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

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

Тема Задача коммивояжера

Студент Жабин Д.В.

Группа ИУ7-54Б

Преподаватель Волкова Л.Л.

Москва, 2021 г.

Содержание

Введение

Задача коммивояжера (или TSP — Travelling Salesman Problem) формулируется как задача поиска минимального по стоимости замкнутого маршрута по всем вершинам без повторений на полном взвешенном графе с конечным количеством вершин [4]. Вершины графа являются городами, которые нужно посетить, а веса рёбер отражают длины или стоимости проезда.

Эта задача имеет множество решений. Одно из них — точный переборный алгоритм, который имеет факториальную сложность.

Ещё одним решением этой задачи является моделирование поведения муравьев. Оно связано с распределением феромона на тропе – ребре графа. 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 испарение 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 локального оптимального решения.

Целью лабораторной работы является получение навыка решения задачи коммивояжера. Для её достижения поставлены следующие задачи:

- изучить алгоритм полного перебора и муравьиный алгоритм для решения задачи коммивояжера;
- разработать схемы этих алгоритмов на основе изученной информации;
- реализовать изученные алгоритмы;
- провести тестирование разработанного программного обеспечения;
- провести анализ скорости работы реализаций алгоритмов.

1 Аналитическая часть

Рассмотрим ключевые особенности алгоритма полного перебора и муравьиного алгоритма для решения задачи коммивояжера.

1.1 Алгоритм полного перебора

Данный алгоритм также называется алгоритмом грубой силы. Его суть заключается в полном переборе всех возможных путей, при этом будут посещены все города и произойдет возврат в начальное положение.

Так, при количестве городов $n = 2$ результатом работы алгоритма является удвоенная длина пути из одного города в другой. При $n = 3$ — сумма длин всех 3 путей. При $n > 3$ нужно перебирать все возможные пути, удовлетворяющие вышеуказанным условиям — например, начиная с первого города: $1 \to 2 \to 3 \to 4 \to 1$, начиная со второго города: $2 \to 3 \to 4 \to 1 \to 2$ и так далее.

1.2 Муравьиный алгоритм

С учетом условий задачи коммивояжера можно описать поведение муравьев при выборе пути.

1. Поскольку каждый город может быть посещён только один раз, то для каждого муравья есть список уже посещённых городов — список запретов. Пусть $J_{i,k}$ — список городов, которые необходимо посетить муравью k, который находится в городе i.

2. Существует видимость — желание посетить город j, находясь в городе i. Пусть видимость обратно пропорциональна расстоянию D_{ij} между городами i и j (1).

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

3. Муравьи могут улавливать след феромона, подтверждающий желание посетить город j из города i на основании опыта других муравьёв. Количество феромона на ребре (i, j) в момент времени t равно $\tau_{ij}(t)$.

4. Вероятностно-пропорциональное правило, определяющее вероятность перехода муравья k из города i в город j (2).

$$P_{ij,\,k}(t) = \begin{cases} \dfrac{\tau_{ij}^{\alpha}(t) * \eta_{ij}^{\beta}}{\sum\limits_{l \in J_{i,k}} \tau_{il}^{\alpha}(t) * \eta_{il}^{\beta}}, & j \in J_{i,k}; \\ 0, & j \notin J_{i,k}. \end{cases} \tag{2}$$

В формуле (2) α и β — параметры, задающие веса следа феромона. При $\alpha = 0$ алгоритм вырождается до жадного алгоритма (будет выбран ближайший город). Можно заметить, что выбор города является вероятностным, правило (2) лишь определяет ширину зоны города j; в общую зону всех городов $J_{i,k}$ бросается случайное число, которое и определяет выбор муравья. Правило (2) не изменяется в ходе алгоритма, но у двух разных муравьёв значения вероятности перехода будут отличаться, так как они имеют разный список разрешённых городов.

5. Пройдя ребро (i, j), муравей откладывает на нём некоторое количество феромона, которое должно быть связано с оптимальностью сделанного выбора. Пусть $T_k(t)$ есть маршрут, пройденный муравьём k к моменту времени t, $L_k(t)$ — длина этого маршрута, а Q — параметр, имеющий значение порядка длины оптимального пути. Тогда откладываемое количество феромона может быть задано в виде (3).

$$\Delta\tau_{ij,\,k}(t) = \begin{cases} \frac{Q}{L_k(t)}, & (i,j) \notin T_k(t); \\ 0, & (i,j) \notin T_k(t). \end{cases} \tag{3}$$

6. Правила внешней среды определяют, в первую очередь, испарение феромона. Пусть $p \in [0, 1]$ есть коэффициент испарения, тогда правило испарения имеет вид (4).

$$\tau_{ij}(t+1) = (1-p) * \tau_{ij}(t) + \sum_{k=1}^{m} \Delta\tau_{ij,\,k}(t), \tag{4}$$

где m — количество муравьев в колонии.

7. В начале алгоритма количества феромона на рёбрах принимается равным небольшому положительному числу. Общее количество муравьёв остаётся постоянным и равным количеству городов, каждый муравей начинает маршрут из своего города.

1.3 Вывод по аналитической части

В данном разделе были рассмотрены ключевые особенности алгоритма грубой силы и муравьиного алгоритма для решения задачи коммивояжера.

2 Конструкторская часть

На основе полученных аналитических данных представим схемы выбранных алгоритмов решения задачи коммивояжера, опишем используемые типы и структуры данных, разработаем тесты для проверки корректности работы программы.

2.1 Схемы алгоритмов

На рисунках 2.1 – 2.4 представлены схемы алгоритма полного перебора и муравьиного алгоритма.

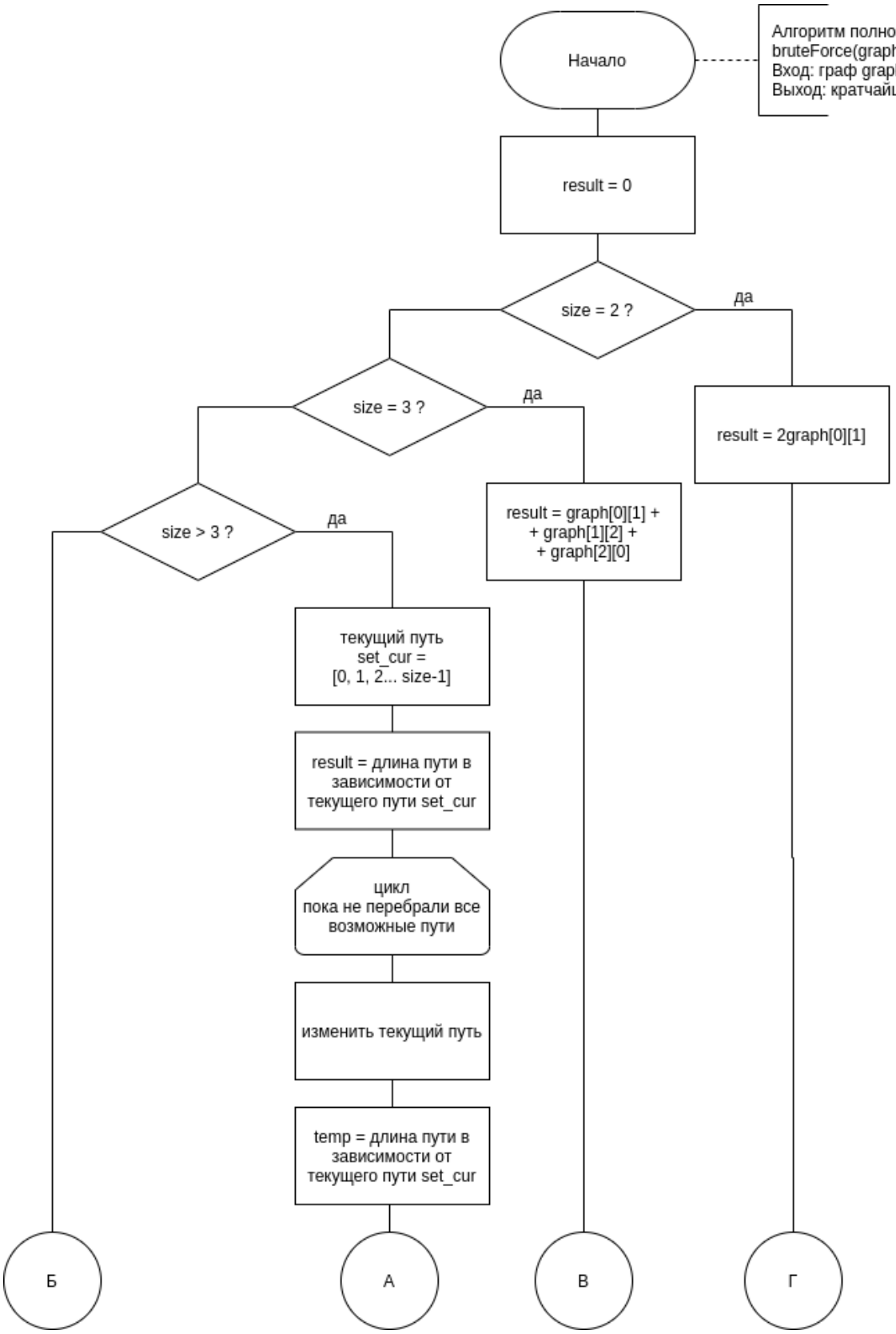

Рисунок 2.1 — Алгоритм полного перебора. Часть 1

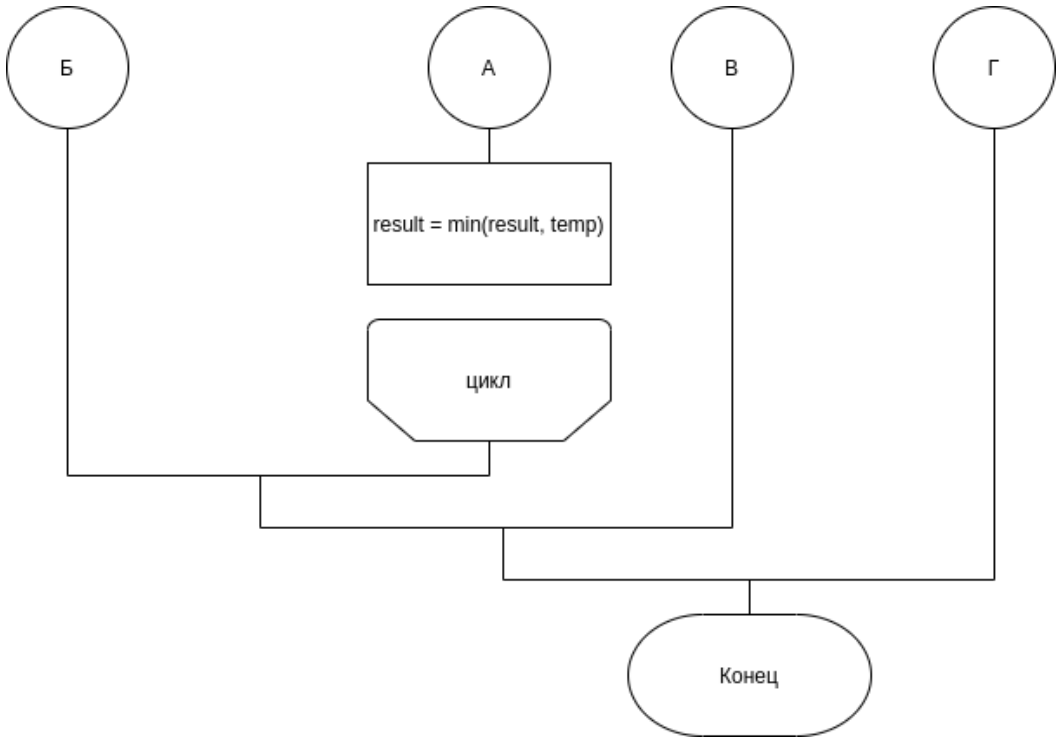

Рисунок 2.2 — Алгоритм полного перебора. Часть 2

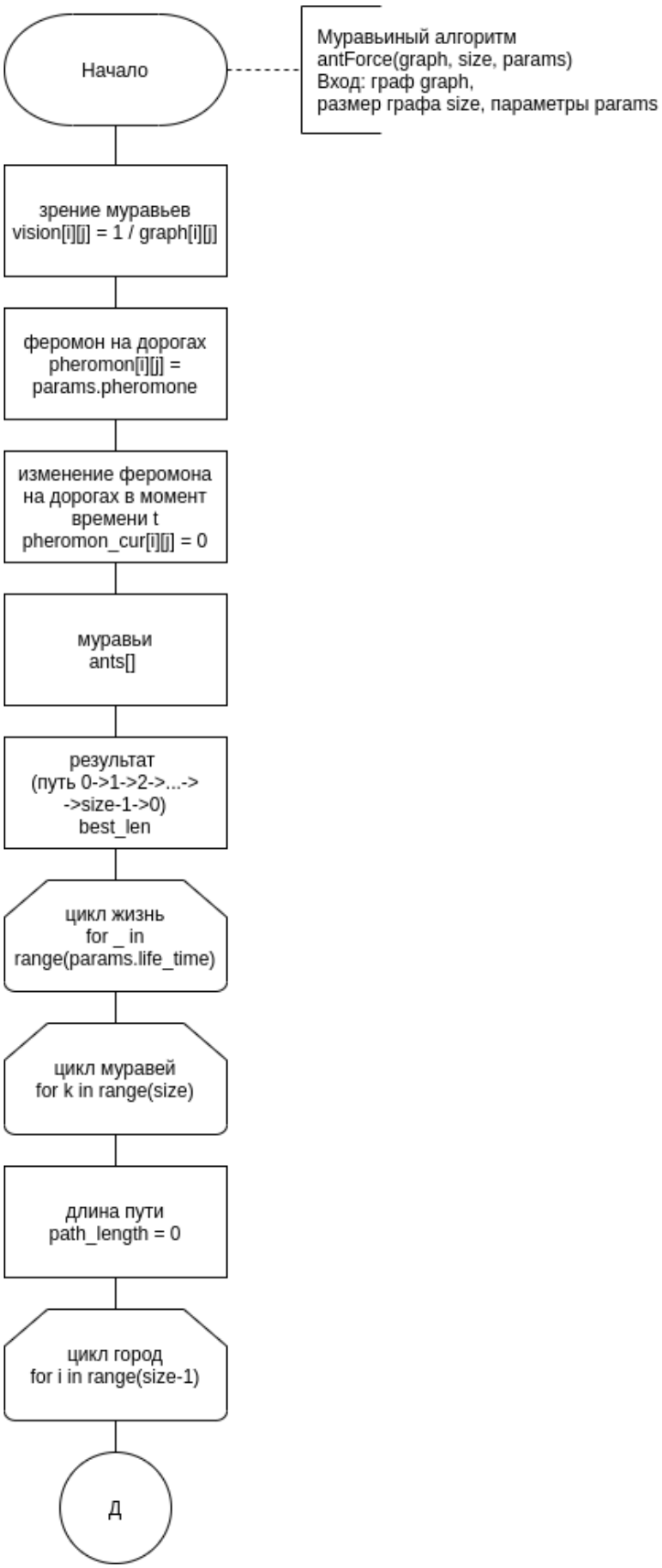

Рисунок 2.3 — Муравьиный алгоритм. Часть 1

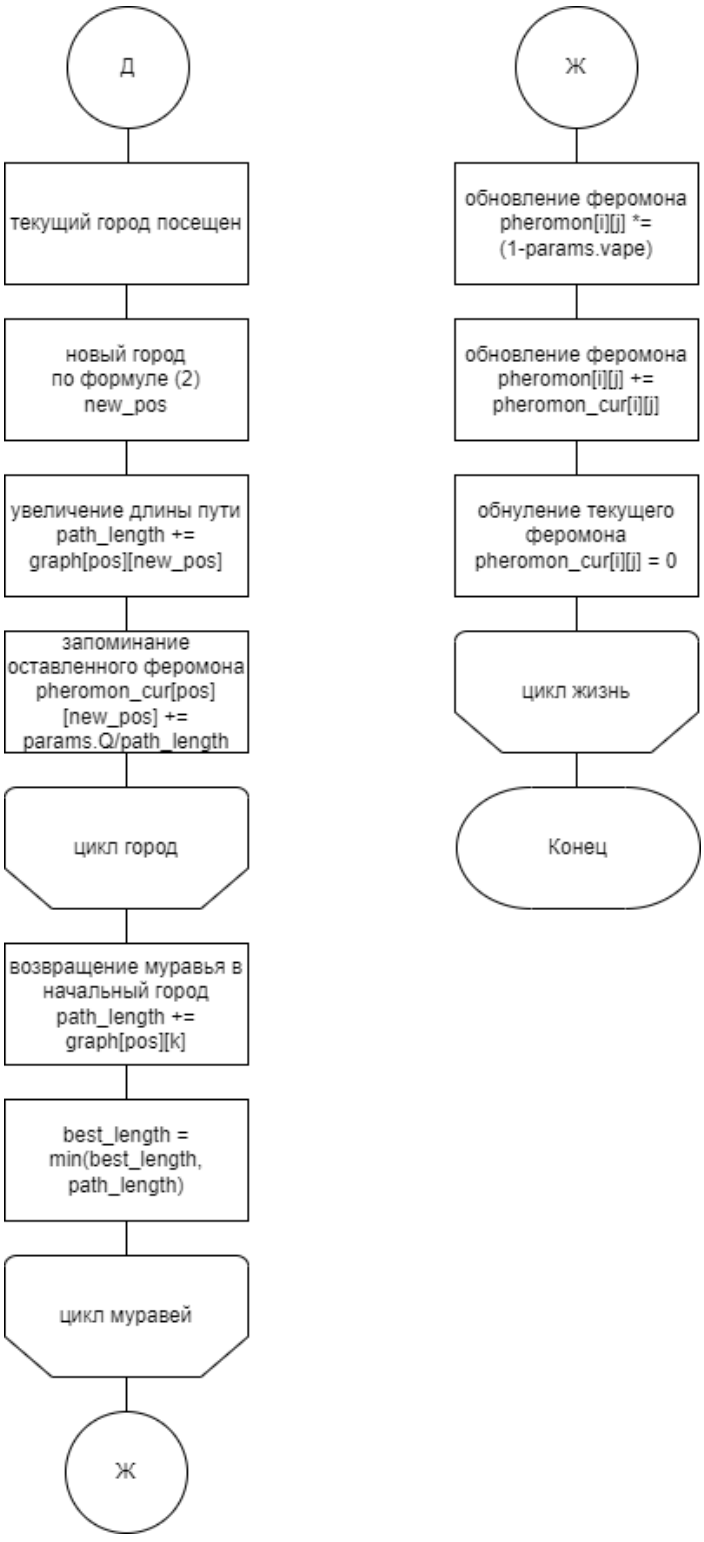

Рисунок 2.4 — Муравьиный алгоритм. Часть 2

2.2 Типы и структуры данных

Для хранения графа используется квадратная матрица размером $n \times n$, где n — количество городов. Тип матрицы — целочисленная. Несмотря на то, что достаточно было бы использования верхнетреугольной или нижне-

треугольной матрицы, используется полная для удобства доступа к любой длине пути.

При реализации муравьиного алгоритма следует объединить входящие параметры алгоритма в отдельную структуру для минимизации количества явных передаваемых параметров в функцию, реализующую муравьиный алгоритм. Так, структура `Params` содержит такие поля: α, β, длина оптимального пути, коэффициент испарения феромона, начальное значение феромона, длина жизни колонии муравьев.

Также использование структуры `Ant` позволит повысить читаемость исходного кода и уменьшит количество ошибок, связанных со списком городов, которые остается посетить каждому муравью. В этой структуре выделены следующие поля: текущая позиция, список непосещенных городов — массив типа `bool`.

2.3 Способ тестирования

Создаваемое программное обеспечение будет протестировано методом `черного ящика`. Для тестирования были выделены следующие классы эквивалентности.

1. Неверный выбор режима работы программы — пустой ввод, нецифровой символ или вещественное число.
2. Верный выбор режима работы программы — цифра из диапазона [1..4] или целое число, выходящее за пределы указанного диапазона.
3. Неверный ввод размера исходного графа — пустой ввод, нецифровой символ, целое неположительное число, единица или вещественное число.
4. Верный ввод размера исходного графа — целое положительное число, большее единицы.
5. Неверный ввод длины пути между городами — пустой ввод, нецифровой символ, целое неположительное или вещественное число.
6. Верный ввод длины пути между городами — целое положительное число.

2.4 Тестовые данные

В таблицах 2.1 – 2.3 представлены тестовые данные.

Таблица 2.1 — Функциональные тесты выбора режима работы программы

Номер тестового случая	Ввод	Ожидаемый результат
1	_[1]	Неверный ввод
2	α	Неверный ввод
3	3.14	Неверный ввод
4	1	Введите размер графа
5	2	Введите размер графа
6	3	Таблица замеров времени
7	4	Выход
8	4	.[2]

Таблица 2.2 — Функциональные тесты ввода размера исходного графа

Номер тестового случая	Ввод	Ожидаемый результат
1	_[1]	Неверный ввод
2	α	Неверный ввод
3	-3	Неверный ввод
4	1	Неверный ввод
5	3.14	Неверный ввод
6	2	Введите длины путей

Таблица 2.3 — Функциональные тесты ввода длин путей между городами

Номер тестового случая	Ввод	Ожидаемый результат
1	_[1]	Неверный ввод
2	α	Неверный ввод
3	2.2	Неверный ввод
4	0	Неверный ввод
5	4	...[3]

2.5 Структура программного обеспечения

Программное обеспечение разработано с использованием объектно-ориентированного подхода. Программа содержит функции, отвечающие за

[1]Пустой ввод

[2]Конец работы программы

[3]Ожидание ввода следующего элемента

решение задачи коммивояжера. Функция `bruteForce` решает данную задачу методом полного перебора всех возможных замкнутых путей. Она принимает на вход исходный граф и его размер. Функция `antForce` решает задачу коммивояжера с помощью муравьиного алгоритма. Она помимо исходного графа и его размера принимает на вход все параметры, необходимые для реализации алгоритма. Обе функции возвращают целое число — длину кратчайшего пути, при котором будут посещены все города и конечной точкой будет начальный город — ответ к задаче коммивояжера.

2.6 Вывод по конструкторской части

В данном разделе были разработаны схемы алгоритмов для решения задачи коммивояжера: алгоритма полного перебора и муравьиного алгоритма, а также подготовлены тестовые данные для программного обеспечения.

3 Технологическая часть

В данном разделе приведены средства реализации и листинги кода, а также перечислены требования к разрабатываемому программному обеспечению.

3.1 Требования к ПО

К программе предъявляется ряд требований:

- требования ко вводу
  - должен быть выбран режим работы программы (целое число в диапазоне [1..4]);
  - должны быть введены следующие значения: размерность графа (целое число, большее 1), длины путей между городами (натуральные числа), при необходимости: параметры α, p, начальное количество феромона на ребрах графа (вещественные числа в диапазоне $[0, 1]$), параметр t_{max} — продолжительность жизни колонии муравьев (натуральное число).
- требования к выводу
  - найденный кратчайший путь;
  - таблица времен работы и точности реализаций алгоритмов (точность алгоритма грубой силы эталонна) на случайных значениях.
- ограничения работы программы
  - при вводе режима работы программы, выходящего за пределы указанного диапазона, программа сообщает об ошибке ввода режима работы.
- функциональные требования к программному обеспечению
  - при запуске программа должна выводить меню с возможными режимами работы;
  - программа должна решать задачу коммивояжера одним из двух вышеуказанных алгоритмов;

– в исследовательском режиме программа должна замерять время работы реализаций обоих алгоритмов, а также оценивать точность решения задачи коммивояжера муравьиным алгоритмом при случайных длинах путей между городами.

3.2 Средства реализации

Для реализации ПО был выбран язык программирования `Python` [1]. Это обусловлено знанием возможностей языка, что обеспечит высокую скорость написания программы без потери ее качества.

В качестве среды разработки была выбрана `Visual Studio Code` [3]. Достаточный опыт работы в этой среде, удобства написания кода и его автодополнения стали ключевыми при выборе.

3.3 Реализация алгоритмов

В листингах 3.1 – 3.2 приведены реализации алгоритмов, использовавшихся в программном обеспечении.

Листинг 3.1 — Алгоритм полного перебора

```
def bruteForce(graph, size):
  result = 0
  if size == 2:
    result = 2*graph[0][1]
  elif size == 3:
    result = graph[0][1] + graph[0][2] + graph[1][2]
  elif size > 3:
    set_cur = [i for i in range(size)]
    result = lenBySet(graph, set_cur, size)
    while nextSet(set_cur, size):
      temp = lenBySet(graph, set_cur, size)
      if temp < result:
        result = temp
  return result
```

Листинг 3.2 — Муравьиный алгоритм

```
def antForce(graph, size, params):
  vision = []
  pheromon = []
  pheromon_cur = []
  ants = []
  best_len = graph[size-1][0]
  for i in range(size):
    if i != size-1:
      best_len += graph[i][i+1]
    ants.append(Ant(i, size))
  vision.append(size*[0])
  pheromon.append(size*[params.pheromone])
  pheromon_cur.append(size*[0.0])
  for j in range(size):
    if i != j:
      vision[i][j] = 1/graph[i][j]
  for _ in range(params.life_time):
    for k in range(size):
      path_length = 0
      for i in range(size-1):
        ants[k].to_visit[ants[k].position] = 0
        probs = size*[0]
        for j in range(size):
          if not ants[k].to_visit[j]:
            probs[j] = [0, j]
          else:
            temp = 0.0
            for z in range(size):
              if ants[k].to_visit[z]:
                temp += pow(pheromon[ants[k].position][z],
                    params.alpha)*\
                     pow(vision[ants[k].position][z],
                        params.beta)
```

```
32                temp = pow(pheromon[ants[k].position][j],
                      params.alpha)*\
33                  pow(vision[ants[k].position][j], params.beta
                        )/temp
34                probs[j] = [temp, j]
35            probs.sort(key=lambda x:x[0])
36            summ = 0.0
37            for j in range(size):
38              if probs[j][0]:
39                temp = summ
40                summ += probs[j][0]
41                probs[j][0] += temp
42            new_pos = newPos(probs, size)
43            path_length += graph[ants[k].position][new_pos]
44            pheromon_cur[ants[k].position][new_pos] += params.
                 opt_len/path_length
45            pheromon_cur[new_pos][ants[k].position] += params.
                 opt_len/path_length
46            ants[k].position = new_pos
47          path_length += graph[ants[k].position][k]
48          pheromon_cur[ants[k].position][k] += params.opt_len/
               path_length
49          pheromon_cur[k][ants[k].position] += params.opt_len/
               path_length
50          ants[k].position = k
51          for j in range(size):
52            ants[k].to_visit[j] = 1
53          if best_len > path_length:
54            best_len = path_length
55        for i in range(size):
56          for j in range(size):
57            if i < j:
58              pheromon[i][j] = (1-params.vape)*pheromon[i][j]
                   + pheromon_cur[i][j]
59              pheromon[j][i] = pheromon[i][j]
60              pheromon_cur[i][j] = 0.0
```

```
61                 pheromon_cur[j][i] = 0.0
62     return best_len
```

3.4 Пример работы программы

На рисунке 3.1 приведен пример работы программы.

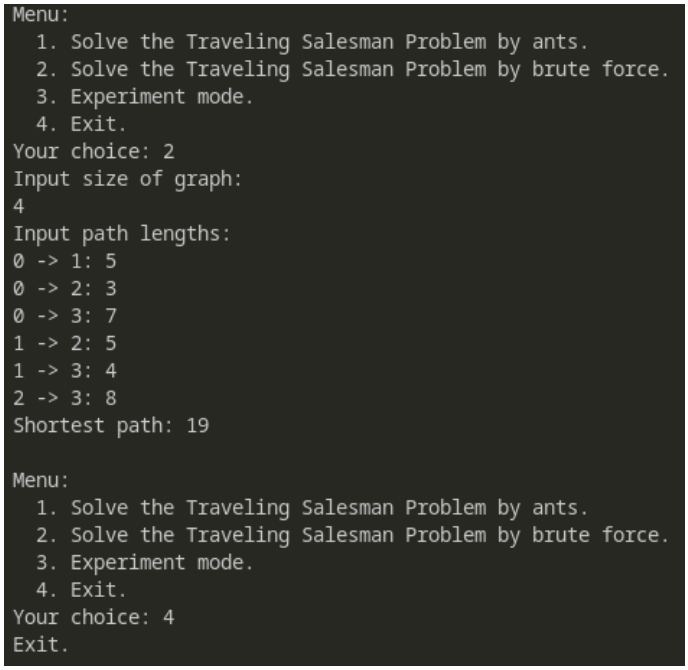

Рисунок 3.1 — Пример работы программы

3.5 Вывод по технологической части

В данном разделе были описаны средства реализации, представлены требования к ПО и реализованы алгоритмы для решения задачи коммивояжера: алгоритм полного перебора и муравьиный алгоритм.

4 Исследовательская часть

В этом разделе будет исследовано быстродействие разработанных реализаций алгоритмов и точность муравьиного алгоритма в зависимости от его входных параметров.

4.1 Технические характеристики

Ниже приведены технические характеристики устройства, на котором было проведено тестирование ПО:

- операционная система Windows 10 64-разрядная;
- оперативная память 16 ГБ;
- процессор Intel(R) Core(TM) i5-4690 @ 3.50ГГц.

4.2 Время выполнения реализаций алгоритмов

Время выполнения реализаций алгоритмов замерялось с помощью специальной функции `process_time()` [2] из модуля `time`, которая возвращает значение в долях секунды процессорного времени текущего процесса. Контрольная точка возвращаемого значения не определена, поэтому допустима только разница между результатами последовательных вызовов.

В таблицах 4.1 и 4.2 приведены результаты замеров. В первой приведены результаты замеров при относительно одинаковых возможных длинах путей (от 10 до 50), во второй же длины путей между городами могут различаться значительно (от 10 до 500). Точностью в таблице названо число совпадений результатов работы муравьиного алгоритма и алгоритма полного перебора (из пяти итераций).

Таблица 4.1 — Время работы реализаций алгоритмов при длинах путей от 10 до 50

Размер	Длина жизни	α	β	Точность	Время полн. перебора	Время мурав.
3	10	0.0	1.0	5	0.00000219	0.00078882
3	10	0.1	0.9	5	0.00000084	0.00041421
3	10	0.2	0.8	5	0.00000034	0.00018570
3	10	0.3	0.7	5	0.00000035	0.00018318
3	10	0.4	0.6	5	0.00000033	0.00018223
3	10	0.5	0.5	5	0.00000034	0.00018198
3	10	0.6	0.4	5	0.00000037	0.00019560
3	10	0.7	0.3	5	0.00000036	0.00018212
3	10	0.8	0.2	5	0.00000036	0.00018297
3	10	0.9	0.1	5	0.00000033	0.00018294
3	10	1.0	0.0	5	0.00000032	0.00018285
3	50	0.0	1.0	5	0.00000033	0.00085964
3	50	0.1	0.9	5	0.00000033	0.00088145
3	50	0.2	0.8	5	0.00000033	0.00087717
3	50	0.3	0.7	5	0.00000035	0.00087489
3	50	0.4	0.6	5	0.00000033	0.00088073
3	50	0.5	0.5	5	0.00000032	0.00087492
3	50	0.6	0.4	5	0.00000036	0.00094207
3	50	0.7	0.3	5	0.00000034	0.00089784
3	50	0.8	0.2	5	0.00000036	0.00088677
3	50	0.9	0.1	5	0.00000034	0.00088047
3	50	1.0	0.0	5	0.00000035	0.00087933
3	100	0.0	1.0	5	0.00000035	0.00170278
3	100	0.1	0.9	5	0.00000037	0.00177776
3	100	0.2	0.8	5	0.00000035	0.00179710
3	100	0.3	0.7	5	0.00000035	0.00178699
3	100	0.4	0.6	5	0.00000036	0.00178131
3	100	0.5	0.5	5	0.00000036	0.00179008
3	100	0.6	0.4	5	0.00000035	0.00179600
3	100	0.7	0.3	5	0.00000036	0.00182403
3	100	0.8	0.2	5	0.00000036	0.00179437
3	100	0.9	0.1	5	0.00000034	0.00179425
3	100	1.0	0.0	5	0.00000065	0.00179299

Таблица 4.1 (продолж.)

Размер	Длина жизни	α	β	Точность	Время полн. перебора	Время мурав.
5	10	0.0	1.0	5	0.00008234	0.00097245
5	10	0.1	0.9	5	0.00008125	0.00099766
5	10	0.2	0.8	5	0.00008129	0.00103464
5	10	0.3	0.7	5	0.00008028	0.00099031
5	10	0.4	0.6	5	0.00008114	0.00099809
5	10	0.5	0.5	5	0.00008147	0.00101160
5	10	0.6	0.4	5	0.00008073	0.00100438
5	10	0.7	0.3	5	0.00007975	0.00100827
5	10	0.8	0.2	4	0.00008039	0.00100578
5	10	0.9	0.1	5	0.00007988	0.00100109
5	10	1.0	0.0	5	0.00008171	0.00101838
5	50	0.0	1.0	5	0.00008018	0.00479159
5	50	0.1	0.9	5	0.00008126	0.00500221
5	50	0.2	0.8	5	0.00008132	0.00501788
5	50	0.3	0.7	5	0.00008105	0.00502501
5	50	0.4	0.6	5	0.00008084	0.00498023
5	50	0.5	0.5	5	0.00008028	0.00501089
5	50	0.6	0.4	5	0.00008124	0.00504253
5	50	0.7	0.3	5	0.00008204	0.00501831
5	50	0.8	0.2	5	0.00008240	0.00501918
5	50	0.9	0.1	5	0.00008108	0.00503401
5	50	1.0	0.0	5	0.00008176	0.00504283
5	100	0.0	1.0	5	0.00008248	0.00963954
5	100	0.1	0.9	5	0.00008121	0.01006820
5	100	0.2	0.8	5	0.00008212	0.01005573
5	100	0.3	0.7	5	0.00008191	0.01000944
5	100	0.4	0.6	5	0.00008123	0.01008802
5	100	0.5	0.5	5	0.00008327	0.01011338
5	100	0.6	0.4	5	0.00008212	0.01011456
5	100	0.7	0.3	5	0.00008270	0.01012282
5	100	0.8	0.2	5	0.00008134	0.01010857
5	100	0.9	0.1	5	0.00008162	0.01007508
5	100	1.0	0.0	5	0.00008352	0.01013730

Таблица 4.1 (продолж.)

Размер	Длина жизни	α	β	Точность	Время полн. перебора	Время мурав.
7	10	0.0	1.0	2	0.00390088	0.00317735
7	10	0.1	0.9	3	0.00391717	0.00334396
7	10	0.2	0.8	3	0.00394429	0.00329568
7	10	0.3	0.7	5	0.00391351	0.00331139
7	10	0.4	0.6	4	0.00392883	0.00333498
7	10	0.5	0.5	3	0.00391856	0.00332484
7	10	0.6	0.4	5	0.00390492	0.00332260
7	10	0.7	0.3	4	0.00387885	0.00331713
7	10	0.8	0.2	2	0.00390997	0.00330329
7	10	0.9	0.1	1	0.00387674	0.00330903
7	10	1.0	0.0	3	0.00396468	0.00332733
7	50	0.0	1.0	5	0.00389941	0.01569061
7	50	0.1	0.9	5	0.00389956	0.01639940
7	50	0.2	0.8	5	0.00386870	0.01652652
7	50	0.3	0.7	5	0.00390567	0.01642507
7	50	0.4	0.6	5	0.00397815	0.01673648
7	50	0.5	0.5	5	0.00391534	0.01657062
7	50	0.6	0.4	5	0.00391928	0.01666064
7	50	0.7	0.3	5	0.00396597	0.01659956
7	50	0.8	0.2	5	0.00394941	0.01667409
7	50	0.9	0.1	5	0.00394745	0.01664485
7	50	1.0	0.0	5	0.00391879	0.01668132
7	100	0.0	1.0	5	0.00394480	0.03162796
7	100	0.1	0.9	5	0.00396030	0.03331885
7	100	0.2	0.8	5	0.00393197	0.03292885
7	100	0.3	0.7	5	0.00389110	0.03308667
7	100	0.4	0.6	5	0.00394055	0.03314562
7	100	0.5	0.5	5	0.00402526	0.03326070
7	100	0.6	0.4	5	0.00399044	0.03315323
7	100	0.7	0.3	5	0.00401980	0.03340191
7	100	0.8	0.2	5	0.00393757	0.03330504
7	100	0.9	0.1	5	0.00392084	0.03315402
7	100	1.0	0.0	4	0.00393564	0.03345400

Таблица 4.1 (продолж.)

Размер	Длина жизни	α	β	Точность	Время полн. перебора	Время мурав.
9	10	0.0	1.0	0	0.41131790	0.00991216
9	10	0.1	0.9	1	0.37120794	0.00929422
9	10	0.2	0.8	0	0.35666487	0.00884292
9	10	0.3	0.7	2	0.32776446	0.00821195
9	10	0.4	0.6	1	0.33273496	0.00906158
9	10	0.5	0.5	1	0.39692422	0.00898752
9	10	0.6	0.4	2	0.32793279	0.00821731
9	10	0.7	0.3	0	0.39141998	0.00988386
9	10	0.8	0.2	0	0.32636173	0.00827970
9	10	0.9	0.1	0	0.32877880	0.00828145
9	10	1.0	0.0	1	0.32997707	0.00828740
9	50	0.0	1.0	1	0.32823027	0.03906523
9	50	0.1	0.9	1	0.32877006	0.04125859
9	50	0.2	0.8	3	0.32958525	0.04121013
9	50	0.3	0.7	4	0.32797867	0.04158168
9	50	0.4	0.6	3	0.32962953	0.04118422
9	50	0.5	0.5	3	0.32817190	0.04154504
9	50	0.6	0.4	3	0.35727065	0.04555401
9	50	0.7	0.3	3	0.55569646	0.05950507
9	50	0.8	0.2	5	0.57243198	0.06031459
9	50	0.9	0.1	5	0.55894089	0.05924416
9	50	1.0	0.0	2	0.56761876	0.05973323
9	100	0.0	1.0	0	0.56204300	0.11373902
9	100	0.1	0.9	3	0.55952135	0.11841433
9	100	0.2	0.8	2	0.57111101	0.12085845
9	100	0.3	0.7	3	0.56669370	0.12064041
9	100	0.4	0.6	5	0.56082930	0.11856957
9	100	0.5	0.5	5	0.56987188	0.11912484
9	100	0.6	0.4	3	0.56953865	0.11965193
9	100	0.7	0.3	5	0.56507396	0.11958399
9	100	0.8	0.2	5	0.56631942	0.12077165
9	100	0.9	0.1	5	0.56608504	0.11889582
9	100	1.0	0.0	5	0.55832422	0.11897110

Таблица 4.2 — Время работы реализаций алгоритмов при длинах путей от 10 до 500

Размер	Длина жизни	α	β	Точность	Время полн. перебора	Время мурав.
3	10	0.0	1.0	5	0.00000051	0.00027483
3	10	0.1	0.9	5	0.00000045	0.00026964
3	10	0.2	0.8	5	0.00000045	0.00027447
3	10	0.3	0.7	5	0.00000046	0.00027675
3	10	0.4	0.6	5	0.00000045	0.00027581
3	10	0.5	0.5	5	0.00000046	0.00027470
3	10	0.6	0.4	5	0.00000046	0.00027806
3	10	0.7	0.3	5	0.00000043	0.00027488
3	10	0.8	0.2	5	0.00000045	0.00027612
3	10	0.9	0.1	5	0.00000045	0.00027632
3	10	1.0	0.0	5	0.00000045	0.00027627
3	50	0.0	1.0	5	0.00000044	0.00130708
3	50	0.1	0.9	5	0.00000044	0.00133097
3	50	0.2	0.8	5	0.00000043	0.00133850
3	50	0.3	0.7	5	0.00000044	0.00133814
3	50	0.4	0.6	5	0.00000046	0.00133159
3	50	0.5	0.5	5	0.00000043	0.00132636
3	50	0.6	0.4	5	0.00000044	0.00133234
3	50	0.7	0.3	5	0.00000044	0.00132780
3	50	0.8	0.2	5	0.00000046	0.00133007
3	50	0.9	0.1	5	0.00000044	0.00132219
3	50	1.0	0.0	5	0.00000046	0.00136712
3	100	0.0	1.0	5	0.00000043	0.00258292
3	100	0.1	0.9	5	0.00000045	0.00264400
3	100	0.2	0.8	5	0.00000048	0.00266488
3	100	0.3	0.7	5	0.00000047	0.00265973
3	100	0.4	0.6	5	0.00000046	0.00268160
3	100	0.5	0.5	5	0.00000044	0.00265839
3	100	0.6	0.4	5	0.00000045	0.00264781
3	100	0.7	0.3	5	0.00000045	0.00265933
3	100	0.8	0.2	5	0.00000044	0.00268129
3	100	0.9	0.1	5	0.00000045	0.00265221
3	100	1.0	0.0	5	0.00000044	0.00266374

Таблица 4.2 (продолж.)

Размер	Длина жизни	α	β	Точность	Время полн. перебора	Время мурав.
5	10	0.0	1.0	5	0.00015329	0.00150426
5	10	0.1	0.9	5	0.00015133	0.00156364
5	10	0.2	0.8	5	0.00015108	0.00155528
5	10	0.3	0.7	5	0.00015012	0.00155513
5	10	0.4	0.6	5	0.00015169	0.00156079
5	10	0.5	0.5	5	0.00014993	0.00155889
5	10	0.6	0.4	5	0.00015135	0.00155541
5	10	0.7	0.3	5	0.00015091	0.00155550
5	10	0.8	0.2	5	0.00015066	0.00155332
5	10	0.9	0.1	5	0.00015287	0.00156550
5	10	1.0	0.0	4	0.00014941	0.00156148
5	50	0.0	1.0	5	0.00015032	0.00742698
5	50	0.1	0.9	5	0.00015113	0.00771625
5	50	0.2	0.8	5	0.00015027	0.00769867
5	50	0.3	0.7	5	0.00015005	0.00766503
5	50	0.4	0.6	5	0.00015009	0.00772060
5	50	0.5	0.5	5	0.00014960	0.00761381
5	50	0.6	0.4	5	0.00014975	0.00757860
5	50	0.7	0.3	5	0.00015013	0.00759881
5	50	0.8	0.2	5	0.00014947	0.00756942
5	50	0.9	0.1	5	0.00014927	0.00755403
5	50	1.0	0.0	5	0.00014754	0.00755866
5	100	0.0	1.0	5	0.00014930	0.01451482
5	100	0.1	0.9	5	0.00014964	0.01520552
5	100	0.2	0.8	5	0.00014933	0.01519972
5	100	0.3	0.7	5	0.00015037	0.01498814
5	100	0.4	0.6	5	0.00014759	0.01491379
5	100	0.5	0.5	5	0.00014860	0.01493828
5	100	0.6	0.4	5	0.00015161	0.01502105
5	100	0.7	0.3	5	0.00015101	0.01514263
5	100	0.8	0.2	5	0.00015009	0.01503636
5	100	0.9	0.1	5	0.00014976	0.01520454
5	100	1.0	0.0	5	0.00014984	0.01527121

Таблица 4.2 (продолж.)

Размер	Длина жизни	α	β	Точность	Время полн. перебора	Время мурав.
7	10	0.0	1.0	5	0.00722309	0.00477381
7	10	0.1	0.9	5	0.00722765	0.00500741
7	10	0.2	0.8	3	0.00725934	0.00499453
7	10	0.3	0.7	4	0.00711199	0.00491041
7	10	0.4	0.6	4	0.00708778	0.00490368
7	10	0.5	0.5	4	0.00700027	0.00488140
7	10	0.6	0.4	4	0.00710628	0.00477898
7	10	0.7	0.3	3	0.00709041	0.00467442
7	10	0.8	0.2	3	0.00705312	0.00468630
7	10	0.9	0.1	3	0.00712018	0.00469490
7	10	1.0	0.0	3	0.00708587	0.00469429
7	50	0.0	1.0	5	0.00702237	0.02299568
7	50	0.1	0.9	5	0.00711434	0.02337047
7	50	0.2	0.8	5	0.00709199	0.02344858
7	50	0.3	0.7	5	0.00717292	0.02344074
7	50	0.4	0.6	5	0.00719640	0.02379326
7	50	0.5	0.5	5	0.00731073	0.02404188
7	50	0.6	0.4	5	0.00724545	0.02376892
7	50	0.7	0.3	5	0.00722176	0.02374369
7	50	0.8	0.2	4	0.00723187	0.02382991
7	50	0.9	0.1	5	0.00724902	0.02378414
7	50	1.0	0.0	4	0.00708376	0.02333004
7	100	0.0	1.0	5	0.00713661	0.04638983
7	100	0.1	0.9	5	0.00721081	0.04732876
7	100	0.2	0.8	5	0.00723819	0.04756339
7	100	0.3	0.7	5	0.00717817	0.04731467
7	100	0.4	0.6	5	0.00717493	0.04675035
7	100	0.5	0.5	5	0.00691494	0.04506613
7	100	0.6	0.4	5	0.00696528	0.04511164
7	100	0.7	0.3	5	0.00686388	0.04536150
7	100	0.8	0.2	5	0.00692658	0.04580344
7	100	0.9	0.1	5	0.00694621	0.04601983
7	100	1.0	0.0	5	0.00691595	0.04633113

Таблица 4.2 (продолж.)

Размер	Длина жизни	α	β	Точность	Время полн. перебора	Время мурав.
9	10	0.0	1.0	3	0.56953802	0.01126735
9	10	0.1	0.9	0	0.56461737	0.01137432
9	10	0.2	0.8	3	0.56095980	0.01142844
9	10	0.3	0.7	1	0.57596527	0.01151098
9	10	0.4	0.6	0	0.57362612	0.01137042
9	10	0.5	0.5	0	0.56511071	0.01132695
9	10	0.6	0.4	3	0.56856427	0.01151129
9	10	0.7	0.3	4	0.57450196	0.01150368
9	10	0.8	0.2	1	0.58161975	0.01151782
9	10	0.9	0.1	1	0.56220697	0.01135985
9	10	1.0	0.0	1	0.57084538	0.01147069
9	50	0.0	1.0	3	0.58971276	0.05691730
9	50	0.1	0.9	3	0.56066670	0.05605785
9	50	0.2	0.8	5	0.57000091	0.05695092
9	50	0.3	0.7	4	0.56561937	0.05670521
9	50	0.4	0.6	4	0.57164225	0.05692357
9	50	0.5	0.5	5	0.57182183	0.05719912
9	50	0.6	0.4	4	0.56485867	0.05670624
9	50	0.7	0.3	4	0.56994893	0.05726910
9	50	0.8	0.2	5	0.57125262	0.05707906
9	50	0.9	0.1	5	0.56632235	0.05630520
9	50	1.0	0.0	4	0.56707140	0.05631264
9	100	0.0	1.0	5	0.37119206	0.08174221
9	100	0.1	0.9	5	0.56988004	0.11301679
9	100	0.2	0.8	4	0.56630840	0.11419434
9	100	0.3	0.7	5	0.57213590	0.11406609
9	100	0.4	0.6	5	0.56611648	0.11420414
9	100	0.5	0.5	5	0.56483259	0.11411940
9	100	0.6	0.4	4	0.56650787	0.11278648
9	100	0.7	0.3	5	0.57804824	0.11418779
9	100	0.8	0.2	4	0.56481363	0.11366978
9	100	0.9	0.1	5	0.56938602	0.11386458
9	100	1.0	0.0	3	0.57131235	0.11474000

4.3 Вывод по исследовательской части

По результатам замеров можно сделать несколько выводов, каждый из которых может опираться на различные входные данные.

Вне зависимости от длин путей между городами на малых размерах исходного графа алгоритм полного перебора будет давать результат лучше, поскольку не так много времени займет перебор всех возможных путей. В совокупности с эталонной точностью алгоритм полного перебора на малых размерах графа, а именно — до 6 городов, имеет выигрыш во времени вне зависимости от остальных параметров.

Длина жизни колонии муравьев влияет на время выполнения реализации муравьиного алгоритма. Так, с увеличением времени жизни возрастает время выполнения. Поэтому лучшие времена муравьиный алгоритм показывает при наименьших временах жизни колонии. При этом малое время жизни дает низкий показатель точности при количестве городов большем 5. Можно сказать, что на больших размерах для достижения высокого показателя точности необходимо указывать достаточное время жизни колонии муравьев, при этом можно достичь эталонной точности при высокой производительности муравьиного алгоритма.

Если рассматривать влияние параметров α и β на точность вычислений, то можно заметить, что при малом количестве городов они не влияют на точность, муравьиный алгоритм и так вычисляет оптимальную длину пути с эталонной точностью. При количестве городов большем 5 и длинах путей от 10 до 50 самый стабильный средний результат показывает пара $\alpha = 0.4$ и $\beta = 0.6$, а при длинах путей от 10 до 500 — $\alpha = 0.7$ и $\beta = 0.3$.

Заключение

В ходе проделанной работы была достигнута поставленная цель и решены следующие задачи:

- изучены алгоритм полного перебора и муравьиный алгоритм для решения задачи коммивояжера;
- разработаны схемы этих алгоритмов;
- реализованы изученные алгоритмы;
- проведено тестирование разработанного программного обеспечения;
- проведен анализ скорости работы реализаций алгоритмов.

Было доказано, что при количестве городов меньшем 7 алгоритм полного перебора показывает лучшую производительность в совокуности с эталонной точностью. при больших размерностях исходного графа муравьиный алгоритм для демонстрации хорошей точности требует продолжительности жизни колонии муравьев большей 100. Несмотря на то, что увеличение этого параметра влечет увеличение времени выполнения алгоритма, он все равно будет давать лучший показатель производительности за счет того, что при таких размерностях графа в алгоритме грубой силы необходимо перебрать огромное количество возможных замкнутых путей.

Список литературы

[1] Python [Электронный ресурс]. Режим доступа: https://python.org/. Дата обращения: 25.12.2021.

[2] Функция process_time [Электронный ресурс]. Режим доступа: https://docs.python.org/3/library/time.html. Дата обращения: 25.12.2021.

[3] Visual Studio Code - Code Editing [Электронный ресурс]. Режим доступа: https://code.visualstudio.com. Дата обращения: 25.12.2021.

[4] Задача коммивояжера [Электронный ресурс]. Режим доступа: https://science.fandom.com/ru/wiki/zadacha-kommivoyazshera.
Дата обращения: 25.12.2021.