



**Министерство науки и высшего образования Российской
Федерации**
**Федеральное государственное автономное 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»

**Лабораторная работа № 6
по дисциплине «Анализ алгоритмов»**

Тема Методы решения задачи коммивояжёра

Студент Куликов Н.В.

Группа ИУ7-53Б

Преподаватели Волкова Л.Л., Строганов Ю.В., Строганов Д.В.

Москва, 2025

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	4
1 Аналитическая часть	5
1.1 Ориентированный граф	5
1.2 Задача коммивояжёра	5
1.2.1 Метод полного перебора	5
1.2.2 Метод на основе муравьиного алгоритма	6
1.3 Вывод	6
2 Конструкторская часть	7
2.1 Функциональные требования	7
2.2 Разработка алгоритмов	7
2.2.1 Алгоритм полного перебора	7
2.2.2 Муравьиный алгоритм с элитными муравьями	9
2.3 Модель вычислений	10
2.4 Трудоёмкость алгоритмов	10
2.4.1 Трудоёмкость алгоритма полного перебора	10
2.4.2 Трудоёмкость муравьиного алгоритма	11
2.5 Вывод	12
3 Технологическая часть	13
3.1 Средства реализации	13
3.2 Реализация алгоритмов	13
3.3 Тестирование	17
3.4 Вывод	17

4	Исследовательская часть	18
4.1	Технические характеристики ЭВМ	18
4.2	Параметризация муравьиного алгоритма	18
4.2.1	Входные графы	18
4.2.2	Проведение параметризации	22
4.3	Вывод	22
	ЗАКЛЮЧЕНИЕ	23
	СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	24
	ПРИЛОЖЕНИЕ А	25

ВВЕДЕНИЕ

Целью данной лабораторной работы было исследование алгоритмов решения задачи коммивояжёра: алгоритма полного перебора, муравьиного алгоритма.

Для достижения поставленной цели необходимо было выполнить следующие задачи:

- 1) описать метод полного перебора для решения задачи коммивояжёра;
- 2) описать метод решения задачи коммивояжёра на основе муравьиного алгоритма;
- 3) реализовать оба алгоритма;
- 4) выполнить оценку трудоёмкости составленных алгоритмов по разработанным схемам алгоритмов;
- 5) выполнить параметризацию последнего метода;
- 6) дать рекомендации о значениях либо диапазонах значений параметров, настройках и/или режимах работы метода, основанного на муравьином алгоритме.

1 Аналитическая часть

1.1 Ориентированный граф

Оrientированный граф G задаётся двумя множествами (1.1)

$$G = (V, E), \quad (1.1)$$

где V — конечное множество, элементы которого называют вершинами или узлами, E — множество упорядоченных пар на V , то есть подмножество множества $V \times V$, элементы которого называют дугами [1].

Если дуга $e = (u, v)$, то говорят, что дуга e ведёт из вершины u в вершину v , и обозначают это $u \rightarrow v$ [1].

1.2 Задача коммивояжёра

Задача коммивояжёра — одна из самых известных задач комбинаторной оптимизации, заключающаяся в поиске самого выгодного маршрута, проходящего через указанные города ровно по одному разу с последующим возвратом в исходный город.

В данной лабораторной работе задача коммивояжёра подразумевает нахождение незамкнутого маршрута, то есть нужно посетить все города, закончить в любом другом городе.

1.2.1 Метод полного перебора

Метод полного перебора подразумевает рассмотрение всех возможных маршрутов путём перестановок вершин. Гарантируется, что на выходе будет самый короткий маршрут.

Достоинством данного метода является гарантия получения оптимального маршрута.

Недостатком данного метода является его алгоритмическая сложность $O(n!)$.

1.2.2 Метод на основе муравьиного алгоритма

Моделирование поведения муравьёв связано с распределением феромона на тропе — ребре графа в задаче коммивояжёра. При этом вероятность включения ребра в маршрут отдельного муравья пропорциональна количеству феромона на этом ребре, а количество откладываемого феромона пропорционально длине маршрута. Чем короче маршрут, тем больше феромона будет отложено на его рёбрах, следовательно, большее количество муравьёв будет включать его в синтез собственных маршрутов. Моделирование такого подхода, использующего только положительную обратную связь, приводит к преждевременной сходимости — большинство муравьёв двигается по локально оптимальному маршруту. Избежать этого можно, моделируя отрицательную обратную связь в виде испарения феромона. При этом если феромон испаряется быстро, то это приводит к потере памяти колонии и забыванию хороших решений, с другой стороны, большое время испарения может привести к получению устойчивого локально оптимального решения [2].

Муравьи имеют собственную «память». Поскольку каждый город может быть посещён только один раз, у каждого муравья есть список уже посещённых городов — список запретов.

Муравьи обладают «зрением» — видимость есть эвристическое желание посетить город j , если муравей находится в городе i .

Муравьи обладают «обонянием» — они могут улавливать след феромона, подтверждающий желание посетить город j из города i , на основании опыта других муравьёв.

Дополнительная модификация алгоритма может состоять во введении так называемых «элитных» муравьёв, которые усиливают ребра наилучшего маршрута, найденного с начала работы алгоритма [2].

1.3 Вывод

В аналитической части были рассмотрены понятие ориентированного графа, задача коммивояжёра и методы её решения.

2 Конструкторская часть

2.1 Функциональные требования

Ориентированный граф с элитными муравьями, карта для воздухоплавателей, время с учётом направления ветра, горы пролететь невозможно.

Входные данные:

- граф в виде матрицы смежности;
- t_{max} — время жизни колонии муравьёв (количество итераций);
- α — коэффициент влияния феромона;
- β — коэффициент влияния веса ребра;
- ρ — коэффициент испарения феромона;
- Q — квота феромона 1 муравья на 1 день;
- e — количество элитных муравьёв.

Выходные данные:

- оптимальный маршрут;
- длина оптимального маршрута.

2.2 Разработка алгоритмов

В данном разделе представлены схемы алгоритмов работы: полного перебора и муравьиного алгоритма с элитными муравьями.

2.2.1 Алгоритм полного перебора

На рисунке 2.1 представлена схема работы алгоритма полного перебора для решения задачи коммивояжёра.

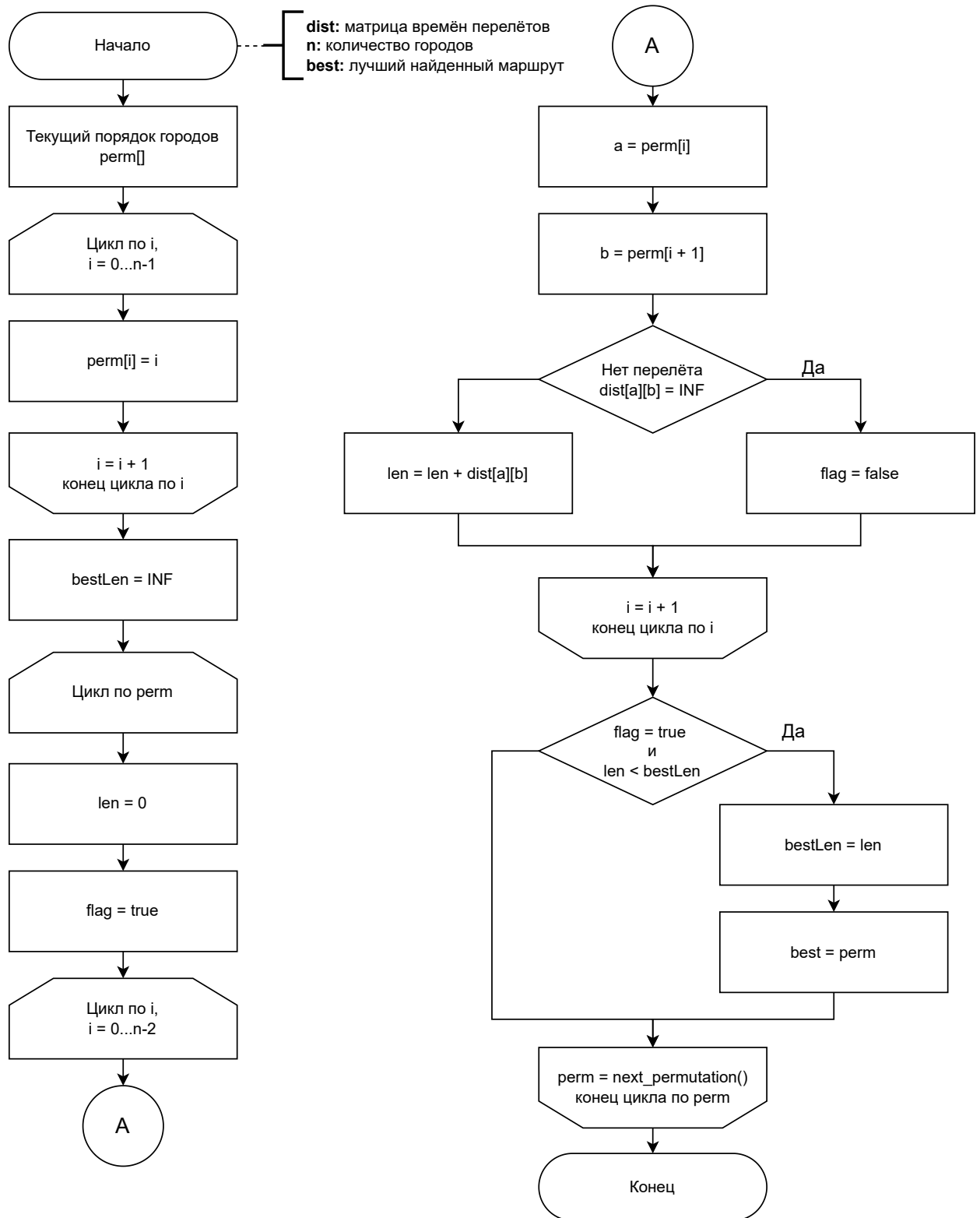


Рисунок 2.1 — Схема работы алгоритма полного перебора

2.2.2 Муравьиный алгоритм с элитными муравьями

На рисунке 2.2 представлена схема работы муравьиного алгоритма с элитными муравьями для решения задачи коммивояжёра.

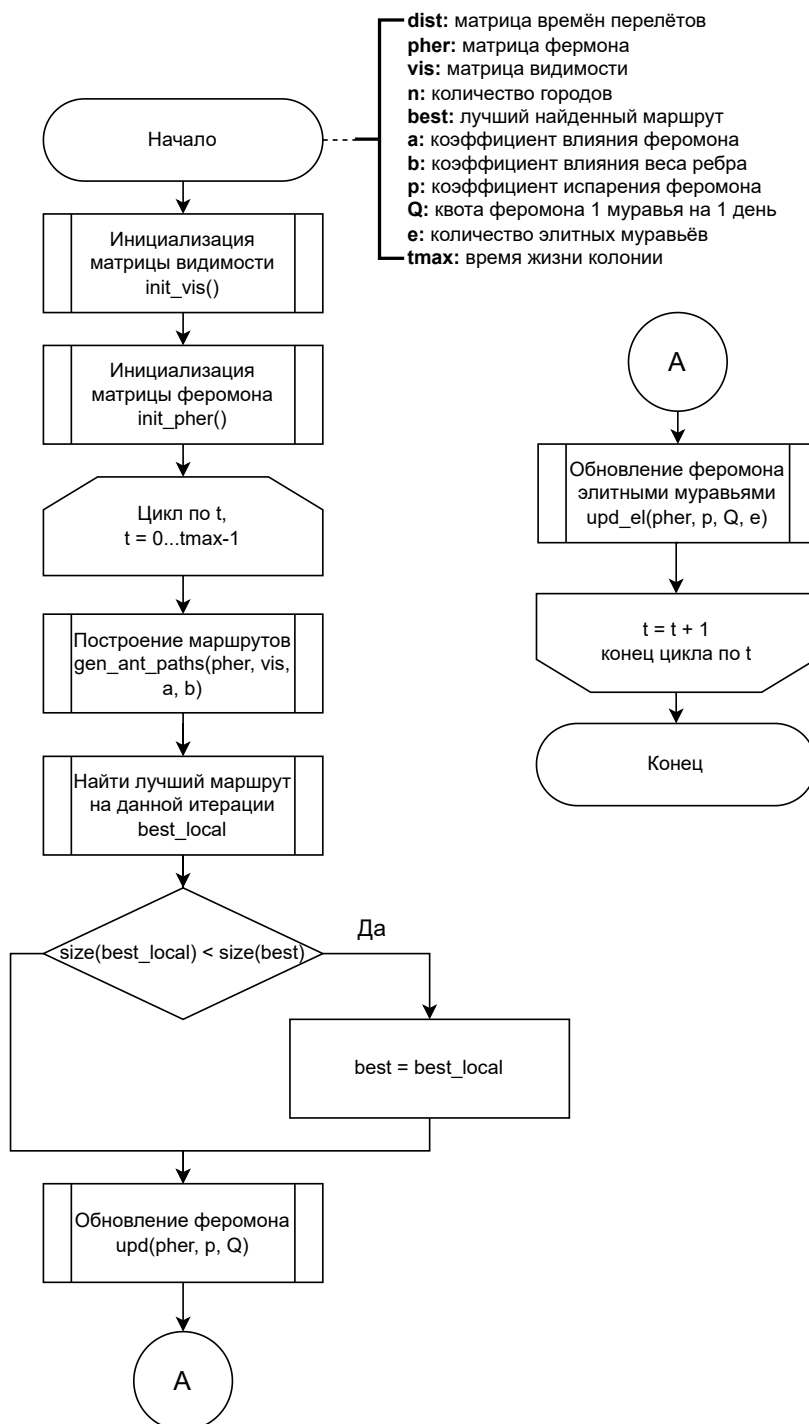


Рисунок 2.2 — Схема работы муравьиного алгоритма с элитными муравьями

2.3 Модель вычислений

Чтобы вычислить трудоёмкость алгоритмов, введена следующая модель вычислений:

1) базовые операции:

— трудоёмкость операций из списка 2.1 равна 1:

$$=, +, +=, -, -=, ++, --, ==, !=, <, <=, >=, >, [], \&\&, \&, >>, <<, ||, |; \quad (2.1)$$

— трудоёмкость операций из списка 2.2 равна 2:

$$*, *=, /, /=, \%, \% =; \quad (2.2)$$

2) трудоёмкость условного перехода равна 0;

3) трудоёмкость условного оператора по формуле (2.3):

$$f_{if} = f_{условия} + \begin{cases} \min(f_A, f_B) & \text{— лучший случай,} \\ \max(f_A, f_B) & \text{— худший случай;} \end{cases} \quad (2.3)$$

4) трудоёмкость цикла по формуле (2.4):

$$f_{for} = f_{инициализации} + f_{сравнения} + M \cdot (f_{тела} + f_{инкремента} + f_{сравнения}), \quad (2.4)$$

где M — число итераций.

2.4 Трудоёмкость алгоритмов

2.4.1 Трудоёмкость алгоритма полного перебора

Трудоёмкость алгоритма полного перебора для худшего случая рассчитана по формуле (2.5):

$$f = 2 + N \cdot (2 + 2) + 1 + N! \cdot (2 + 2 + (N - 1) \cdot (3 + 3 + 4 + 2) + 2 + 2 + 2) = 3 + 4N + 12N \cdot N! - 2 \cdot N!, \quad (2.5)$$

где N — количество вершин графа.

Асимптотическая оценка временной сложности алгоритма: $O(N \cdot N!)$.

2.4.2 Трудоёмкость муравьиного алгоритма

Пусть T — число итераций, N — количество вершин графа, $A = N$ — количество муравьёв.

Трудоёмкость муравьиного алгоритма рассчитана по формуле (2.6):

$$f = f_{initVis} + f_{initPher} + 2 + T \cdot (f_{path} + f_{findBest} + f_{upd} + 2), \quad (2.6)$$

где $f_{initVis}$ — трудоёмкость инициализации матрицы видимости, $f_{initPher}$ — трудоёмкость инициализации матрицы феромона, f_{path} — трудоёмкость построения маршрутов муравьями, $f_{findBest}$ — трудоёмкость нахождения лучшего маршрута, f_{upd} — трудоёмкость обновления матрицы феромона.

Трудоёмкость инициализации матрицы видимости рассчитана по формуле (2.7):

$$f_{initVis} = 2 + N \cdot (4 + 16N) = 2 + 4N + 16N^2. \quad (2.7)$$

Трудоёмкость инициализации матрицы феромона рассчитана по формуле (2.8):

$$f_{initPher} = 2 + N \cdot (4 + 12N) = 2 + 4N + 12N^2. \quad (2.8)$$

Трудоёмкость построения маршрутов муравьями рассчитана по формуле (2.9):

$$\begin{aligned} f_{path} &= 2 + N \cdot (11 + N \cdot (6 + N \cdot (8 + 3 + 3 + 6 + 6))) = \\ &= 2 + N \cdot (11 + 6N + 26N^2) = 2 + 11N + 6N^2 + 26N^3. \end{aligned} \quad (2.9)$$

Трудоёмкость нахождения лучшего маршрута рассчитана по формуле (2.10):

$$\begin{aligned} f_{findBest} &= 2 + 2 + N \cdot (2 + 2 + 1 + 2) + 2 + 2 + N \cdot (4 + 2) = \\ &= 8 + 13N. \end{aligned} \quad (2.10)$$

Трудоёмкость обновления матрицы феромона рассчитана по формуле (2.11):

$$\begin{aligned} f_{upd} = & 2 + N \cdot (2 + N \cdot (2 + 1 + 2 + 2) + 2) + 2 + N \cdot (2 + \\ & + 2 + 1 + 2 + 1 + 2 + (N - 1) \cdot (3 + 4 + 3 + 2) + 2) + 4 + 1 + \\ & + 2 + (N - 1) \cdot (2 + 3 + 3 + 2) = 1 + 14N + 19N^2. \end{aligned} \quad (2.11)$$

Общая трудоёмкость рассчитана по формуле (2.12):

$$\begin{aligned} f = & 2 + 4N + 16N^2 + 2 + 4N + 12N^2 + 2 + T \cdot (2 + 11N + 6N^2 + \\ & + 26N^3 + 8 + 13N + 1 + 14N + 19N^2 + 2) = 6 + 8N + 28N^2 + \\ & + 13T + 38TN + 25TN^2 + 26TN^3. \end{aligned} \quad (2.12)$$

Асимптотическая оценка временной сложности алгоритма: $O(T \cdot N^3)$.

2.5 Вывод

В данном разделе были описаны функциональные требования к программе, построены схемы работы алгоритмов решения задачи коммивояжёра: метода полного перебора и муравьиного алгоритма, выполнена оценка трудоёмкости алгоритмов.

3 Технологическая часть

3.1 Средства реализации

Для реализации алгоритмов был выбран язык C++17 [3]. Выбор обусловлен тем, что C++ имеет все необходимые стандартные библиотеки для решения задачи.

3.2 Реализация алгоритмов

В листингах 3.1 – 3.2 показаны реализации алгоритмов: полного перебора и муравьиного.

```

void solve_bruteforce_iterative() {
    int perm[MAX_N];
    for (int i = 0; i < n; ++i)
        perm[i] = i;
    bestLenBrute = INF;
    bool first = true;
    do {
        double len = 0.0;
        bool ok = true;
        for (int i = 0; i < n - 1; ++i) {
            int a = perm[i];
            int b = perm[i + 1];

            if (distM[a][b] >= INF / 2.0) {
                ok = false;
                break;
            }
            len += distM[a][b];
            if (len >= bestLenBrute) {
                ok = false;
                break;
            }
        }
        if (ok) {
            if (first || len < bestLenBrute) {
                first = false;
                bestLenBrute = len;
                memcpy(bestPathBrute, perm, n * sizeof(int));
            }
        }
    } while (std::next_permutation(perm, perm + n));
}

```

Листинг 3.1 — Реализация алгоритма полного перебора

```

void run_aco() {
    init_heuristic();
    init_pheromone();
    bestLenACO = INF;
    for (int iter = 0; iter < MAX_ITER; ++iter) {
        for (int k = 0; k < NUM_ANTS; ++k)
            generate_ant_solution(k);

        int bestAntIter = -1;
        double bestLenIter = INF;

        for (int k = 0; k < NUM_ANTS; ++k) {
            if (antLen[k] < bestLenIter) {
                bestLenIter = antLen[k];
                bestAntIter = k;
            }
        }

        if (bestLenIter < bestLenACO) {
            bestLenACO = bestLenIter;
            for (int i = 0; i < n; ++i)
                bestPathACO[i] = antPath[bestAntIter][i];
        }

        for (int i = 0; i < n; ++i)
            for (int j = 0; j < n; ++j)
                pheromone[i][j] *= (1.0 - RHO);

        for (int k = 0; k < NUM_ANTS; ++k) {
            if (antLen[k] >= INF / 2.0)
                continue;

            double contrib = QVAL / antLen[k];
            for (int s = 0; s < n - 1; ++s) {
                int i = antPath[k][s];
                int j = antPath[k][s + 1];
                pheromone[i][j] += contrib;
            }
        }
    }
}

```

```
if (bestLenACO < INF / 2.0) {  
    double eliteContrib = ELITE_ANTS * QVAL / bestLenACO;  
    for (int s = 0; s < n - 1; ++s) {  
        int i = bestPathACO[s];  
        int j = bestPathACO[s + 1];  
        pheromone[i][j] += eliteContrib;  
    }  
}  
}  
}
```

Листинг 3.2 — Реализация муравьиного алгоритма

3.3 Тестирование

В таблице 3.1 представлены результаты тестирования реализации алгоритма полного перебора. Все тесты пройдены успешно.

Таблица 3.1 — Результаты тестирования алгоритма полного перебора

№	Входной граф (матрица смежности)	Ожидаемый вывод	Фактический вывод
1	$\begin{pmatrix} 0 & 1 & 10 \\ 10 & 0 & 1 \\ 10 & 10 & 0 \end{pmatrix}$	Маршрут: $0 \rightarrow 1 \rightarrow 2$, $L_{\text{best}} = 2$.	Маршрут: $0 \rightarrow 1 \rightarrow 2$, $L_{\text{best}} = 2$.
2	$\begin{pmatrix} 0 & 1 & \infty & \infty \\ 5 & 0 & 1 & \infty \\ \infty & 5 & 0 & 1 \\ \infty & \infty & 5 & 0 \end{pmatrix}$	Маршрут: $0 \rightarrow 1 \rightarrow 2 \rightarrow 3$, $L_{\text{best}} = 3$.	Маршрут: $0 \rightarrow 1 \rightarrow 2 \rightarrow 3$, $L_{\text{best}} = 3$.
3	(0)	Маршрут: 0, $L_{\text{best}} = 0$.	Маршрут: 0, $L_{\text{best}} = 0$.
4	$\begin{pmatrix} 0 & 1 & \infty \\ 1 & 0 & \infty \\ \infty & \infty & 0 \end{pmatrix}$	Незамкнутого маршрута не существует, $L_{\text{best}} = \infty$.	Незамкнутого маршрута не существует, $L_{\text{best}} = \infty$.

3.4 Вывод

В этом разделе были описаны средства реализации алгоритмов. Также были продемонстрированы листинги реализаций: полного перебора и муравьиного алгоритма. Приведены результаты тестирования.

4 Исследовательская часть

4.1 Технические характеристики ЭВМ

Исследование проводилось на ноутбуке ACER Predator со следующими техническими характеристиками:

- процессор Intel(R) Core(TM) i7-10750H с тактовой частотой 2.60ГГц;
- ОЗУ 16 ГБ;
- ОС Windows 10 Pro 64 разрядная.

Во время исследования ноутбук был подключён к электропитанию, сторонними приложениями нагружен не был.

4.2 Параметризация муравьиного алгоритма

4.2.1 Входные графы

Было построено три ориентированных графа в виде матриц смежности по реальным расстояниям между городами на картах Нидерландов [4], Швейцарии [5] и Австрии [6].

В выражении 4.1 содержится матрица смежности ориентированного графа, построенная на основе карты городов Нидерландов.

$$\begin{pmatrix}
 0 & 20 & 45 & 60 & 75 & 40 & 90 & 120 & 110 & 185 \\
 20 & 0 & 30 & 45 & 60 & 55 & 95 & 125 & 115 & 200 \\
 45 & 30 & 0 & 25 & 35 & 50 & 80 & 110 & 105 & 210 \\
 60 & 45 & 25 & 0 & 25 & 60 & 90 & 115 & 120 & 220 \\
 75 & 60 & 35 & 25 & 0 & 55 & 80 & 100 & 105 & 225 \\
 40 & 55 & 50 & 60 & 55 & 0 & 50 & 75 & 65 & 180 \\
 90 & 95 & 80 & 90 & 80 & 50 & 0 & 35 & 55 & 195 \\
 120 & 125 & 110 & 115 & 100 & 75 & 35 & 0 & 60 & 215 \\
 110 & 115 & 105 & 120 & 105 & 65 & 55 & 60 & 0 & 170 \\
 185 & 200 & 210 & 220 & 225 & 180 & 195 & 215 & 170 & 0
 \end{pmatrix} \quad (4.1)$$

Нумерация вершин:

- 1) Amsterdam;
- 2) Haarlem;
- 3) Leiden;
- 4) Den Haag;
- 5) Rotterdam;
- 6) Utrecht;
- 7) Hertogenbosch;
- 8) Eindhoven;
- 9) Nijmegen;
- 10) Groningen.

Минимальная длина незамкнутого маршрута равна 455.

В выражении 4.2 содержится матрица смежности ориентированного графа, построенная на основе карты городов Швейцарии.

$$\begin{pmatrix}
 0 & 48 & 128 & \infty & 208 & \infty & \infty & \infty & \infty & \infty \\
 72 & 0 & 64 & 128 & 168 & 112 & \infty & \infty & \infty & \infty \\
 192 & 96 & 0 & 72 & 96 & 80 & \infty & \infty & \infty & 60 \\
 \infty & 192 & 108 & 0 & 64 & 88 & 104 & \infty & \infty & \infty \\
 312 & 252 & 144 & 96 & 0 & 60 & 72 & 96 & \infty & \infty \\
 \infty & 168 & 120 & 132 & 40 & 0 & \infty & 88 & 120 & 84 \\
 \infty & \infty & \infty & 156 & 108 & \infty & 0 & 85 & \infty & \infty \\
 \infty & \infty & \infty & \infty & 144 & 132 & 85 & 0 & 150 & \infty \\
 \infty & \infty & \infty & \infty & \infty & 180 & \infty & 150 & 0 & \infty \\
 \infty & \infty & 60 & \infty & \infty & 56 & \infty & \infty & \infty & 0
 \end{pmatrix} \quad (4.2)$$

Нумерация вершин:

- 1) Geneva;
- 2) Lausanne;
- 3) Bern;
- 4) Basel;
- 5) Zurich;
- 6) Luzern;
- 7) St. Gallen;
- 8) Chur;
- 9) Lugano;
- 10) Interlaken.

Минимальная длина незамкнутого маршрута равна 703.

В выражении 4.3 содержится матрица смежности ориентированного графа, построенная на основе карты городов Австрии.

$$\begin{pmatrix}
 0 & 72 & \infty & \infty & \infty & \infty & 200 & \infty & \infty & 60 \\
 48 & 0 & 144 & \infty & \infty & \infty & \infty & \infty & \infty & \infty \\
 \infty & 96 & 0 & 156 & \infty & \infty & 152 & 210 & \infty & \infty \\
 \infty & \infty & 104 & 0 & 168 & \infty & 176 & 160 & \infty & \infty \\
 \infty & \infty & \infty & 112 & 0 & 216 & \infty & 184 & \infty & \infty \\
 \infty & \infty & \infty & \infty & 144 & 0 & \infty & \infty & \infty & \infty \\
 200 & \infty & 228 & 264 & \infty & \infty & 0 & 168 & \infty & 128 \\
 \infty & \infty & 210 & 240 & 276 & \infty & 112 & 0 & 72 & 144 \\
 \infty & \infty & \infty & \infty & \infty & \infty & \infty & 48 & 0 & \infty \\
 60 & \infty & \infty & \infty & \infty & \infty & 192 & 216 & \infty & 0
 \end{pmatrix} \quad (4.3)$$

Нумерация вершин:

- 1) Wien;
- 2) St. Polten;
- 3) Linz;
- 4) Salzburg;
- 5) Innsbruck;
- 6) Bregenz;
- 7) Graz;
- 8) Klagenfurt;
- 9) Villach;
- 10) Eisenstadt.

Минимальная длина незамкнутого маршрута равна 996.

4.2.2 Проведение параметризации

Была проведена параметризация муравьиного алгоритма с элитными муравьями по параметрам:

- 1) t_{max} — количество итераций;
- 2) α — коэффициент влияния феромона;
- 3) ρ — коэффициент испарения феромона.

Для построения таблиц были выбраны следующие сокращения:

- 1) \max — максимальное отклонение;
- 2) avg — среднее арифметическое отклонение;
- 3) med — медианное отклонение.

В приложении А в таблицах А.1 – А.2 представлены результаты проведённой параметризации.

Можно сформулировать следующие рекомендации о выборе параметров:

- $t_{max} \in (50, 100)$;
- $\alpha \in \{1\}$;
- $\rho \in (0.1, 0.4)$.

В таблице результатов параметризации иногда появляются очень большие отклонения порядка $\delta \approx 10^9$ это означает то, что в этих запусках муравьиный алгоритм вообще не нашёл допустимый маршрут. Такие провалы появляются, если ρ близко к 1, так как предыдущий опыт предыдущих муравьёв практически полностью стирается.

4.3 Вывод

В данном разделе были описаны технические характеристики машины, на которой проводилось исследование. Описаны входные графы. Проведена параметризация муравьиного алгоритма с элитными муравьями, на основе результатов которой даны рекомендации о значениях либо диапазонах значений параметров.

ЗАКЛЮЧЕНИЕ

В результате лабораторной работы были исследованы алгоритмы решения задачи коммивояжёра: алгоритм полного перебора, муравьиный алгоритм.

Выполнены следующие задачи:

- 1) описан метод полного перебора для решения задачи коммивояжёра;
- 2) описан метод решения задачи коммивояжёра на основе муравьиного алгоритма;
- 3) реализованы оба алгоритма;
- 4) выполнена оценка трудоёмкости составленных алгоритмов по разработанным схемам алгоритмов;
- 5) выполнена параметризация последнего метода;
- 6) даны рекомендации о значениях либо диапазонах значений параметров, настройках и/или режимах работы метода, основанного на муравьином алгоритме ($t_{max} \in (50, 100)$, $\alpha \in \{1\}$, $\rho \in (0.1, 0.4)$).

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Белоусов И.В., Ткачев С.Б. Дискретная математика. — Москва: Издательство МГТУ им. Н.Э. Баумана, 2004. — С. 276 – 277.
2. Ульянов М.В. Ресурсно–эффективные компьютерные алгоритмы. Разработка и анализ. — Москва: Изд-во Наука, 2007. — С. 203 – 206.
3. ISO/IEC JTC1 SC22 WG21 N4860. 2018.
4. Большая карта регионов Нидерландов. [Электронный ресурс]. — Режим доступа: <https://www.maps-of-the-world.org/europe/netherlands>. (Дата обращения: 2025-11-29).
5. Большая политическая карта Швейцарии. [Электронный ресурс]. — Режим доступа: <https://www.maps-of-the-world.org/europe/switzerland>. (Дата обращения: 2025-11-29).
6. Большая политическая карта Австрии. [Электронный ресурс]. — Режим доступа: <https://www.maps-of-the-world.org/europe/austria> (Дата обращения: 2025-11-29).

ПРИЛОЖЕНИЕ А

Таблица А.1 — Результаты параметризации муравьиного алгоритма (часть 1)

Параметры			Граф 1			Граф 2			Граф 3		
t_{\max}	α	ρ	max	avg	med	max	avg	med	max	avg	med
10	0.20	0.10	155.0	116.0	110.0	152.0	85.5	88.0	148.0	89.2	108.0
10	0.20	0.20	135.0	90.5	92.5	188.0	110.4	138.0	156.0	94.0	108.0
10	0.20	0.40	135.0	103.5	110.0	264.0	116.0	94.0	272.0	104.2	108.0
10	0.20	0.80	115.0	75.0	70.0	224.0	136.3	123.5	344.0	122.0	108.0
10	0.20	1.00	190.0	96.5	77.5	999999297.0	100000156.5	277.5	999999004.0	499999624.0	499999674.0
10	0.50	0.10	125.0	88.0	87.5	167.0	90.2	94.0	122.0	67.6	108.0
10	0.50	0.20	115.0	84.0	90.0	172.0	109.1	122.0	122.0	55.4	54.0
10	0.50	0.40	105.0	75.5	80.0	180.0	89.6	98.0	108.0	54.0	54.0
10	0.50	0.80	95.0	54.5	60.0	186.0	108.9	121.5	156.0	82.0	108.0
10	0.50	1.00	105.0	58.0	65.0	336.0	289.5	303.5	999999004.0	499999628.2	499999674.0
10	1.00	0.10	95.0	66.0	70.0	164.0	67.2	88.0	108.0	43.2	0.0
10	1.00	0.20	95.0	63.5	62.5	136.0	51.2	44.0	108.0	32.4	0.0
10	1.00	0.40	90.0	59.0	60.0	160.0	100.4	126.0	122.0	44.6	0.0
10	1.00	0.80	85.0	61.0	60.0	236.0	85.6	44.0	108.0	64.8	108.0
10	1.00	1.00	80.0	61.0	62.5	139.0	77.9	92.0	122.0	44.6	0.0
50	0.20	0.10	105.0	48.0	50.0	131.0	33.5	36.0	108.0	32.4	0.0
50	0.20	0.20	75.0	29.5	30.0	108.0	48.8	36.0	156.0	91.2	108.0
50	0.20	0.40	55.0	26.0	27.5	152.0	44.0	28.0	156.0	56.6	54.0
50	0.20	0.80	40.0	21.0	22.5	188.0	83.9	100.0	156.0	77.4	108.0
50	0.20	1.00	165.0	110.5	107.5	999999297.0	100000178.5	280.0	999999004.0	499999599.4	499999674.0
50	0.50	0.10	80.0	46.5	57.5	44.0	11.6	0.0	122.0	44.6	0.0
50	0.50	0.20	70.0	34.0	40.0	108.0	31.2	14.0	108.0	21.6	0.0
50	0.50	0.40	50.0	32.5	32.5	84.0	18.4	0.0	124.0	77.2	108.0
50	0.50	0.80	55.0	26.5	27.5	131.0	35.1	22.0	156.0	75.8	108.0
50	0.50	1.00	145.0	83.0	77.5	619.0	299.4	277.5	999999004.0	599999501.6	999999004.0

Таблица А.2 — Результаты параметризации муравьиного алгоритма (часть 2)

Параметры			Граф 1			Граф 2			Граф 3		
t_{\max}	α	ρ	max	avg	med	max	avg	med	max	avg	med
50	1.00	0.10	80.0	33.5	30.0	116.0	16.0	0.0	0.0	0.0	0.0
50	1.00	0.20	55.0	26.5	22.5	44.0	12.8	0.0	0.0	0.0	0.0
50	1.00	0.40	70.0	32.0	30.0	44.0	12.8	0.0	0.0	0.0	0.0
50	1.00	0.80	60.0	38.5	40.0	28.0	5.6	0.0	108.0	10.8	0.0
50	1.00	1.00	55.0	36.0	40.0	115.0	33.5	44.0	0.0	0.0	0.0
100	0.20	0.10	25.0	13.0	17.5	44.0	7.2	0.0	108.0	54.0	54.0
100	0.20	0.20	50.0	17.5	12.5	132.0	57.5	44.0	156.0	60.2	54.0
100	0.20	0.40	30.0	15.5	17.5	131.0	53.5	44.0	156.0	68.4	54.0
100	0.20	0.80	60.0	23.5	17.5	183.0	77.7	88.0	156.0	71.0	108.0
100	0.20	1.00	150.0	104.0	95.0	999999297.0	100000181.7	317.0	999999004.0	100000031.6	108.0
100	0.50	0.10	50.0	14.5	10.0	44.0	8.8	0.0	108.0	10.8	0.0
100	0.50	0.20	30.0	12.0	15.0	0.0	0.0	0.0	108.0	21.6	0.0
100	0.50	0.40	50.0	16.0	12.5	44.0	8.8	0.0	156.0	69.6	108.0
100	0.50	0.80	30.0	11.5	15.0	131.0	35.1	44.0	108.0	64.8	108.0
100	0.50	1.00	155.0	87.0	82.5	999999297.0	100000150.0	246.0	999999004.0	499999601.2	499999655.0
100	1.00	0.10	55.0	9.5	12.5	0.0	0.0	0.0	0.0	0.0	0.0
100	1.00	0.20	35.0	22.0	27.5	84.0	12.8	0.0	0.0	0.0	0.0
100	1.00	0.40	35.0	15.0	22.5	0.0	0.0	0.0	0.0	0.0	0.0
100	1.00	0.80	50.0	19.5	17.5	0.0	0.0	0.0	0.0	0.0	0.0
100	1.00	1.00	45.0	25.0	27.5	44.0	4.4	0.0	0.0	0.0	0.0