 4.2

α	β	p	Количество дней	Результат	Ошибка
0.3	0.7	0.7	200	20	1
0.3	0.7	0.8	50	20	2
0.3	0.7	0.8	100	20	1
0.3	0.7	0.8	200	20	1
0.3	0.7	0.9	50	20	1
0.3	0.7	0.9	100	20	1
0.3	0.7	0.9	200	20	1
0.4	0.6	0.1	50	20	1
0.4	0.6	0.1	100	20	1
0.4	0.6	0.1	200	20	1
0.4	0.6	0.2	50	20	1
0.4	0.6	0.2	100	20	0
0.4	0.6	0.2	200	20	1
0.4	0.6	0.3	50	20	1
0.4	0.6	0.3	100	20	0

Продолжение таблицы 4.2

α	β	p	Количество дней	Результат	Ошибка
0.4	0.6	0.3	200	20	0
0.4	0.6	0.4	50	20	1
0.4	0.6	0.4	100	20	1
0.4	0.6	0.4	200	20	0
0.4	0.6	0.5	50	20	1
0.4	0.6	0.5	100	20	1
0.4	0.6	0.5	200	20	0
0.4	0.6	0.6	50	20	1
0.4	0.6	0.6	100	20	1
0.4	0.6	0.6	200	20	1
0.4	0.6	0.7	50	20	1
0.4	0.6	0.7	100	20	1
0.4	0.6	0.7	200	20	1
0.4	0.6	0.8	50	20	1
0.4	0.6	0.8	100	20	1

Продолжение таблицы 4.2

α	β	p	Количество дней	Результат	Ошибка
0.4	0.6	0.8	200	20	1
0.4	0.6	0.9	50	20	1
0.4	0.6	0.9	100	20	1
0.4	0.6	0.9	200	20	1
0.5	0.5	0.1	50	20	1
0.5	0.5	0.1	100	20	1
0.5	0.5	0.1	200	20	1
0.5	0.5	0.2	50	20	1
0.5	0.5	0.2	100	20	1
0.5	0.5	0.2	200	20	1
0.5	0.5	0.3	50	20	1
0.5	0.5	0.3	100	20	1
0.5	0.5	0.3	200	20	1
0.5	0.5	0.4	50	20	1
0.5	0.5	0.4	100	20	1

Продолжение таблицы 4.2

α	β	p	Количество дней	Результат	Ошибка
0.5	0.5	0.4	200	20	1
0.5	0.5	0.5	50	20	1
0.5	0.5	0.5	100	20	1
0.5	0.5	0.5	200	20	1
0.5	0.5	0.6	50	20	1
0.5	0.5	0.6	100	20	2
0.5	0.5	0.6	200	20	1
0.5	0.5	0.7	50	20	1
0.5	0.5	0.7	100	20	2
0.5	0.5	0.7	200	20	1
0.5	0.5	0.8	50	20	2
0.5	0.5	0.8	100	20	2
0.5	0.5	0.8	200	20	0
0.5	0.5	0.9	50	20	1
0.5	0.5	0.9	100	20	1

Продолжение таблицы 4.2

α	β	p	Количество дней	Результат	Ошибка
0.5	0.5	0.9	200	20	1
0.6	0.4	0.1	50	20	1
0.6	0.4	0.1	100	20	1
0.6	0.4	0.1	200	20	1
0.6	0.4	0.2	50	20	1
0.6	0.4	0.2	100	20	1
0.6	0.4	0.2	200	20	1
0.6	0.4	0.3	50	20	1
0.6	0.4	0.3	100	20	1
0.6	0.4	0.3	200	20	1
0.6	0.4	0.4	50	20	2
0.6	0.4	0.4	100	20	1
0.6	0.4	0.4	200	20	1
0.6	0.4	0.5	50	20	2
0.6	0.4	0.5	100	20	1

Продолжение таблицы 4.2

α	β	p	Количество дней	Результат	Ошибка
0.6	0.4	0.5	200	20	2
0.6	0.4	0.6	50	20	2
0.6	0.4	0.6	100	20	2
0.6	0.4	0.6	200	20	1
0.6	0.4	0.7	50	20	1
0.6	0.4	0.7	100	20	1
0.6	0.4	0.7	200	20	1
0.6	0.4	0.8	50	20	1
0.6	0.4	0.8	100	20	1
0.6	0.4	0.8	200	20	1
0.6	0.4	0.9	50	20	1
0.6	0.4	0.9	100	20	1
0.6	0.4	0.9	200	20	1
0.7	0.3	0.1	50	20	1
0.7	0.3	0.1	100	20	1

Продолжение таблицы 4.2

α	β	p	Количество дней	Результат	Ошибка
0.7	0.3	0.1	200	20	2
0.7	0.3	0.2	50	20	0
0.7	0.3	0.2	100	20	1
0.7	0.3	0.2	200	20	1
0.7	0.3	0.3	50	20	3
0.7	0.3	0.3	100	20	1
0.7	0.3	0.3	200	20	0
0.7	0.3	0.4	50	20	1
0.7	0.3	0.4	100	20	1
0.7	0.3	0.4	200	20	1
0.7	0.3	0.5	50	20	2
0.7	0.3	0.5	100	20	1
0.7	0.3	0.5	200	20	2
0.7	0.3	0.6	50	20	3
0.7	0.3	0.6	100	20	2

Продолжение таблицы 4.2

α	β	p	Количество дней	Результат	Ошибка
0.7	0.3	0.6	200	20	1
0.7	0.3	0.7	50	20	2
0.7	0.3	0.7	100	20	1
0.7	0.3	0.7	200	20	1
0.7	0.3	0.8	50	20	0
0.7	0.3	0.8	100	20	2
0.7	0.3	0.8	200	20	1
0.7	0.3	0.9	50	20	1
0.7	0.3	0.9	100	20	3
0.7	0.3	0.9	200	20	1
0.8	0.2	0.1	50	20	2
0.8	0.2	0.1	100	20	3
0.8	0.2	0.1	200	20	1
0.8	0.2	0.2	50	20	3
0.8	0.2	0.2	100	20	1

Продолжение таблицы 4.2

α	β	p	Количество дней	Результат	Ошибка
0.8	0.2	0.2	200	20	1
0.8	0.2	0.3	50	20	1
0.8	0.2	0.3	100	20	1
0.8	0.2	0.3	200	20	1
0.8	0.2	0.4	50	20	2
0.8	0.2	0.4	100	20	1
0.8	0.2	0.4	200	20	1
0.8	0.2	0.5	50	20	1
0.8	0.2	0.5	100	20	1
0.8	0.2	0.5	200	20	1
0.8	0.2	0.6	50	20	3
0.8	0.2	0.6	100	20	1
0.8	0.2	0.6	200	20	1
0.8	0.2	0.7	50	20	2
0.8	0.2	0.7	100	20	2

Продолжение таблицы 4.2

α	β	p	Количество дней	Результат	Ошибка
0.8	0.2	0.7	200	20	1
0.8	0.2	0.8	50	20	2
0.8	0.2	0.8	100	20	2
0.8	0.2	0.8	200	20	2
0.8	0.2	0.9	50	20	1
0.8	0.2	0.9	100	20	2
0.8	0.2	0.9	200	20	1
0.9	0.1	0.1	50	20	1
0.9	0.1	0.1	100	20	3
0.9	0.1	0.1	200	20	1
0.9	0.1	0.2	50	20	3
0.9	0.1	0.2	100	20	2
0.9	0.1	0.2	200	20	1
0.9	0.1	0.3	50	20	1
0.9	0.1	0.3	100	20	3

Продолжение таблицы 4.2

α	β	p	Количество дней	Результат	Ошибка
0.9	0.1	0.3	200	20	1
0.9	0.1	0.4	50	20	4
0.9	0.1	0.4	100	20	1
0.9	0.1	0.4	200	20	1
0.9	0.1	0.5	50	20	3
0.9	0.1	0.5	100	20	2
0.9	0.1	0.5	200	20	1
0.9	0.1	0.6	50	20	1
0.9	0.1	0.6	100	20	1
0.9	0.1	0.6	200	20	0
0.9	0.1	0.7	50	20	2
0.9	0.1	0.7	100	20	2
0.9	0.1	0.7	200	20	2
0.9	0.1	0.8	50	20	3
0.9	0.1	0.8	100	20	3

Продолжение таблицы 4.2

α	β	p	Количество дней	Результат	Ошибка
0.9	0.1	0.8	200	20	2
0.9	0.1	0.9	50	20	2
0.9	0.1	0.9	100	20	1
0.9	0.1	0.9	200	20	1

4.3.2 Класс данных №2

Все расстояния между вершинами для класса данных №2 находятся на отрезке $[100; 3000]$. Матрица расстояний для класса данных №2:

$$B = \begin{pmatrix} 0 & 2573 & 1541 & 1785 & 237 & 1194 & 1950 & 1710 & 840 & 1608 \\ 2573 & 0 & 2158 & 376 & 2854 & 1395 & 967 & 2254 & 562 & 2803 \\ 1541 & 2158 & 0 & 2033 & 2548 & 2867 & 815 & 2421 & 688 & 134 \\ 1785 & 376 & 2033 & 0 & 2243 & 116 & 118 & 2842 & 926 & 1618 \\ 237 & 2854 & 2548 & 2243 & 0 & 1677 & 2177 & 2252 & 2102 & 228 \\ 1194 & 1395 & 2867 & 116 & 1677 & 0 & 157 & 1367 & 637 & 2295 \\ 1950 & 967 & 815 & 118 & 2177 & 157 & 0 & 105 & 804 & 2111 \\ 1710 & 2254 & 2421 & 2842 & 2252 & 1367 & 105 & 0 & 159 & 524 \\ 840 & 562 & 688 & 926 & 2102 & 637 & 804 & 159 & 0 & 1110 \\ 1608 & 2803 & 134 & 1618 & 228 & 2295 & 2111 & 524 & 1110 & 0 \end{pmatrix} \tag{4.2}$$

В таблице 4.3 показаны значения, полученные в результате проведения параметризации муравьиного алгоритма для класса данных №2.

Таблица 4.3 – Параметры для класса данных №2

α	β	p	Количество дней	Результат	Ошибка
0.1	0.9	0.1	50	3926	0
0.1	0.9	0.1	100	3926	278
0.1	0.9	0.1	200	3926	0
0.1	0.9	0.2	50	3926	0
0.1	0.9	0.2	100	3926	0
0.1	0.9	0.2	200	3926	0
0.1	0.9	0.3	50	3926	0
0.1	0.9	0.3	100	3926	0
0.1	0.9	0.3	200	3926	0
0.1	0.9	0.4	50	3926	278
0.1	0.9	0.4	100	3926	278
0.1	0.9	0.4	200	3926	0
0.1	0.9	0.5	50	3926	0
0.1	0.9	0.5	100	3926	0
0.1	0.9	0.5	200	3926	0

Продолжение таблицы 4.3

α	β	p	Количество дней	Результат	Ошибка
0.1	0.9	0.6	50	3926	278
0.1	0.9	0.6	100	3926	0
0.1	0.9	0.6	200	3926	0
0.1	0.9	0.7	50	3926	0
0.1	0.9	0.7	100	3926	0
0.1	0.9	0.7	200	3926	0
0.1	0.9	0.8	50	3926	0
0.1	0.9	0.8	100	3926	0
0.1	0.9	0.8	200	3926	0
0.1	0.9	0.9	50	3926	0
0.1	0.9	0.9	100	3926	0
0.1	0.9	0.9	200	3926	0
0.2	0.8	0.1	50	3926	584
0.2	0.8	0.1	100	3926	0
0.2	0.8	0.1	200	3926	0

Продолжение таблицы 4.3

α	β	p	Количество дней	Результат	Ошибка
0.2	0.8	0.2	50	3926	387
0.2	0.8	0.2	100	3926	0
0.2	0.8	0.2	200	3926	0
0.2	0.8	0.3	50	3926	0
0.2	0.8	0.3	100	3926	0
0.2	0.8	0.3	200	3926	0
0.2	0.8	0.4	50	3926	0
0.2	0.8	0.4	100	3926	0
0.2	0.8	0.4	200	3926	0
0.2	0.8	0.5	50	3926	387
0.2	0.8	0.5	100	3926	0
0.2	0.8	0.5	200	3926	387
0.2	0.8	0.6	50	3926	568
0.2	0.8	0.6	100	3926	0
0.2	0.8	0.6	200	3926	278

Продолжение таблицы 4.3

α	β	p	Количество дней	Результат	Ошибка
0.2	0.8	0.7	50	3926	568
0.2	0.8	0.7	100	3926	0
0.2	0.8	0.7	200	3926	0
0.2	0.8	0.8	50	3926	0
0.2	0.8	0.8	100	3926	0
0.2	0.8	0.8	200	3926	0
0.2	0.8	0.9	50	3926	0
0.2	0.8	0.9	100	3926	0
0.2	0.8	0.9	200	3926	0
0.3	0.7	0.1	50	3926	0
0.3	0.7	0.1	100	3926	0
0.3	0.7	0.1	200	3926	0
0.3	0.7	0.2	50	3926	278
0.3	0.7	0.2	100	3926	0
0.3	0.7	0.2	200	3926	278

Продолжение таблицы 4.3

α	β	p	Количество дней	Результат	Ошибка
0.3	0.7	0.3	50	3926	0
0.3	0.7	0.3	100	3926	0
0.3	0.7	0.3	200	3926	0
0.3	0.7	0.4	50	3926	0
0.3	0.7	0.4	100	3926	278
0.3	0.7	0.4	200	3926	0
0.3	0.7	0.5	50	3926	278
0.3	0.7	0.5	100	3926	0
0.3	0.7	0.5	200	3926	0
0.3	0.7	0.6	50	3926	278
0.3	0.7	0.6	100	3926	0
0.3	0.7	0.6	200	3926	0
0.3	0.7	0.7	50	3926	0
0.3	0.7	0.7	100	3926	0
0.3	0.7	0.7	200	3926	387

Продолжение таблицы 4.3

α	β	p	Количество дней	Результат	Ошибка
0.3	0.7	0.8	50	3926	0
0.3	0.7	0.8	100	3926	278
0.3	0.7	0.8	200	3926	0
0.3	0.7	0.9	50	3926	0
0.3	0.7	0.9	100	3926	0
0.3	0.7	0.9	200	3926	0
0.4	0.6	0.1	50	3926	387
0.4	0.6	0.1	100	3926	387
0.4	0.6	0.1	200	3926	278
0.4	0.6	0.2	50	3926	278
0.4	0.6	0.2	100	3926	278
0.4	0.6	0.2	200	3926	0
0.4	0.6	0.3	50	3926	0
0.4	0.6	0.3	100	3926	568
0.4	0.6	0.3	200	3926	0

Продолжение таблицы 4.3

α	β	p	Количество дней	Результат	Ошибка
0.4	0.6	0.4	50	3926	0
0.4	0.6	0.4	100	3926	278
0.4	0.6	0.4	200	3926	0
0.4	0.6	0.5	50	3926	0
0.4	0.6	0.5	100	3926	584
0.4	0.6	0.5	200	3926	0
0.4	0.6	0.6	50	3926	278
0.4	0.6	0.6	100	3926	0
0.4	0.6	0.6	200	3926	0
0.4	0.6	0.7	50	3926	278
0.4	0.6	0.7	100	3926	278
0.4	0.6	0.7	200	3926	568
0.4	0.6	0.8	50	3926	387
0.4	0.6	0.8	100	3926	568
0.4	0.6	0.8	200	3926	0

Продолжение таблицы 4.3

α	β	p	Количество дней	Результат	Ошибка
0.4	0.6	0.9	50	3926	387
0.4	0.6	0.9	100	3926	0
0.4	0.6	0.9	200	3926	0
0.5	0.5	0.1	50	3926	847
0.5	0.5	0.1	100	3926	568
0.5	0.5	0.1	200	3926	278
0.5	0.5	0.2	50	3926	1338
0.5	0.5	0.2	100	3926	278
0.5	0.5	0.2	200	3926	387
0.5	0.5	0.3	50	3926	1454
0.5	0.5	0.3	100	3926	584
0.5	0.5	0.3	200	3926	278
0.5	0.5	0.4	50	3926	608
0.5	0.5	0.4	100	3926	568
0.5	0.5	0.4	200	3926	0

Продолжение таблицы 4.3

α	β	p	Количество дней	Результат	Ошибка
0.5	0.5	0.5	50	3926	847
0.5	0.5	0.5	100	3926	584
0.5	0.5	0.5	200	3926	0
0.5	0.5	0.6	50	3926	1367
0.5	0.5	0.6	100	3926	0
0.5	0.5	0.6	200	3926	584
0.5	0.5	0.7	50	3926	387
0.5	0.5	0.7	100	3926	789
0.5	0.5	0.7	200	3926	0
0.5	0.5	0.8	50	3926	1378
0.5	0.5	0.8	100	3926	0
0.5	0.5	0.8	200	3926	0
0.5	0.5	0.9	50	3926	1085
0.5	0.5	0.9	100	3926	0
0.5	0.5	0.9	200	3926	278

Продолжение таблицы 4.3

α	β	p	Количество дней	Результат	Ошибка
0.6	0.4	0.1	50	3926	1559
0.6	0.4	0.1	100	3926	387
0.6	0.4	0.1	200	3926	0
0.6	0.4	0.2	50	3926	568
0.6	0.4	0.2	100	3926	1085
0.6	0.4	0.2	200	3926	278
0.6	0.4	0.3	50	3926	847
0.6	0.4	0.3	100	3926	708
0.6	0.4	0.3	200	3926	0
0.6	0.4	0.4	50	3926	1748
0.6	0.4	0.4	100	3926	789
0.6	0.4	0.4	200	3926	608
0.6	0.4	0.5	50	3926	1715
0.6	0.4	0.5	100	3926	0
0.6	0.4	0.5	200	3926	0

Продолжение таблицы 4.3

α	β	p	Количество дней	Результат	Ошибка
0.6	0.4	0.6	50	3926	0
0.6	0.4	0.6	100	3926	708
0.6	0.4	0.6	200	3926	0
0.6	0.4	0.7	50	3926	1250
0.6	0.4	0.7	100	3926	278
0.6	0.4	0.7	200	3926	568
0.6	0.4	0.8	50	3926	1453
0.6	0.4	0.8	100	3926	0
0.6	0.4	0.8	200	3926	278
0.6	0.4	0.9	50	3926	1548
0.6	0.4	0.9	100	3926	387
0.6	0.4	0.9	200	3926	387
0.7	0.3	0.1	50	3926	1588
0.7	0.3	0.1	100	3926	789
0.7	0.3	0.1	200	3926	708

Продолжение таблицы 4.3

α	β	p	Количество дней	Результат	Ошибка
0.7	0.3	0.2	50	3926	2288
0.7	0.3	0.2	100	3926	1610
0.7	0.3	0.2	200	3926	1259
0.7	0.3	0.3	50	3926	2813
0.7	0.3	0.3	100	3926	0
0.7	0.3	0.3	200	3926	387
0.7	0.3	0.4	50	3926	1872
0.7	0.3	0.4	100	3926	708
0.7	0.3	0.4	200	3926	0
0.7	0.3	0.5	50	3926	0
0.7	0.3	0.5	100	3926	1808
0.7	0.3	0.5	200	3926	278
0.7	0.3	0.6	50	3926	1928
0.7	0.3	0.6	100	3926	1083
0.7	0.3	0.6	200	3926	387

Продолжение таблицы 4.3

α	β	p	Количество дней	Результат	Ошибка
0.7	0.3	0.7	50	3926	919
0.7	0.3	0.7	100	3926	0
0.7	0.3	0.7	200	3926	1272
0.7	0.3	0.8	50	3926	278
0.7	0.3	0.8	100	3926	1250
0.7	0.3	0.8	200	3926	278
0.7	0.3	0.9	50	3926	387
0.7	0.3	0.9	100	3926	708
0.7	0.3	0.9	200	3926	0
0.8	0.2	0.1	50	3926	0
0.8	0.2	0.1	100	3926	2922
0.8	0.2	0.1	200	3926	1250
0.8	0.2	0.2	50	3926	2309
0.8	0.2	0.2	100	3926	1913
0.8	0.2	0.2	200	3926	1673

Продолжение таблицы 4.3

α	β	p	Количество дней	Результат	Ошибка
0.8	0.2	0.3	50	3926	2609
0.8	0.2	0.3	100	3926	847
0.8	0.2	0.3	200	3926	1454
0.8	0.2	0.4	50	3926	1945
0.8	0.2	0.4	100	3926	1674
0.8	0.2	0.4	200	3926	1259
0.8	0.2	0.5	50	3926	2083
0.8	0.2	0.5	100	3926	708
0.8	0.2	0.5	200	3926	1259
0.8	0.2	0.6	50	3926	2843
0.8	0.2	0.6	100	3926	1619
0.8	0.2	0.6	200	3926	1588
0.8	0.2	0.7	50	3926	2439
0.8	0.2	0.7	100	3926	568
0.8	0.2	0.7	200	3926	0

Продолжение таблицы 4.3

α	β	p	Количество дней	Результат	Ошибка
0.8	0.2	0.8	50	3926	1738
0.8	0.2	0.8	100	3926	1085
0.8	0.2	0.8	200	3926	1083
0.8	0.2	0.9	50	3926	1743
0.8	0.2	0.9	100	3926	2710
0.8	0.2	0.9	200	3926	2030
0.9	0.1	0.1	50	3926	3115
0.9	0.1	0.1	100	3926	1673
0.9	0.1	0.1	200	3926	584
0.9	0.1	0.2	50	3926	708
0.9	0.1	0.2	100	3926	2340
0.9	0.1	0.2	200	3926	847
0.9	0.1	0.3	50	3926	2508
0.9	0.1	0.3	100	3926	1313
0.9	0.1	0.3	200	3926	789

Продолжение таблицы 4.3

α	β	p	Количество дней	Результат	Ошибка
0.9	0.1	0.4	50	3926	3030
0.9	0.1	0.4	100	3926	2210
0.9	0.1	0.4	200	3926	789
0.9	0.1	0.5	50	3926	2850
0.9	0.1	0.5	100	3926	2773
0.9	0.1	0.5	200	3926	1660
0.9	0.1	0.6	50	3926	2288
0.9	0.1	0.6	100	3926	2217
0.9	0.1	0.6	200	3926	1660
0.9	0.1	0.7	50	3926	2350
0.9	0.1	0.7	100	3926	387
0.9	0.1	0.7	200	3926	2320
0.9	0.1	0.8	50	3926	2807
0.9	0.1	0.8	100	3926	1493
0.9	0.1	0.8	200	3926	1378

Продолжение таблицы 4.3

α	β	p	Количество дней	Результат	Ошибка
0.9	0.1	0.9	50	3926	2633
0.9	0.1	0.9	100	3926	1559
0.9	0.1	0.9	200	3926	1420

Вывод

В текущем разделе был проведен эксперимент по измерению времени работы двух алгортмов для решения задачи коммивояжера. Согласно полученным при проведении эксперимента данным, муравьиный алгоритм начинает работать в разы быстрее алгоритма полного перебора при количестве вершин больше 7. Также была проведена параметризация для муравьиного алгоритма на двух классах данных. Наиболее подходящие параметры для класса данных №1:

1) $\alpha = 0.1,\ \beta = 0.9,\ p = 0.3;$
2) $\alpha = 0.1,\ \beta = 0.9,\ p = 0.4;$
3) $\alpha = 0.2,\ \beta = 0.8,\ p = 0.5;$
4) $\alpha = 0.2,\ \beta = 0.8,\ p = 0.6;$
5) $\alpha = 0.2,\ \beta = 0.8,\ p = 0.8.$

Наиболее подходящие параметры для класса данных №2:

1) $\alpha = 0.1,\ \beta = 0.9,\ p = 0.1;$
2) $\alpha = 0.1,\ \beta = 0.9,\ p = 0.3;$
3) $\alpha = 0.1,\ \beta = 0.9,\ p = 0.6;$
4) $\alpha = 0.2,\ \beta = 0.8,\ p = 0.7.$

ЗАКЛЮЧЕНИЕ

В рамках данной лабораторной работы была достигнута поставленная цель: была изучена задача коммивояжера, а также были реализованы алгоритм полного перебора и муравьиный алгоритм для решения этой задачи.

Решены все поставленные задачи:

1) исследована задача коммивояжера;
2) реализован алгоритм полного перебора для решения задачи коммивояжера;
3) реализован муравьиный алгоритм для решения задачи коммивояжера;
4) проведена параметризация муравьиного алгоритма на двух классах данных;
5) проведен сравнительный анализ времени работы двух алгоритмов для решения задачи коммивояжера на основе экспериментальных данных.

В исследовательском разделе в ходе проведения эксперимента было подтверждено значительное отличие длительности выполнения реализаций алгоритма полного перебора и муравьиного алгоритма. Муравьиный алгоритм начинает работать быстрее алгоритма полного перебора при количестве вершин больше 7. Реализация муравьиного алгоритма при 9 вершинах по времени выполнения превосходит алгоритм полного перебора на 813%.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Computer tools for solving the traveling salesman problem / I. Brezina [и др.] // Development Management. — 2020. — С. 25—39.
2. *Штовба С. Д.* Муравьиные алгоритмы // Exponenta Pro. — 2003. — Т. 4. — С. 70—75.