



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»**

«Муравьиный алгоритм»

Студент \_\_\_\_\_ Маслова Марина Дмитриевна

Группа \_\_\_\_\_ ИУ7-53Б

Оценка (баллы) \_\_\_\_\_

Преподаватель \_\_\_\_\_ Волкова Лилия Леонидовна

# Содержание

<b>Введение</b>	<b>3</b>
<b>1 Аналитическая часть</b>	<b>4</b>
1.1 Задача коммивояжера . . . . .	4
1.2 Алгоритм полного перебора . . . . .	4
1.3 Муравьиный алгоритм . . . . .	4
1.4 Вывод . . . . .	6
<b>2 Конструкторская часть</b>	<b>7</b>
2.1 Разработка алгоритмов . . . . .	7
2.2 Структура разрабатываемого ПО . . . . .	9
2.3 Классы эквивалентности при тестировании . . . . .	9
2.4 Вывод . . . . .	9
<b>3 Технологическая часть</b>	<b>10</b>
3.1 Требования к программному обеспечению . . . . .	10
3.2 Средства реализации . . . . .	10
3.3 Листинги кода . . . . .	11
3.4 Описание тестирования . . . . .	14
3.5 Вывод . . . . .	15
<b>4 Исследовательская часть</b>	<b>16</b>
4.1 Технические характеристики . . . . .	16
4.2 Примеры работы программы . . . . .	16
4.3 Результаты тестирования . . . . .	17
4.4 Постановка эксперимента по замеру времени . . . . .	17
4.5 Результаты эксперимента . . . . .	18
4.6 Вывод . . . . .	23
<b>Заключение</b>	<b>25</b>
<b>Список литературы</b>	<b>26</b>
<b>Приложение А Параметризация для класса данных 1</b>	<b>27</b>



# Введение

В последние два десятилетия при оптимизации сложных систем исследователи все чаще применяют природные механизмы поиска наилучших решений. Одним из таких механизмов являются муравьиные алгоритмы – новый перспективный метод оптимизации, базирующийся на моделировании поведения колонии муравьев [1]. Первой задачей, к которой был применен муравьиный алгоритм, является проблема коммивояжера (или Travelling Salesman Problem – TSP) [2]. На основе этой задачи будет проводиться исследование муравьиного алгоритма в данной лабораторной работе.

**Целью данной работы** является проведение сравнительного анализа метода полного перебора и эвристического метода на базе муравьиного алгоритма.

Для достижения поставленной цели необходимо выполнить следующие **задачи**:

- описать алгоритм полного перебора для решения задачи коммивояжера;
- описать муравьиный алгоритм для решения задачи коммивояжера;
- описать функциональные требования к программе;
- разработать описанные алгоритмы;
- реализовать алгоритм полного перебора и муравьиный алгоритм для решения задачи коммивояжера;
- провести тестирование реализованных алгоритмов;
- провести сравнительный анализ алгоритмов по времени работы реализаций;
- провести параметризацию муравьиног алгоритма на двух классах данных;
- сделать выводы по полученным результатам.

# 1 Аналитическая часть

В данном разделе представлено теоретическое описание задачи коммивояжера, а алгоритм полного перебора и муравьиный алгоритм для её решения.

## 1.1 Задача коммивояжера

**Задача коммивояжера** (или *задача о путешествующем торговце*) состоит в поиске кратчайшего замкнутого маршрута, проходящего через все города ровно 1 раз [1]. Если говорить формально, то необходимо найти гамильтонов цикл минимального веса во взвешенном полном графе.

## 1.2 Алгоритм полного перебора

**Алгоритм полного перебора** для решения задачи коммивояжера предполагает рассмотрение всех возможных путей в графе и выбор наименьшего из них. Данный алгоритм имеет высокую сложность  $O(n!)$ , что большие затраты по времени даже при небольших значениях числа вершин в графе.

## 1.3 Муравьиный алгоритм

Муравьиные алгоритмы представляют собой новый перспективный метод решения задач оптимизации, в основе которого лежит моделирование поведения колонии муравьев [3]. Колония представляет собой систему с очень простыми правилами автономного поведения особей.

Каждый муравей определяет для себя маршрут, который необходимо пройти на основе феромона, который он ощущает, во время прохождения, каждый муравей оставляет феромон на своем пути, чтобы остальные муравьи могли по нему ориентироваться. При этом феромон испаряется для исключения случая, когда все муравьи движутся одним и тем же субоптимальным маршрутом. В результате при прохождении каждым муравьем различного маршрута наибольшее число феромона остается на оптимальном пути.

Для каждого муравья переход из города  $i$  в город  $j$  зависит от трех составляющих:

– Память муравья (tabu list) — 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.

– Видимость — величина, обратная расстоянию:  $\eta_{ij} = 1/D_{ij}$ , где  $D_{ij}$  — расстояние между городами  $i$  и  $j$ . Эта величина выражает эвристическое желание муравья посетить город  $i$  из города  $j$ , чем ближе город, тем больше желание его посетить.

– Виртуальный след феромона  $\tau_{ij}$  на ребре  $(i, j)$  представляет подтвержденное муравьиным опытом желание посетить город  $j$  из города  $i$ .

Вероятность перехода  $k$ -ого муравья из города  $i$  в город  $j$  на  $t$ -й итерации определяется формулой 1.1:

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

где  $\alpha, \beta$  — настраиваемые параметры,  $J_{i,k}$  — список городов, которые надо посетить  $k$ -ому муравью, находящемуся в  $i$ -ом городе,  $\tau$  — концентрация феромона, а при  $\alpha = 0$  алгоритм вырождается в жадный.

После завершения движения всеми муравьями происходит обновление феромона. Если  $p \in [0, 1]$  — коэффициент испарения феромона, то новое значения феромона на ребре  $(i, j)$  рассчитывается по формуле 1.2:

$$\tau_{ij}(t+1) = (1-p)\tau_{ij}(t) + \Delta\tau_{ij}, \quad \Delta\tau_{ij} = \sum_{k=1}^N \tau_{ij}^k \quad (1.2)$$

где

$$\Delta\tau_{ij}^k = \begin{cases} \frac{Q}{L_k}, & \text{ребро посещено } k\text{-ым муравьем,} \\ 0, & \text{иначе} \end{cases} \quad (1.3)$$

$L_k$  — длина пути  $k$ -ого муравья,  $Q$  — некоторая константа порядка длины путей,  $N$  — количество муравьев [1].

## 1.4 Вывод

В данном разделе была описана задача коммивояжера и алгоритмы её решения: полный перебор и муравьиный алгоритм. Из представленных описаний можно предъявить ряд требований к разрабатываемому программному обеспечению:

- на вход должна подаваться матрица смежности графа, представляющего города и стоимость переезда между ними;
- для муравьиного алгоритма также должны подаваться на вход коэффициенты  $\alpha, p$  и количество итераций;
- на выходе должны выдаваться искомый путь и его длина;
- при неверных входных данных должна выдаваться ошибка.

## 2 Конструкторская часть

В данном разделе разрабатываются алгоритмы решения задачи коммивояжера: полный перебор и муравьиный алгоритм, также описывается структура программы и способы её тестирования.

### 2.1 Разработка алгоритмов

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

На рисунке 2.2 представлена схема муравьиного алгоритма для решения задачи коммивояжера.

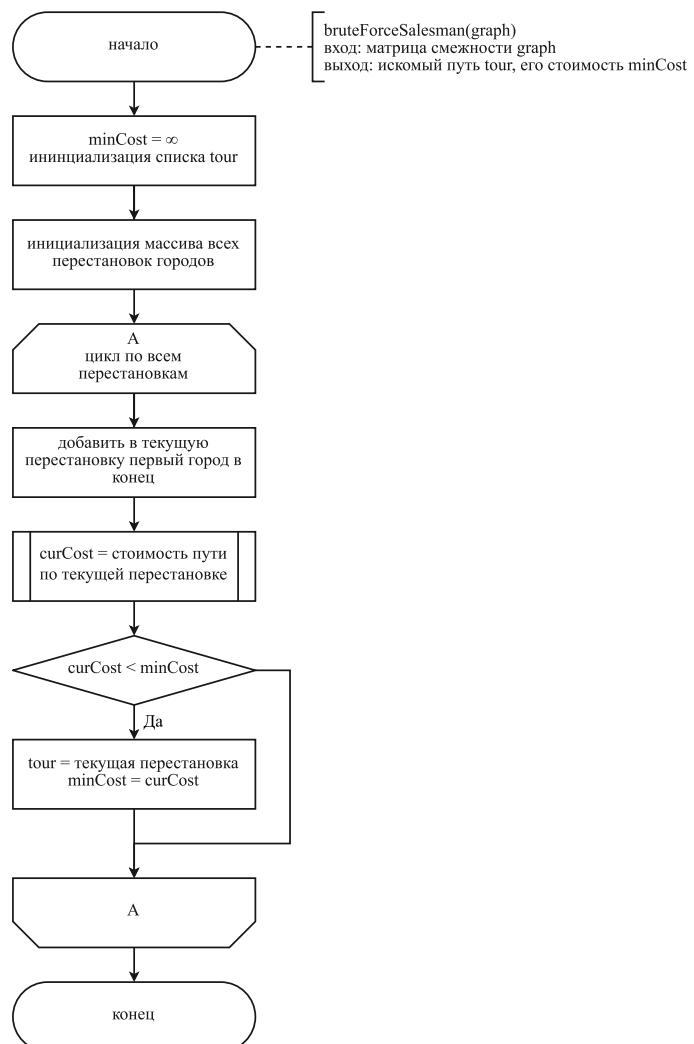


Рисунок 2.1 – Схема алгоритма полного перебора

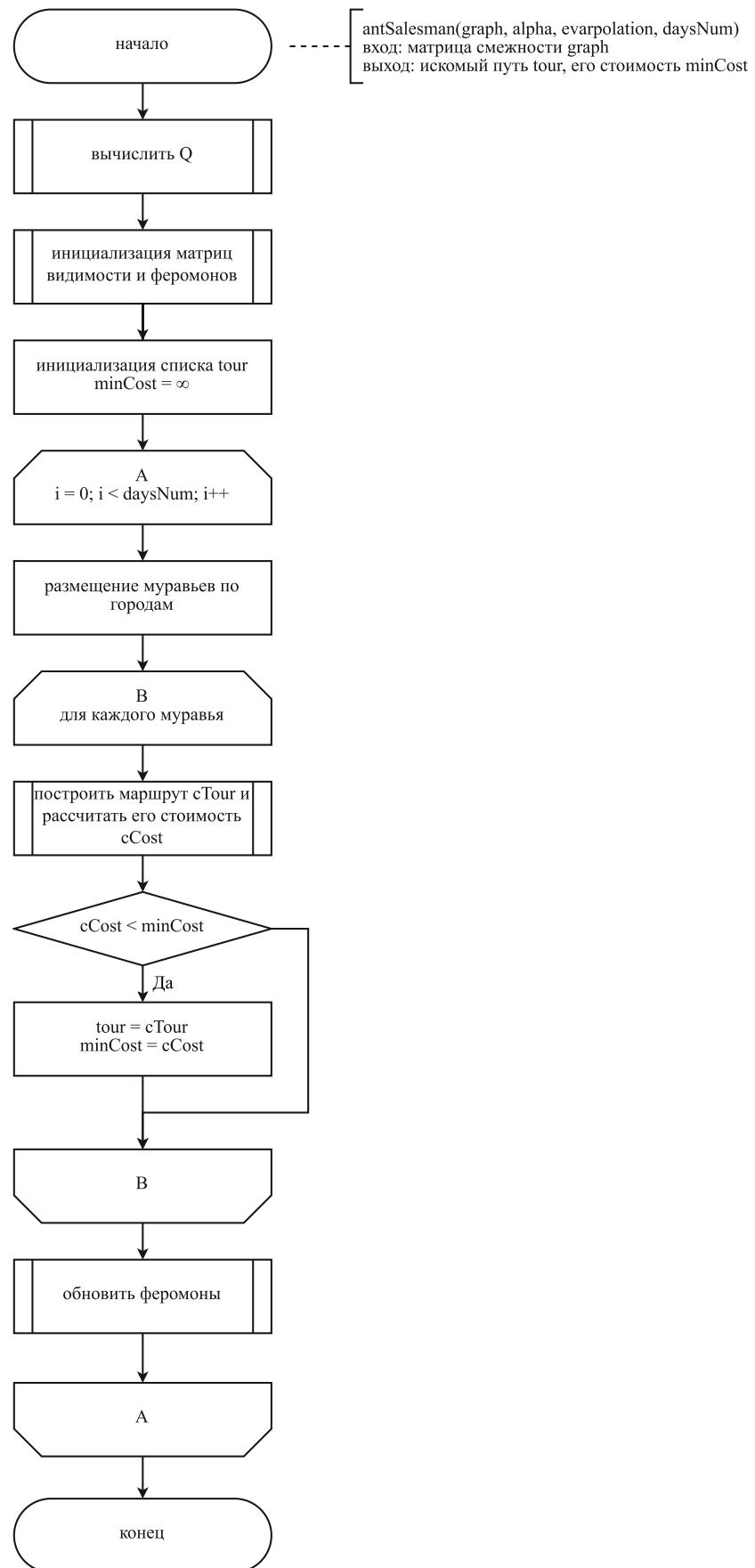


Рисунок 2.2 – Схема муравьиного алгоритма

## **2.2 Структура разрабатываемого ПО**

Для реализации разрабатываемого программного обеспечения будет использоваться метод структурного программирования. Каждый из алгоритмов будет представлен отдельной функцией, при необходимости будут выделены подпрограммы для каждой из них. Также будут реализованы функции для ввода-вывода и функция, вызывающая все подпрограммы для связности и полноценности программы.

## **2.3 Классы эквивалентности при тестировании**

Для тестирования программного обеспечения во множестве тестов будут выделены следующие классы эквивалентности:

- неверно выбран файл с матрицей смежности;
- неверный ввод коэффициента  $\alpha$ ;
- неверный ввод коэффициента  $p$ ;
- неверный ввод количество интераций;
- корректные данные.

## **2.4 Вывод**

В данном разделе были разработаны алгоритмы решения задачи коммивояжера, была описана структура разрабатываемого ПО. Для дальнейшей проверки правильности работы программы были выделены классы эквивалентности тестов.

### **3 Технологическая часть**

В данном разделе описаны требования к программному обеспечению, средства реализации, приведены листинги кода и данные, на которых будет проводиться тестирование.

#### **3.1 Требования к программному обеспечению**

Программа должна предоставлять следующие возможности:

- ввод имени файла с матрицей смежности графа;
- ввод коэффициентов для муравьиного алгоритма;
- вывод искомого пути и минимальной стоимости;
- подсчет времени работы алгоритмов на различных значениях количества вершин;
- параметризация муравьиного алгоритма с оценкой точности при заданных параметрах.

#### **3.2 Средства реализации**

Для реализации данной лабораторной работы выбран интерпретируемый язык программирования высокого уровня Python[4], так как он позволяет реализовывать сложные задачи за кратчайшие сроки за счет простоты синтаксиса и наличия большого количества подключаемых библиотек.

В качестве среды разработки выбран текстовый редактор Vim[5] с установленными плагинами автодополнения и поиска ошибок в процессе написания, так как он реализует быстрое перемещение по тексту программы и простое взаимодействие с командной строкой.

Замеры времени проводились при помощи функции process\_time\_ns из библиотеки time[6].

### 3.3 Листинги кода

В данном подразделе представлены листинги кода алгоритмов:

- реализация алгоритма полного перебора (листинги 3.1-3.2);
- реализация муравьиного алгоритма (листинги 3.3-3.9).

Листинг 3.1 – Реализация алгоритма полного перебора

```
1 def salesman(graph):
2     if len(graph) > 10:
3         return []
4     tour = []
5     minCost = float("inf")
6
7     for perm in permutations(list(range(len(graph)))):
8         curTour = list(perm)
9         curTour.append(perm[0])
10
11        curCost = getTourCost(curTour, graph)
12
13        if curCost < minCost:
14            tour = curTour
15            minCost = curCost
16
17    return tour, minCost
```

Листинг 3.2 – Функция поиска стоимости пути

```
1 def getTourCost(tour, graph):
2     cost = 0
3
4     for i in range(len(tour) - 1):
5         cost += graph[tour[i]][tour[i + 1]]
6
7     return cost
```

Листинг 3.3 – Реализация муравьиного алгоритма

```
1 def salesman(graph, alpha, evarpolation, daysNum):
2     num = len(graph)
3
4     if num <= 1:
5         return [0] if num else [], 0
6
7     Q = findCostOrder(graph)
8     vis, phero = getEdgeCoefs(graph)
9     tour = []
```

### Листинг 3.3 (продолжение)

```
10 minCost = float("inf")
11
12     for i in range(daysNum):
13         ants = [{'tabu' : [j], 'cost' : 0} for j in range(num)]
14
15     for j, ant in enumerate(ants):
16         ants[j] = getAntTour(ant, graph, vis, phero, alpha)
17
18     if ants[j]['cost'] < minCost:
19         minCost = ants[j]['cost']
20         tour = ants[j]['tabu']
21
22     updatePheromones(phero, ants, Q, evarpolation)
23
24 return tour, minCost
```

### Листинг 3.4 – Функция поиска параметра Q

```
1 def findCostOrder(graph):
2     size = len(graph)
3     return sum(graph[i][j] if i != j else 0
4                 for j in range(size)
5                     for i in range(size)) / size
```

### Листинг 3.5 – Функция инициализации видимости и феромонов

```
1 def getEdgeCoefs(graph):
2     size = len(graph)
3
4     eta = [[0 for j in range(size)] for i in range(size)]
5     for i in range(size):
6         for j in range(size):
7             if i != j and graph[i][j]:
8                 eta[i][j] = 1 / graph[i][j]
9
10    tau = [[START_PHEROMONE for j in range(size)] for i in range(size)]
11
12    return eta, tau
```

### Листинг 3.6 – Функция получения тура и его стоимости для одного муравья

```
1 def getAntTour(ant, graph, vis, phero, alpha):
2     size = len(graph)
3
4     while len(ant['tabu']) != size:
5         nextCity = chooseCity(ant['tabu'], size, vis, phero, alpha)
```

### Листинг 3.6 (продолжение)

```
6      ant['cost'] += graph[ant['tabu'][-1]][nextCity]
7      ant['tabu'].append(nextCity)
8
9
10     ant['cost'] += graph[ant['tabu'][-1]][ant['tabu'][0]]
11     ant['tabu'].append(ant['tabu'][0])
12
13     return ant
```

### Листинг 3.7 – Функция выбора муравьем следующего города

```
1 def chooseCity(tabuList, size, vis, phero, alpha):
2     probabilities = getProbabilities(size, tabuList, vis, phero, alpha)
3
4     choise = random()
5     probSum = probabilities[0]
6     nextCity = 0
7
8     while choise > probSum and nextCity < size - 1:
9         nextCity += 1
10        probSum += probabilities[nextCity]
11
12    return nextCity
```

### Листинг 3.8 – Получение вероятностей перехода муравья в каждый город

```
1 def getProbNumerator(cFrom, cTo, vis, phero, alpha):
2     return pow(phero[cFrom][cTo], alpha) * pow(vis[cFrom][cTo], 1 - alpha)
3
4
5 def getProbabilities(citiesNum, tabu, vis, phero, alpha):
6     probabilities = [0] * citiesNum
7
8     for city in range(citiesNum):
9         if city not in tabu:
10             curCity = tabu[-1]
11
12             probabilities[city] = getProbNumerator(curCity, city, vis, phero,
13                                         alpha)
14
15     denominator = sum(probabilities)
16
17     for i in range(citiesNum):
18         probabilities[i] /= denominator
19
20     return probabilities
```

### Листинг 3.9 – Функция обновления феромона

```
1 def updatePheromones(pheromones, ants, Q, evaporation):
2     size = len(pheromones)
3
4     addPheromone = [[0 for i in range(size)] for j in range(size)]
5
6     for ant in ants:
7         delta = Q / ant['cost']
8         for i in range(size - 1):
9             addPheromone[ant['tabu'][i]][ant['tabu'][i + 1]] += delta
10
11    for i in range(size):
12        for j in range(size):
13            pheromones[i][j] *= (1 - evaporation)
14            pheromones[i][j] += addPheromone[i][j]
15            pheromones[i][j] = 0.1 if pheromones[i][j] < MIN_PHEROMONE else pheromones[i][j]
```

## 3.4 Описание тестирования

В таблице 3.1 приведены функциональные тесты программы.

Таблица 3.1 – Функциональные тесты

Входные данные	Ожидаемый результат
пустая строка	Сообщение об ошибке
./data/g5.txt	Сообщение об ошибке
j	Сообщение об ошибке
./data/g5.txt 0.5	Сообщение об ошибке
j	Сообщение об ошибке
./data/g5.txt 0.5 0.5	Сообщение об ошибке
j	Сообщение об ошибке
./data/g5.txt 0.5 0.5 100	0 -> 2 -> 1 -> 4 -> 3 -> 0 19.0

## **3.5 Вывод**

В данном разделе были реализованы алгоритмы решения задачи коммивояжера: полный перебор и муравьиный алгоритм. Также были написаны тесты для каждого класса эквивалентности, описанного в конструкторском разделе.

## 4 Исследовательская часть

### 4.1 Технические характеристики

Технические характеристики устройства, на котором выполнялось тестирование:

- Операционная система: Manjaro [7] Linux x86\_64.
- Память: 8 GiB.
- Процессор: Intel® Core™ i5-8265U, 4 физических ядра, 8 логических ядра[8].

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

### 4.2 Примеры работы программы

На рисунке 4.1 представлены результаты работы программы.

```
ЗАДАЧА КОММИВОЯЖЕРА

РЕЗУЛЬТАТЫ
Полный перебор
Тур с минимальной стоимостью:
0 → 2 → 1 → 4 → 3 → 0
Стоимость: 19.0
Введите вес следа феромона α: 0.5
Введите коэффициент испарения феромона ρ: 0.5
Введите количество дней: 100
Муравьиный алгоритм
Тур с минимальной стоимостью:
4 → 1 → 2 → 0 → 3 → 4
Стоимость: 19.0
```

Рисунок 4.1 – Пример работы программы

## **4.3 Результаты тестирования**

Программа была протестирована на входных данных, приведенных в таблице 3.1. Полученные результаты работы программы совпали с ожидаемыми результатами.

## **4.4 Постановка эксперимента по замеру времени**

Для оценки времени работы алгоритма полного перебора и муравьиного алгоритма был проведен эксперимент, в котором определялось влияние количества вершин в графе на время работы каждого из алгоритмов. Тестирование проводилось на количестве вершин от 2 до 10 с шагом 1. Так как от запуска к запуску процессорное время, затрачиваемое на выполнение алгоритмов, менялось в определенном промежутке, необходимо было усреднить вычисляемые значения. Для этого каждый алгоритм на каждом значении запускался по 10 раз, и для полученных 10 значений определялось среднее арифметическое, которое заносилось в таблицу.

Также был проведен эксперимент для определения оптимальных параметров муравьиного алгоритма. Для этого алгоритм запускался на всех сочетаниях значений параметров, где коэффициенты  $\alpha$  и  $r$  изменились в промежутке от 0 до 1 с шагом 0.1, а количество итераций принимало значения 1, 5, 10, 50, 100, 500, 100; и полученный результат сравнивался с точным результатом, полученным с помощью алгоритма полного перебора. Параметризация проводилась на двух различных классах данных с малым (класс данных 1, формула 4.1) и большим (класс данных 2, формула 4.2) разбросом значений стоимостей перехода из одного города в другой. Размер матриц, на которых проводилась параметризация был выбран равный 9, как наибольший размер, для которого алгоритм полного перебора решает задачу за малое время.

$$K_1 = \begin{pmatrix} 0 & 4 & 6 & 4 & 9 & 1 & 8 & 5 & 8 \\ 2 & 0 & 7 & 6 & 9 & 9 & 6 & 5 & 2 \\ 1 & 3 & 0 & 2 & 2 & 8 & 10 & 3 & 2 \\ 9 & 7 & 8 & 0 & 10 & 6 & 1 & 1 & 4 \\ 3 & 10 & 4 & 3 & 0 & 7 & 10 & 8 & 1 \\ 8 & 8 & 7 & 4 & 1 & 0 & 2 & 1 & 3 \\ 9 & 10 & 6 & 2 & 3 & 2 & 0 & 4 & 9 \\ 10 & 7 & 4 & 7 & 3 & 7 & 3 & 0 & 2 \\ 6 & 4 & 3 & 7 & 4 & 10 & 3 & 5 & 0 \end{pmatrix} \quad (4.1)$$

$$K_2 = \begin{pmatrix} 0 & 1932 & 1753 & 666 & 2036 & 282 & 1055 & 1564 & 1427 \\ 1608 & 0 & 314 & 897 & 396 & 509 & 141 & 1000 & 673 \\ 1657 & 2136 & 0 & 2331 & 1431 & 7 & 2203 & 713 & 1804 \\ 2291 & 544 & 400 & 0 & 2171 & 1590 & 189 & 1074 & 1886 \\ 1335 & 1590 & 1184 & 640 & 0 & 1266 & 296 & 738 & 698 \\ 116 & 93 & 1577 & 788 & 2439 & 0 & 1231 & 2139 & 1665 \\ 315 & 1152 & 1870 & 2491 & 842 & 986 & 0 & 1235 & 13 \\ 986 & 2155 & 2397 & 1765 & 871 & 694 & 1811 & 0 & 1167 \\ 784 & 15 & 1487 & 81 & 740 & 372 & 1279 & 2453 & 0 \end{pmatrix} \quad (4.2)$$

Результаты эксперимента были представлены в виде таблиц и графиков, приведенных в следующем подразделе.

## 4.5 Результаты эксперимента

В таблице 4.1 представлены результаты измерения времени работы алгоритмов решения задачи коммивояжера от количества вершин в графе. На рисунке 4.2 представлен соответствующий график.

В таблицах 4.2, 4.3 представлена часть результатов параметризации муравьиного алгоритма на классе данных 1 с малым разбросом значений и классе данных 2 с большим разбросом значений соответственно. Полный таблицы результатов представлены в приложениях А и Б соответственно.

Таблица 4.1 – Время работы от количества вершин

Количество вершин	Перебор, нс	Муравьиный, нс
2	12851	3242286
3	6983	4819831
4	23088	9056740
5	92330	15627924
6	595158	25246094
7	4650471	37470645
8	39062941	53224607
9	375918869	73403558
10	4142465482	102014261

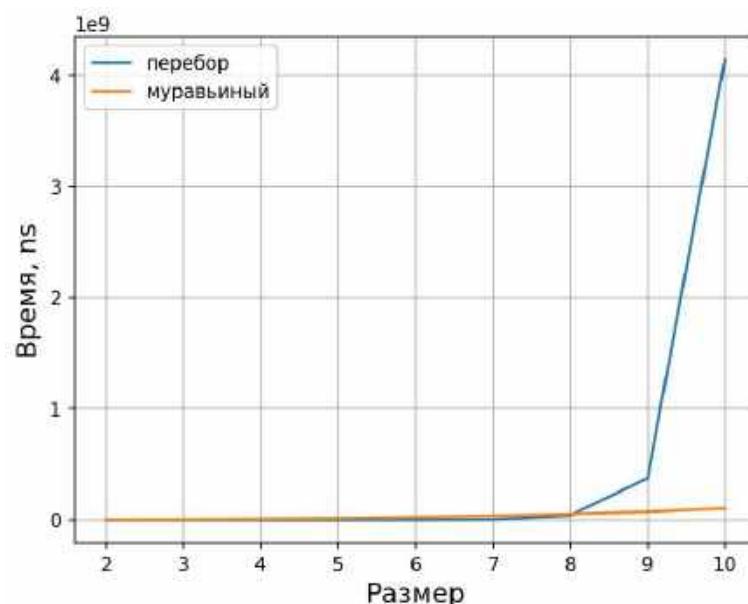


Рисунок 4.2 – График зависимости времени работы от числа заявок

Таблица 4.2 – Параметризация для класса данных 2

$\alpha$	$p$	Число итераций	Ошибка
0.0	0.1	1	10
0.0	0.1	5	6
0.0	0.1	10	0
0.0	0.1	50	1
0.0	0.1	100	0
0.0	0.1	500	0
0.0	0.1	1000	0

Таблица 4.2 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.0	0.9	1	0
0.0	0.9	5	5
0.0	0.9	10	0
0.0	0.9	50	0
0.0	0.9	100	0
0.0	0.9	500	0
0.0	0.9	1000	0
0.1	0.4	1	11
0.1	0.4	5	0
0.1	0.4	10	0
0.1	0.4	50	1
0.1	0.4	100	0
0.1	0.4	500	0
0.1	0.4	1000	0
0.1	0.7	1	10
0.1	0.7	5	4
0.1	0.7	10	0
0.1	0.7	50	2
0.1	0.7	100	0
0.1	0.7	500	0
0.1	0.7	1000	0
0.1	0.8	1	8
0.1	0.8	5	5
0.1	0.8	10	0
0.1	0.8	50	2
0.1	0.8	100	0
0.1	0.8	500	0
0.1	0.8	1000	0
0.2	0.5	1	10
0.2	0.5	5	3
0.2	0.5	10	0

Таблица 4.2 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.2	0.5	50	0
0.2	0.5	100	1
0.2	0.5	500	0
0.2	0.5	1000	0
0.3	0.0	1	6
0.3	0.0	5	4
0.3	0.0	10	0
0.3	0.0	50	1
0.3	0.0	100	0
0.3	0.0	500	0
0.3	0.0	1000	0
0.5	0.5	1	10
0.5	0.5	5	9
0.5	0.5	10	0
0.5	0.5	50	0
0.5	0.5	100	0
0.5	0.5	500	0
0.5	0.5	1000	0

Таблица 4.3 – Параметризация для класса данных 2

$\alpha$	$p$	Число итераций	Ошибка
0.2	0.0	1	1678
0.2	0.0	5	0
0.2	0.0	10	112
0.2	0.0	50	0
0.2	0.0	100	0
0.2	0.0	500	0
0.2	0.0	1000	0
0.3	0.0	1	1702

Таблица 4.3 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.3	0.0	5	0
0.3	0.0	10	112
0.3	0.0	50	0
0.3	0.0	100	0
0.3	0.0	500	0
0.3	0.0	1000	0
0.4	0.1	1	3192
0.4	0.1	5	0
0.4	0.1	10	0
0.4	0.1	50	112
0.4	0.1	100	0
0.4	0.1	500	0
0.4	0.1	1000	0
0.4	0.5	1	1559
0.4	0.5	5	0
0.4	0.5	10	112
0.4	0.5	50	0
0.4	0.5	100	0
0.4	0.5	500	0
0.4	0.5	1000	0
0.5	0.5	1	2985
0.5	0.5	5	469
0.5	0.5	10	112
0.5	0.5	50	0
0.5	0.5	100	0
0.5	0.5	500	0
0.5	0.5	1000	0
0.5	0.6	1	1230
0.5	0.6	5	919
0.5	0.6	10	112
0.5	0.6	50	0

Таблица 4.3 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.5	0.6	100	0
0.5	0.6	500	0
0.5	0.6	1000	0
0.5	0.8	1	3313
0.5	0.8	5	0
0.5	0.8	10	112
0.5	0.8	50	0
0.5	0.8	100	0
0.5	0.8	500	0
0.5	0.8	1000	0
0.6	0.6	1	1611
0.6	0.6	5	0
0.6	0.6	10	509
0.6	0.6	50	0
0.6	0.6	100	0
0.6	0.6	500	0
0.6	0.6	1000	0

## 4.6 Вывод

По результатам эксперимента можно сделать следующие выводы:

- при количестве вершин в графе до 8 алгоритм полного перебора работает быстрее муравьиного алгоритма, однако если на 3 вершинах он преобладает над муравьиным в 700 раз, то уже на 5 он преобладает в 150 раз, а на 7 – в 8, что говорит о быстром росте сложности алгоритма полного перебора;
- при количестве вершин в графе более 8 время выполнения алгоритма полного перебора резко возрастает, в то время как у муравьиного алгоритма при переходе на следующее количество вершин время возрастает не более чем в 1.5 раза;
- муравьиный алгоритм дает лучшие результаты как при малом, так при большой разбросе на значения коэффициента  $\alpha$  до 0.5, при этом

- строгой зависимости точности алгоритма от коэффициентов  $\alpha$  и  $p$  не наблюдается;
- при малых разбросах значений стоимостей лучшие результаты работы алгоритма налюдались на парах значений: (0.0, 0.1), (0.0, 0.9), (0.1, 0.4), (0.1, 0.7), (0.1, 0.8), (0.2, 0.5), (0.3, 0.0) и (0.5, 0.5); – при этом рост количества дней в полной мере устранил ошибки только при значениях (0.5, 0.5), при других значениях находилось такое количество дней, при котором находился субоптимальный маршрут с большей стоимостью;
  - при больших разбросах значений стоимостей лучшие результаты работы алгоритма налюдались на парах значений: (0.2, 0.0), (0.3, 0.0), (0.4, 0.1), (0.4, 0.5), (0.5, 0.5), (0.5, 0.6), (0.5, 0.8) и (0.6, 0.6); – при этом рост количества дней в полной мере устранил ошибки только при значениях (0.5, 0.5) и близким к ним (0.5, 0.6), при других значениях находилось такое количество дней, при котором находился субоптимальный маршрут с большей стоимостью;

Таким образом, скорость роста сложности алгоритма перебора в разы выше скорости роста сложности муравьиного алгоритма, но при малых значениях количества вершин в графе следует использовать именно метод полного перебора, так как он в любом случае дает точный результат и при малых значениях имеет преимущество в скорости перед муравьиным алгоритмом. В то же время использовать алгоритм полного пребора при значениях количества вершин в графе больше 8 не представляется возможным из-за неопределенного долгого ожидания завершения работы алгоритма, на таких значениях следует использовать муравьиный алгоритм.

При использовании муравьиного алгоритма в случае, если разброс данных не известен, следует выбирать оптимальную по результатам эксперимента пару значений  $(\alpha, p)$  равную (0.5, 0.5) при этом количество итераций выбирая от 50 и выше. При известном разбросе параметров, можно найти пару параметров, при которой алгоритм будет давать точный результат за меньшее число итераций. Поиск такого значения требует дополнительного исследования, которое в данной лабораторной работе не проводилось, был сделан вывод только о том, что значения параметра  $\alpha$  следует выбирать в промежутке [0.0, 0.5].

# Заключение

В ходе исследования был проведен сравнительный алгоритма полного перебора и муравьиного алгоритма для решения задачи коммивояжера. При малых значениях количества вершин в графе до 8 следует использовать алгоритм полного перебора, при больших значениях следует использовать муравьиный алгоритм, который дает точный результат при различных разбросах стоимостей при паре значений параметров  $(\alpha, p)$  равной  $(0.5, 0.5)$  и количестве итераций от 50.

В ходе выполнения лабораторной работы:

- были описаны и разработаны алгоритм полного перебора и муравьиный алгоритм для решения задачи коммивояжера;
- был реализован каждый из описанных алгоритмов;
- было проведено тестирование каждого из алгоритмов;
- по экспериментальным данным были сделаны выводы об эффективности по времени каждого из реализованных алгоритмов;
- была проведена параметризация муравьиного алгоритма на двух классах данных и сделаны выводы по её результатам.

Таким образом, все поставленные задачи были выполнены, а цель достигнута.

# Список литературы

- [1] Штовба С. Д. Муравьиные алгоритмы // Exponenta Pro. Математика в приложениях. — 2003. — № 4. — С. 70–75. — URL: [https://www.researchgate.net/profile/Serhiy-Shtovba/publication/279535061\\_Stovba\\_SD\\_Muravinye\\_algoritmy\\_Exponenta\\_Pro\\_Matematika\\_v\\_prilozeniah\\_-\\_2003\\_-\\_No4\\_-\\_S\\_70-75/links/55b7d36808aed621de04d99f/Stovba-SD-Muravinye-algoritmy-Exponenta-Pro-Matematika-v-prilozeniah-2003-No4-S-70-75.pdf](https://www.researchgate.net/profile/Serhiy-Shtovba/publication/279535061_Stovba_SD_Muravinye_algoritmy_Exponenta_Pro_Matematika_v_prilozeniah_-_2003_-_No4_-_S_70-75/links/55b7d36808aed621de04d99f/Stovba-SD-Muravinye-algoritmy-Exponenta-Pro-Matematika-v-prilozeniah-2003-No4-S-70-75.pdf) (дата обращения: 12.12.2021).
- [2] Зайченко Ю., Мурга Н. Исследование муравьиных алгоритмов оптимизации в задачи коммивояжера // International Journal "Information Models and Analyses". — 2013. — Т. 2, № 4. — С. 370–384. — URL: <http://www.foibg.com/ijima/vol02/ijima02-04-p08.pdf> (дата обращения: 12.12.2021).
- [3] М.В. Ульянов. Ресурсно-эффективные компьютерные алгоритмы. — ФИЗМАТЛИТ, 2008. — с. 304.
- [4] Welcome to Python. — URL: <https://www.python.org> (дата обращения: 12.10.2021).
- [5] welcome home : vim online. — URL: <https://www.vim.org/> (дата обращения: 12.10.2021).
- [6] time — Time access and conversions. — URL: [https://docs.python.org/3/library/time.html#time.process\\_time\\_ns](https://docs.python.org/3/library/time.html#time.process_time_ns) (дата обращения: 04.10.2021).
- [7] Manjaro — enjoy the simplicity. — URL: <https://manjaro.org/> (дата обращения: 17.10.2021).
- [8] Процессор Intel® Core™ i5-8265U. — Режим доступа: <https://ark.intel.com/content/www/ru/ru/ark/products/149088/intel-core-i5-8265u-processor-6m-cache-up-to-3-90-ghz.html> (дата обращения: 17.10.2021).

# Приложение А

## Параметризация для класса данных 1

Таблица А.1 – Параметризация для  
класса данных с малым  
разбросом значений

$\alpha$	$p$	Число итераций	Ошибка
0.0	0.0	1	8
0.0	0.0	5	1
0.0	0.0	10	3
0.0	0.0	50	0
0.0	0.0	100	0
0.0	0.0	500	0
0.0	0.0	1000	0
0.0	0.1	1	10
0.0	0.1	5	6
0.0	0.1	10	0
0.0	0.1	50	1
0.0	0.1	100	0
0.0	0.1	500	0
0.0	0.1	1000	0
0.0	0.2	1	9
0.0	0.2	5	7
0.0	0.2	10	3
0.0	0.2	50	0
0.0	0.2	100	0
0.0	0.2	500	0
0.0	0.2	1000	0
0.0	0.3	1	6
0.0	0.3	5	3
0.0	0.3	10	3

Таблица А.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.0	0.3	50	0
0.0	0.3	100	1
0.0	0.3	500	0
0.0	0.3	1000	0
0.0	0.4	1	7
0.0	0.4	5	5
0.0	0.4	10	2
0.0	0.4	50	0
0.0	0.4	100	0
0.0	0.4	500	0
0.0	0.4	1000	0
0.0	0.5	1	7
0.0	0.5	5	2
0.0	0.5	10	3
0.0	0.5	50	3
0.0	0.5	100	0
0.0	0.5	500	0
0.0	0.5	1000	0
0.0	0.6	1	10
0.0	0.6	5	3
0.0	0.6	10	4
0.0	0.6	50	2
0.0	0.6	100	0
0.0	0.6	500	0
0.0	0.6	1000	0
0.0	0.7	1	9
0.0	0.7	5	3
0.0	0.7	10	4
0.0	0.7	50	3
0.0	0.7	100	1
0.0	0.7	500	0

Таблица А.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.0	0.7	1000	0
0.0	0.8	1	9
0.0	0.8	5	5
0.0	0.8	10	3
0.0	0.8	50	0
0.0	0.8	100	0
0.0	0.8	500	0
0.0	0.8	1000	0
0.0	0.9	1	0
0.0	0.9	5	5
0.0	0.9	10	0
0.0	0.9	50	0
0.0	0.9	100	0
0.0	0.9	500	0
0.0	0.9	1000	0
0.0	1.0	1	6
0.0	1.0	5	4
0.0	1.0	10	1
0.0	1.0	50	0
0.0	1.0	100	0
0.0	1.0	500	0
0.0	1.0	1000	0
0.1	0.0	1	14
0.1	0.0	5	6
0.1	0.0	10	3
0.1	0.0	50	0
0.1	0.0	100	0
0.1	0.0	500	0
0.1	0.0	1000	0
0.1	0.1	1	4
0.1	0.1	5	4

Таблица А.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.1	0.1	10	5
0.1	0.1	50	0
0.1	0.1	100	0
0.1	0.1	500	0
0.1	0.1	1000	0
0.1	0.2	1	7
0.1	0.2	5	4
0.1	0.2	10	4
0.1	0.2	50	0
0.1	0.2	100	0
0.1	0.2	500	0
0.1	0.2	1000	0
0.1	0.3	1	7
0.1	0.3	5	6
0.1	0.3	10	3
0.1	0.3	50	1
0.1	0.3	100	0
0.1	0.3	500	0
0.1	0.3	1000	0
0.1	0.4	1	11
0.1	0.4	5	0
0.1	0.4	10	0
0.1	0.4	50	1
0.1	0.4	100	0
0.1	0.4	500	0
0.1	0.4	1000	0
0.1	0.5	1	11
0.1	0.5	5	6
0.1	0.5	10	3
0.1	0.5	50	0
0.1	0.5	100	0

Таблица А.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.1	0.5	500	0
0.1	0.5	1000	0
0.1	0.6	1	7
0.1	0.6	5	3
0.1	0.6	10	3
0.1	0.6	50	0
0.1	0.6	100	1
0.1	0.6	500	0
0.1	0.6	1000	0
0.1	0.7	1	10
0.1	0.7	5	4
0.1	0.7	10	0
0.1	0.7	50	2
0.1	0.7	100	0
0.1	0.7	500	0
0.1	0.7	1000	0
0.1	0.8	1	8
0.1	0.8	5	5
0.1	0.8	10	0
0.1	0.8	50	2
0.1	0.8	100	0
0.1	0.8	500	0
0.1	0.8	1000	0
0.1	0.9	1	8
0.1	0.9	5	3
0.1	0.9	10	4
0.1	0.9	50	1
0.1	0.9	100	1
0.1	0.9	500	0
0.1	0.9	1000	0
0.1	1.0	1	7

Таблица А.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.1	1.0	5	0
0.1	1.0	10	4
0.1	1.0	50	0
0.1	1.0	100	0
0.1	1.0	500	0
0.1	1.0	1000	0
0.2	0.0	1	8
0.2	0.0	5	6
0.2	0.0	10	4
0.2	0.0	50	0
0.2	0.0	100	1
0.2	0.0	500	0
0.2	0.0	1000	0
0.2	0.1	1	6
0.2	0.1	5	1
0.2	0.1	10	3
0.2	0.1	50	3
0.2	0.1	100	0
0.2	0.1	500	0
0.2	0.1	1000	0
0.2	0.2	1	13
0.2	0.2	5	5
0.2	0.2	10	1
0.2	0.2	50	2
0.2	0.2	100	0
0.2	0.2	500	0
0.2	0.2	1000	0
0.2	0.3	1	9
0.2	0.3	5	5
0.2	0.3	10	3
0.2	0.3	50	0

Таблица А.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.2	0.3	100	0
0.2	0.3	500	0
0.2	0.3	1000	0
0.2	0.4	1	17
0.2	0.4	5	3
0.2	0.4	10	3
0.2	0.4	50	3
0.2	0.4	100	0
0.2	0.4	500	0
0.2	0.4	1000	0
0.2	0.5	1	10
0.2	0.5	5	3
0.2	0.5	10	0
0.2	0.5	50	0
0.2	0.5	100	1
0.2	0.5	500	0
0.2	0.5	1000	0
0.2	0.6	1	12
0.2	0.6	5	0
0.2	0.6	10	1
0.2	0.6	50	0
0.2	0.6	100	0
0.2	0.6	500	0
0.2	0.6	1000	0
0.2	0.7	1	6
0.2	0.7	5	0
0.2	0.7	10	3
0.2	0.7	50	3
0.2	0.7	100	0
0.2	0.7	500	0
0.2	0.7	1000	0

Таблица А.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.2	0.8	1	8
0.2	0.8	5	7
0.2	0.8	10	5
0.2	0.8	50	2
0.2	0.8	100	0
0.2	0.8	500	0
0.2	0.8	1000	0
0.2	0.9	1	7
0.2	0.9	5	4
0.2	0.9	10	2
0.2	0.9	50	0
0.2	0.9	100	1
0.2	0.9	500	0
0.2	0.9	1000	0
0.2	1.0	1	8
0.2	1.0	5	4
0.2	1.0	10	6
0.2	1.0	50	0
0.2	1.0	100	0
0.2	1.0	500	0
0.2	1.0	1000	0
0.3	0.0	1	6
0.3	0.0	5	4
0.3	0.0	10	0
0.3	0.0	50	1
0.3	0.0	100	0
0.3	0.0	500	0
0.3	0.0	1000	0
0.3	0.1	1	6
0.3	0.1	5	8
0.3	0.1	10	5

Таблица А.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.3	0.1	50	0
0.3	0.1	100	0
0.3	0.1	500	0
0.3	0.1	1000	0
0.3	0.2	1	8
0.3	0.2	5	5
0.3	0.2	10	3
0.3	0.2	50	0
0.3	0.2	100	0
0.3	0.2	500	0
0.3	0.2	1000	0
0.3	0.3	1	9
0.3	0.3	5	4
0.3	0.3	10	3
0.3	0.3	50	0
0.3	0.3	100	0
0.3	0.3	500	0
0.3	0.3	1000	0
0.3	0.4	1	16
0.3	0.4	5	5
0.3	0.4	10	2
0.3	0.4	50	0
0.3	0.4	100	0
0.3	0.4	500	0
0.3	0.4	1000	0
0.3	0.5	1	7
0.3	0.5	5	5
0.3	0.5	10	4
0.3	0.5	50	0
0.3	0.5	100	0
0.3	0.5	500	0

Таблица А.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.3	0.5	1000	0
0.3	0.6	1	10
0.3	0.6	5	2
0.3	0.6	10	6
0.3	0.6	50	0
0.3	0.6	100	0
0.3	0.6	500	0
0.3	0.6	1000	0
0.3	0.7	1	12
0.3	0.7	5	3
0.3	0.7	10	3
0.3	0.7	50	0
0.3	0.7	100	0
0.3	0.7	500	0
0.3	0.7	1000	0
0.3	0.8	1	11
0.3	0.8	5	5
0.3	0.8	10	4
0.3	0.8	50	0
0.3	0.8	100	0
0.3	0.8	500	0
0.3	0.8	1000	0
0.3	0.9	1	12
0.3	0.9	5	1
0.3	0.9	10	3
0.3	0.9	50	1
0.3	0.9	100	0
0.3	0.9	500	0
0.3	0.9	1000	0
0.3	1.0	1	5
0.3	1.0	5	0

Таблица А.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.3	1.0	10	5
0.3	1.0	50	0
0.3	1.0	100	0
0.3	1.0	500	0
0.3	1.0	1000	0
0.4	0.0	1	10
0.4	0.0	5	6
0.4	0.0	10	2
0.4	0.0	50	0
0.4	0.0	100	0
0.4	0.0	500	0
0.4	0.0	1000	0
0.4	0.1	1	10
0.4	0.1	5	2
0.4	0.1	10	5
0.4	0.1	50	1
0.4	0.1	100	0
0.4	0.1	500	0
0.4	0.1	1000	0
0.4	0.2	1	16
0.4	0.2	5	5
0.4	0.2	10	5
0.4	0.2	50	0
0.4	0.2	100	0
0.4	0.2	500	0
0.4	0.2	1000	0
0.4	0.3	1	16
0.4	0.3	5	4
0.4	0.3	10	3
0.4	0.3	50	0
0.4	0.3	100	0

Таблица А.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.4	0.3	500	0
0.4	0.3	1000	0
0.4	0.4	1	8
0.4	0.4	5	5
0.4	0.4	10	3
0.4	0.4	50	0
0.4	0.4	100	1
0.4	0.4	500	0
0.4	0.4	1000	0
0.4	0.5	1	15
0.4	0.5	5	4
0.4	0.5	10	6
0.4	0.5	50	0
0.4	0.5	100	0
0.4	0.5	500	0
0.4	0.5	1000	0
0.4	0.6	1	16
0.4	0.6	5	5
0.4	0.6	10	5
0.4	0.6	50	0
0.4	0.6	100	0
0.4	0.6	500	0
0.4	0.6	1000	0
0.4	0.7	1	17
0.4	0.7	5	2
0.4	0.7	10	5
0.4	0.7	50	1
0.4	0.7	100	1
0.4	0.7	500	0
0.4	0.7	1000	0
0.4	0.8	1	11

Таблица А.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.4	0.8	5	4
0.4	0.8	10	1
0.4	0.8	50	0
0.4	0.8	100	0
0.4	0.8	500	0
0.4	0.8	1000	0
0.4	0.9	1	9
0.4	0.9	5	6
0.4	0.9	10	3
0.4	0.9	50	0
0.4	0.9	100	0
0.4	0.9	500	0
0.4	0.9	1000	0
0.4	1.0	1	10
0.4	1.0	5	4
0.4	1.0	10	3
0.4	1.0	50	0
0.4	1.0	100	1
0.4	1.0	500	0
0.4	1.0	1000	0
0.5	0.0	1	17
0.5	0.0	5	3
0.5	0.0	10	6
0.5	0.0	50	0
0.5	0.0	100	1
0.5	0.0	500	0
0.5	0.0	1000	0
0.5	0.1	1	7
0.5	0.1	5	3
0.5	0.1	10	8
0.5	0.1	50	1

Таблица А.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.5	0.1	100	0
0.5	0.1	500	0
0.5	0.1	1000	0
0.5	0.2	1	19
0.5	0.2	5	4
0.5	0.2	10	1
0.5	0.2	50	0
0.5	0.2	100	0
0.5	0.2	500	0
0.5	0.2	1000	0
0.5	0.3	1	13
0.5	0.3	5	11
0.5	0.3	10	1
0.5	0.3	50	0
0.5	0.3	100	0
0.5	0.3	500	0
0.5	0.3	1000	0
0.5	0.4	1	17
0.5	0.4	5	3
0.5	0.4	10	5
0.5	0.4	50	1
0.5	0.4	100	0
0.5	0.4	500	0
0.5	0.4	1000	0
0.5	0.5	1	10
0.5	0.5	5	9
0.5	0.5	10	0
0.5	0.5	50	0
0.5	0.5	100	0
0.5	0.5	500	0
0.5	0.5	1000	0

Таблица А.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.5	0.6	1	12
0.5	0.6	5	4
0.5	0.6	10	1
0.5	0.6	50	0
0.5	0.6	100	0
0.5	0.6	500	0
0.5	0.6	1000	0
0.5	0.7	1	16
0.5	0.7	5	6
0.5	0.7	10	3
0.5	0.7	50	0
0.5	0.7	100	0
0.5	0.7	500	0
0.5	0.7	1000	0
0.5	0.8	1	21
0.5	0.8	5	3
0.5	0.8	10	5
0.5	0.8	50	0
0.5	0.8	100	0
0.5	0.8	500	0
0.5	0.8	1000	0
0.5	0.9	1	14
0.5	0.9	5	10
0.5	0.9	10	6
0.5	0.9	50	3
0.5	0.9	100	0
0.5	0.9	500	0
0.5	0.9	1000	0
0.5	1.0	1	11
0.5	1.0	5	6
0.5	1.0	10	4

Таблица А.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.5	1.0	50	1
0.5	1.0	100	0
0.5	1.0	500	0
0.5	1.0	1000	0
0.6	0.0	1	6
0.6	0.0	5	7
0.6	0.0	10	4
0.6	0.0	50	0
0.6	0.0	100	0
0.6	0.0	500	0
0.6	0.0	1000	0
0.6	0.1	1	10
0.6	0.1	5	6
0.6	0.1	10	4
0.6	0.1	50	1
0.6	0.1	100	0
0.6	0.1	500	0
0.6	0.1	1000	0
0.6	0.2	1	10
0.6	0.2	5	4
0.6	0.2	10	5
0.6	0.2	50	0
0.6	0.2	100	0
0.6	0.2	500	0
0.6	0.2	1000	0
0.6	0.3	1	9
0.6	0.3	5	4
0.6	0.3	10	1
0.6	0.3	50	1
0.6	0.3	100	0
0.6	0.3	500	0

Таблица А.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.6	0.3	1000	0
0.6	0.4	1	17
0.6	0.4	5	10
0.6	0.4	10	1
0.6	0.4	50	0
0.6	0.4	100	0
0.6	0.4	500	0
0.6	0.4	1000	0
0.6	0.5	1	14
0.6	0.5	5	7
0.6	0.5	10	3
0.6	0.5	50	0
0.6	0.5	100	0
0.6	0.5	500	0
0.6	0.5	1000	0
0.6	0.6	1	5
0.6	0.6	5	5
0.6	0.6	10	1
0.6	0.6	50	1
0.6	0.6	100	1
0.6	0.6	500	0
0.6	0.6	1000	0
0.6	0.7	1	8
0.6	0.7	5	4
0.6	0.7	10	7
0.6	0.7	50	0
0.6	0.7	100	0
0.6	0.7	500	0
0.6	0.7	1000	0
0.6	0.8	1	15
0.6	0.8	5	3

Таблица А.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.6	0.8	10	5
0.6	0.8	50	0
0.6	0.8	100	0
0.6	0.8	500	0
0.6	0.8	1000	0
0.6	0.9	1	14
0.6	0.9	5	0
0.6	0.9	10	2
0.6	0.9	50	0
0.6	0.9	100	0
0.6	0.9	500	0
0.6	0.9	1000	0
0.6	1.0	1	15
0.6	1.0	5	5
0.6	1.0	10	4
0.6	1.0	50	0
0.6	1.0	100	0
0.6	1.0	500	0
0.6	1.0	1000	0
0.7	0.0	1	15
0.7	0.0	5	5
0.7	0.0	10	5
0.7	0.0	50	0
0.7	0.0	100	0
0.7	0.0	500	0
0.7	0.0	1000	0
0.7	0.1	1	14
0.7	0.1	5	7
0.7	0.1	10	5
0.7	0.1	50	0
0.7	0.1	100	0

Таблица А.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.7	0.1	500	0
0.7	0.1	1000	0
0.7	0.2	1	4
0.7	0.2	5	8
0.7	0.2	10	2
0.7	0.2	50	2
0.7	0.2	100	0
0.7	0.2	500	0
0.7	0.2	1000	0
0.7	0.3	1	12
0.7	0.3	5	10
0.7	0.3	10	3
0.7	0.3	50	1
0.7	0.3	100	0
0.7	0.3	500	0
0.7	0.3	1000	0
0.7	0.4	1	19
0.7	0.4	5	9
0.7	0.4	10	7
0.7	0.4	50	1
0.7	0.4	100	0
0.7	0.4	500	0
0.7	0.4	1000	0
0.7	0.5	1	15
0.7	0.5	5	13
0.7	0.5	10	2
0.7	0.5	50	0
0.7	0.5	100	0
0.7	0.5	500	0
0.7	0.5	1000	0
0.7	0.6	1	9

Таблица А.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.7	0.6	5	5
0.7	0.6	10	10
0.7	0.6	50	0
0.7	0.6	100	0
0.7	0.6	500	0
0.7	0.6	1000	0
0.7	0.7	1	15
0.7	0.7	5	10
0.7	0.7	10	5
0.7	0.7	50	2
0.7	0.7	100	0
0.7	0.7	500	0
0.7	0.7	1000	0
0.7	0.8	1	17
0.7	0.8	5	9
0.7	0.8	10	6
0.7	0.8	50	0
0.7	0.8	100	0
0.7	0.8	500	0
0.7	0.8	1000	0
0.7	0.9	1	19
0.7	0.9	5	11
0.7	0.9	10	1
0.7	0.9	50	0
0.7	0.9	100	0
0.7	0.9	500	0
0.7	0.9	1000	0
0.7	1.0	1	15
0.7	1.0	5	4
0.7	1.0	10	3
0.7	1.0	50	1

Таблица А.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.7	1.0	100	0
0.7	1.0	500	0
0.7	1.0	1000	0
0.8	0.0	1	20
0.8	0.0	5	7
0.8	0.0	10	4
0.8	0.0	50	1
0.8	0.0	100	0
0.8	0.0	500	0
0.8	0.0	1000	0
0.8	0.1	1	14
0.8	0.1	5	4
0.8	0.1	10	3
0.8	0.1	50	0
0.8	0.1	100	2
0.8	0.1	500	0
0.8	0.1	1000	0
0.8	0.2	1	13
0.8	0.2	5	2
0.8	0.2	10	5
0.8	0.2	50	3
0.8	0.2	100	1
0.8	0.2	500	0
0.8	0.2	1000	0
0.8	0.3	1	13
0.8	0.3	5	11
0.8	0.3	10	7
0.8	0.3	50	4
0.8	0.3	100	0
0.8	0.3	500	0
0.8	0.3	1000	0

Таблица А.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.8	0.4	1	4
0.8	0.4	5	8
0.8	0.4	10	7
0.8	0.4	50	3
0.8	0.4	100	0
0.8	0.4	500	0
0.8	0.4	1000	0
0.8	0.5	1	23
0.8	0.5	5	4
0.8	0.5	10	5
0.8	0.5	50	3
0.8	0.5	100	3
0.8	0.5	500	0
0.8	0.5	1000	0
0.8	0.6	1	7
0.8	0.6	5	6
0.8	0.6	10	3
0.8	0.6	50	1
0.8	0.6	100	0
0.8	0.6	500	0
0.8	0.6	1000	0
0.8	0.7	1	13
0.8	0.7	5	3
0.8	0.7	10	1
0.8	0.7	50	0
0.8	0.7	100	0
0.8	0.7	500	0
0.8	0.7	1000	0
0.8	0.8	1	13
0.8	0.8	5	14
0.8	0.8	10	1

Таблица А.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.8	0.8	50	1
0.8	0.8	100	0
0.8	0.8	500	0
0.8	0.8	1000	0
0.8	0.9	1	22
0.8	0.9	5	6
0.8	0.9	10	4
0.8	0.9	50	2
0.8	0.9	100	3
0.8	0.9	500	0
0.8	0.9	1000	0
0.8	1.0	1	20
0.8	1.0	5	3
0.8	1.0	10	5
0.8	1.0	50	1
0.8	1.0	100	1
0.8	1.0	500	0
0.8	1.0	1000	0
0.9	0.0	1	12
0.9	0.0	5	12
0.9	0.0	10	5
0.9	0.0	50	10
0.9	0.0	100	1
0.9	0.0	500	0
0.9	0.0	1000	0
0.9	0.1	1	15
0.9	0.1	5	7
0.9	0.1	10	9
0.9	0.1	50	5
0.9	0.1	100	0
0.9	0.1	500	0

Таблица А.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.9	0.1	1000	0
0.9	0.2	1	10
0.9	0.2	5	8
0.9	0.2	10	6
0.9	0.2	50	0
0.9	0.2	100	0
0.9	0.2	500	0
0.9	0.2	1000	0
0.9	0.3	1	16
0.9	0.3	5	7
0.9	0.3	10	6
0.9	0.3	50	1
0.9	0.3	100	0
0.9	0.3	500	0
0.9	0.3	1000	0
0.9	0.4	1	18
0.9	0.4	5	12
0.9	0.4	10	2
0.9	0.4	50	3
0.9	0.4	100	3
0.9	0.4	500	0
0.9	0.4	1000	0
0.9	0.5	1	14
0.9	0.5	5	6
0.9	0.5	10	7
0.9	0.5	50	0
0.9	0.5	100	1
0.9	0.5	500	0
0.9	0.5	1000	0
0.9	0.6	1	10
0.9	0.6	5	14

Таблица А.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.9	0.6	10	8
0.9	0.6	50	1
0.9	0.6	100	0
0.9	0.6	500	0
0.9	0.6	1000	0
0.9	0.7	1	16
0.9	0.7	5	17
0.9	0.7	10	0
0.9	0.7	50	1
0.9	0.7	100	1
0.9	0.7	500	0
0.9	0.7	1000	0
0.9	0.8	1	17
0.9	0.8	5	18
0.9	0.8	10	7
0.9	0.8	50	1
0.9	0.8	100	0
0.9	0.8	500	0
0.9	0.8	1000	0
0.9	0.9	1	16
0.9	0.9	5	3
0.9	0.9	10	5
0.9	0.9	50	3
0.9	0.9	100	0
0.9	0.9	500	0
0.9	0.9	1000	0
0.9	1.0	1	18
0.9	1.0	5	9
0.9	1.0	10	11
0.9	1.0	50	3
0.9	1.0	100	3

Таблица А.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.9	1.0	500	0
0.9	1.0	1000	0
1.0	0.0	1	21
1.0	0.0	5	13
1.0	0.0	10	5
1.0	0.0	50	4
1.0	0.0	100	6
1.0	0.0	500	0
1.0	0.0	1000	0
1.0	0.1	1	17
1.0	0.1	5	15
1.0	0.1	10	11
1.0	0.1	50	1
1.0	0.1	100	2
1.0	0.1	500	0
1.0	0.1	1000	0
1.0	0.2	1	14
1.0	0.2	5	6
1.0	0.2	10	10
1.0	0.2	50	4
1.0	0.2	100	0
1.0	0.2	500	0
1.0	0.2	1000	0
1.0	0.3	1	12
1.0	0.3	5	9
1.0	0.3	10	8
1.0	0.3	50	5
1.0	0.3	100	2
1.0	0.3	500	0
1.0	0.3	1000	0
1.0	0.4	1	7

Таблица А.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
1.0	0.4	5	13
1.0	0.4	10	1
1.0	0.4	50	0
1.0	0.4	100	3
1.0	0.4	500	0
1.0	0.4	1000	0
1.0	0.5	1	20
1.0	0.5	5	6
1.0	0.5	10	13
1.0	0.5	50	1
1.0	0.5	100	0
1.0	0.5	500	0
1.0	0.5	1000	0
1.0	0.6	1	20
1.0	0.6	5	7
1.0	0.6	10	10
1.0	0.6	50	3
1.0	0.6	100	2
1.0	0.6	500	0
1.0	0.6	1000	1
1.0	0.7	1	8
1.0	0.7	5	11
1.0	0.7	10	14
1.0	0.7	50	4
1.0	0.7	100	3
1.0	0.7	500	0
1.0	0.7	1000	0
1.0	0.8	1	11
1.0	0.8	5	8
1.0	0.8	10	10
1.0	0.8	50	3

Таблица А.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
1.0	0.8	100	5
1.0	0.8	500	0
1.0	0.8	1000	0
1.0	0.9	1	8
1.0	0.9	5	9
1.0	0.9	10	11
1.0	0.9	50	0
1.0	0.9	100	2
1.0	0.9	500	0
1.0	0.9	1000	0
1.0	1.0	1	18
1.0	1.0	5	16
1.0	1.0	10	18
1.0	1.0	50	2
1.0	1.0	100	3
1.0	1.0	500	0
1.0	1.0	1000	0

# Приложение Б

## Параметризация для класса данных 2

Таблица Б.1 – Параметризация для  
класса данных с  
большим разбросом  
значений

$\alpha$	$p$	Число итераций	Ошибка
0.0	0.0	1	1277
0.0	0.0	5	112
0.0	0.0	10	112
0.0	0.0	50	0
0.0	0.0	100	0
0.0	0.0	500	0
0.0	0.0	1000	0
0.0	0.1	1	469
0.0	0.1	5	339
0.0	0.1	10	112
0.0	0.1	50	112
0.0	0.1	100	0
0.0	0.1	500	0
0.0	0.1	1000	0
0.0	0.2	1	469
0.0	0.2	5	112
0.0	0.2	10	112
0.0	0.2	50	0
0.0	0.2	100	0
0.0	0.2	500	0
0.0	0.2	1000	0
0.0	0.3	1	1673
0.0	0.3	5	112
0.0	0.3	10	112

Таблица Б.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.0	0.3	50	0
0.0	0.3	100	0
0.0	0.3	500	0
0.0	0.3	1000	0
0.0	0.4	1	509
0.0	0.4	5	112
0.0	0.4	10	524
0.0	0.4	50	112
0.0	0.4	100	0
0.0	0.4	500	0
0.0	0.4	1000	0
0.0	0.5	1	112
0.0	0.5	5	339
0.0	0.5	10	0
0.0	0.5	50	112
0.0	0.5	100	0
0.0	0.5	500	0
0.0	0.5	1000	0
0.0	0.6	1	1439
0.0	0.6	5	339
0.0	0.6	10	112
0.0	0.6	50	0
0.0	0.6	100	0
0.0	0.6	500	0
0.0	0.6	1000	0
0.0	0.7	1	112
0.0	0.7	5	112
0.0	0.7	10	112
0.0	0.7	50	0
0.0	0.7	100	0
0.0	0.7	500	0

Таблица Б.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.0	0.7	1000	0
0.0	0.8	1	469
0.0	0.8	5	469
0.0	0.8	10	0
0.0	0.8	50	0
0.0	0.8	100	0
0.0	0.8	500	0
0.0	0.8	1000	0
0.0	0.9	1	1277
0.0	0.9	5	112
0.0	0.9	10	112
0.0	0.9	50	0
0.0	0.9	100	0
0.0	0.9	500	0
0.0	0.9	1000	0
0.0	1.0	1	1253
0.0	1.0	5	339
0.0	1.0	10	112
0.0	1.0	50	112
0.0	1.0	100	0
0.0	1.0	500	0
0.0	1.0	1000	0
0.1	0.0	1	469
0.1	0.0	5	339
0.1	0.0	10	0
0.1	0.0	50	0
0.1	0.0	100	112
0.1	0.0	500	0
0.1	0.0	1000	0
0.1	0.1	1	905
0.1	0.1	5	112

Таблица Б.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.1	0.1	10	112
0.1	0.1	50	112
0.1	0.1	100	0
0.1	0.1	500	0
0.1	0.1	1000	0
0.1	0.2	1	469
0.1	0.2	5	112
0.1	0.2	10	112
0.1	0.2	50	0
0.1	0.2	100	0
0.1	0.2	500	0
0.1	0.2	1000	0
0.1	0.3	1	469
0.1	0.3	5	509
0.1	0.3	10	112
0.1	0.3	50	0
0.1	0.3	100	0
0.1	0.3	500	0
0.1	0.3	1000	0
0.1	0.4	1	1253
0.1	0.4	5	112
0.1	0.4	10	112
0.1	0.4	50	112
0.1	0.4	100	0
0.1	0.4	500	0
0.1	0.4	1000	0
0.1	0.5	1	112
0.1	0.5	5	112
0.1	0.5	10	639
0.1	0.5	50	0
0.1	0.5	100	0

Таблица Б.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.1	0.5	500	0
0.1	0.5	1000	0
0.1	0.6	1	1702
0.1	0.6	5	112
0.1	0.6	10	112
0.1	0.6	50	112
0.1	0.6	100	0
0.1	0.6	500	0
0.1	0.6	1000	0
0.1	0.7	1	2388
0.1	0.7	5	801
0.1	0.7	10	112
0.1	0.7	50	112
0.1	0.7	100	0
0.1	0.7	500	0
0.1	0.7	1000	0
0.1	0.8	1	112
0.1	0.8	5	669
0.1	0.8	10	339
0.1	0.8	50	0
0.1	0.8	100	0
0.1	0.8	500	0
0.1	0.8	1000	0
0.1	0.9	1	1625
0.1	0.9	5	112
0.1	0.9	10	469
0.1	0.9	50	112
0.1	0.9	100	0
0.1	0.9	500	0
0.1	0.9	1000	0
0.1	1.0	1	112

Таблица Б.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.1	1.0	5	112
0.1	1.0	10	112
0.1	1.0	50	0
0.1	1.0	100	0
0.1	1.0	500	0
0.1	1.0	1000	0
0.2	0.0	1	1678
0.2	0.0	5	0
0.2	0.0	10	112
0.2	0.0	50	0
0.2	0.0	100	0
0.2	0.0	500	0
0.2	0.0	1000	0
0.2	0.1	1	1439
0.2	0.1	5	0
0.2	0.1	10	639
0.2	0.1	50	0
0.2	0.1	100	0
0.2	0.1	500	0
0.2	0.1	1000	0
0.2	0.2	1	1456
0.2	0.2	5	524
0.2	0.2	10	0
0.2	0.2	50	0
0.2	0.2	100	0
0.2	0.2	500	0
0.2	0.2	1000	0
0.2	0.3	1	1192
0.2	0.3	5	112
0.2	0.3	10	0
0.2	0.3	50	112

Таблица Б.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.2	0.3	100	0
0.2	0.3	500	0
0.2	0.3	1000	0
0.2	0.4	1	1277
0.2	0.4	5	0
0.2	0.4	10	469
0.2	0.4	50	112
0.2	0.4	100	0
0.2	0.4	500	0
0.2	0.4	1000	0
0.2	0.5	1	2423
0.2	0.5	5	851
0.2	0.5	10	112
0.2	0.5	50	0
0.2	0.5	100	112
0.2	0.5	500	0
0.2	0.5	1000	0
0.2	0.6	1	2570
0.2	0.6	5	112
0.2	0.6	10	0
0.2	0.6	50	0
0.2	0.6	100	0
0.2	0.6	500	0
0.2	0.6	1000	0
0.2	0.7	1	1277
0.2	0.7	5	726
0.2	0.7	10	0
0.2	0.7	50	112
0.2	0.7	100	0
0.2	0.7	500	0
0.2	0.7	1000	0

Таблица Б.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.2	0.8	1	1559
0.2	0.8	5	112
0.2	0.8	10	112
0.2	0.8	50	0
0.2	0.8	100	0
0.2	0.8	500	0
0.2	0.8	1000	0
0.2	0.9	1	669
0.2	0.9	5	112
0.2	0.9	10	0
0.2	0.9	50	0
0.2	0.9	100	0
0.2	0.9	500	0
0.2	0.9	1000	0
0.2	1.0	1	469
0.2	1.0	5	469
0.2	1.0	10	112
0.2	1.0	50	112
0.2	1.0	100	0
0.2	1.0	500	0
0.2	1.0	1000	0
0.3	0.0	1	1702
0.3	0.0	5	0
0.3	0.0	10	112
0.3	0.0	50	0
0.3	0.0	100	0
0.3	0.0	500	0
0.3	0.0	1000	0
0.3	0.1	1	1350
0.3	0.1	5	748
0.3	0.1	10	112

Таблица Б.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.3	0.1	50	0
0.3	0.1	100	0
0.3	0.1	500	0
0.3	0.1	1000	0
0.3	0.2	1	1652
0.3	0.2	5	112
0.3	0.2	10	0
0.3	0.2	50	112
0.3	0.2	100	0
0.3	0.2	500	0
0.3	0.2	1000	0
0.3	0.3	1	1253
0.3	0.3	5	112
0.3	0.3	10	112
0.3	0.3	50	0
0.3	0.3	100	0
0.3	0.3	500	0
0.3	0.3	1000	0
0.3	0.4	1	1112
0.3	0.4	5	801
0.3	0.4	10	112
0.3	0.4	50	0
0.3	0.4	100	0
0.3	0.4	500	0
0.3	0.4	1000	0
0.3	0.5	1	1315
0.3	0.5	5	0
0.3	0.5	10	296
0.3	0.5	50	112
0.3	0.5	100	0
0.3	0.5	500	0

Таблица Б.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.3	0.5	1000	0
0.3	0.6	1	2397
0.3	0.6	5	112
0.3	0.6	10	0
0.3	0.6	50	0
0.3	0.6	100	0
0.3	0.6	500	0
0.3	0.6	1000	0
0.3	0.7	1	112
0.3	0.7	5	112
0.3	0.7	10	112
0.3	0.7	50	0
0.3	0.7	100	0
0.3	0.7	500	0
0.3	0.7	1000	0
0.3	0.8	1	1501
0.3	0.8	5	469
0.3	0.8	10	296
0.3	0.8	50	112
0.3	0.8	100	0
0.3	0.8	500	0
0.3	0.8	1000	0
0.3	0.9	1	3625
0.3	0.9	5	748
0.3	0.9	10	0
0.3	0.9	50	112
0.3	0.9	100	0
0.3	0.9	500	0
0.3	0.9	1000	0
0.3	1.0	1	469
0.3	1.0	5	469

Таблица Б.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.3	1.0	10	112
0.3	1.0	50	0
0.3	1.0	100	112
0.3	1.0	500	0
0.3	1.0	1000	0
0.4	0.0	1	919
0.4	0.0	5	112
0.4	0.0	10	469
0.4	0.0	50	0
0.4	0.0	100	0
0.4	0.0	500	0
0.4	0.0	1000	0
0.4	0.1	1	3192
0.4	0.1	5	0
0.4	0.1	10	0
0.4	0.1	50	112
0.4	0.1	100	0
0.4	0.1	500	0
0.4	0.1	1000	0
0.4	0.2	1	2092
0.4	0.2	5	851
0.4	0.2	10	0
0.4	0.2	50	0
0.4	0.2	100	0
0.4	0.2	500	0
0.4	0.2	1000	0
0.4	0.3	1	2702
0.4	0.3	5	524
0.4	0.3	10	0
0.4	0.3	50	112
0.4	0.3	100	0

Таблица Б.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.4	0.3	500	0
0.4	0.3	1000	0
0.4	0.4	1	3046
0.4	0.4	5	546
0.4	0.4	10	112
0.4	0.4	50	0
0.4	0.4	100	0
0.4	0.4	500	0
0.4	0.4	1000	0
0.4	0.5	1	1559
0.4	0.5	5	0
0.4	0.5	10	112
0.4	0.5	50	0
0.4	0.5	100	0
0.4	0.5	500	0
0.4	0.5	1000	0
0.4	0.6	1	0
0.4	0.6	5	112
0.4	0.6	10	339
0.4	0.6	50	0
0.4	0.6	100	0
0.4	0.6	500	0
0.4	0.6	1000	0
0.4	0.7	1	1952
0.4	0.7	5	339
0.4	0.7	10	0
0.4	0.7	50	0
0.4	0.7	100	0
0.4	0.7	500	0
0.4	0.7	1000	0
0.4	0.8	1	1861

Таблица Б.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.4	0.8	5	509
0.4	0.8	10	0
0.4	0.8	50	0
0.4	0.8	100	0
0.4	0.8	500	0
0.4	0.8	1000	0
0.4	0.9	1	639
0.4	0.9	5	546
0.4	0.9	10	0
0.4	0.9	50	112
0.4	0.9	100	0
0.4	0.9	500	0
0.4	0.9	1000	0
0.4	1.0	1	1847
0.4	1.0	5	112
0.4	1.0	10	112
0.4	1.0	50	0
0.4	1.0	100	0
0.4	1.0	500	0
0.4	1.0	1000	0
0.5	0.0	1	2022
0.5	0.0	5	112
0.5	0.0	10	469
0.5	0.0	50	0
0.5	0.0	100	0
0.5	0.0	500	0
0.5	0.0	1000	0
0.5	0.1	1	4148
0.5	0.1	5	112
0.5	0.1	10	112
0.5	0.1	50	0

Таблица Б.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.5	0.1	100	112
0.5	0.1	500	0
0.5	0.1	1000	0
0.5	0.2	1	2046
0.5	0.2	5	112
0.5	0.2	10	112
0.5	0.2	50	0
0.5	0.2	100	112
0.5	0.2	500	0
0.5	0.2	1000	0
0.5	0.3	1	3409
0.5	0.3	5	905
0.5	0.3	10	112
0.5	0.3	50	0
0.5	0.3	100	112
0.5	0.3	500	0
0.5	0.3	1000	0
0.5	0.4	1	3478
0.5	0.4	5	296
0.5	0.4	10	524
0.5	0.4	50	0
0.5	0.4	100	0
0.5	0.4	500	0
0.5	0.4	1000	0
0.5	0.5	1	2985
0.5	0.5	5	469
0.5	0.5	10	112
0.5	0.5	50	0
0.5	0.5	100	0
0.5	0.5	500	0
0.5	0.5	1000	0

Таблица Б.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.5	0.6	1	1230
0.5	0.6	5	919
0.5	0.6	10	112
0.5	0.6	50	0
0.5	0.6	100	0
0.5	0.6	500	0
0.5	0.6	1000	0
0.5	0.7	1	3191
0.5	0.7	5	0
0.5	0.7	10	112
0.5	0.7	50	112
0.5	0.7	100	0
0.5	0.7	500	0
0.5	0.7	1000	0
0.5	0.8	1	3313
0.5	0.8	5	0
0.5	0.8	10	112
0.5	0.8	50	0
0.5	0.8	100	0
0.5	0.8	500	0
0.5	0.8	1000	0
0.5	0.9	1	1678
0.5	0.9	5	112
0.5	0.9	10	112
0.5	0.9	50	112
0.5	0.9	100	112
0.5	0.9	500	0
0.5	0.9	1000	0
0.5	1.0	1	4544
0.5	1.0	5	112
0.5	1.0	10	669

Таблица Б.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.5	1.0	50	0
0.5	1.0	100	0
0.5	1.0	500	0
0.5	1.0	1000	0
0.6	0.0	1	3065
0.6	0.0	5	1625
0.6	0.0	10	546
0.6	0.0	50	0
0.6	0.0	100	112
0.6	0.0	500	0
0.6	0.0	1000	0
0.6	0.1	1	469
0.6	0.1	5	524
0.6	0.1	10	726
0.6	0.1	50	112
0.6	0.1	100	0
0.6	0.1	500	0
0.6	0.1	1000	0
0.6	0.2	1	2609
0.6	0.2	5	1861
0.6	0.2	10	339
0.6	0.2	50	0
0.6	0.2	100	112
0.6	0.2	500	0
0.6	0.2	1000	0
0.6	0.3	1	2092
0.6	0.3	5	509
0.6	0.3	10	339
0.6	0.3	50	112
0.6	0.3	100	0
0.6	0.3	500	0

Таблица Б.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.6	0.3	1000	0
0.6	0.4	1	2725
0.6	0.4	5	469
0.6	0.4	10	339
0.6	0.4	50	0
0.6	0.4	100	0
0.6	0.4	500	0
0.6	0.4	1000	0
0.6	0.5	1	2120
0.6	0.5	5	112
0.6	0.5	10	469
0.6	0.5	50	112
0.6	0.5	100	0
0.6	0.5	500	0
0.6	0.5	1000	0
0.6	0.6	1	1611
0.6	0.6	5	0
0.6	0.6	10	509
0.6	0.6	50	0
0.6	0.6	100	0
0.6	0.6	500	0
0.6	0.6	1000	0
0.6	0.7	1	3631
0.6	0.7	5	469
0.6	0.7	10	112
0.6	0.7	50	0
0.6	0.7	100	0
0.6	0.7	500	0
0.6	0.7	1000	0
0.6	0.8	1	2877
0.6	0.8	5	469

Таблица Б.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.6	0.8	10	112
0.6	0.8	50	112
0.6	0.8	100	0
0.6	0.8	500	0
0.6	0.8	1000	0
0.6	0.9	1	3573
0.6	0.9	5	339
0.6	0.9	10	112
0.6	0.9	50	0
0.6	0.9	100	0
0.6	0.9	500	0
0.6	0.9	1000	0
0.6	1.0	1	3573
0.6	1.0	5	469
0.6	1.0	10	469
0.6	1.0	50	112
0.6	1.0	100	0
0.6	1.0	500	0
0.6	1.0	1000	0
0.7	0.0	1	3574
0.7	0.0	5	2069
0.7	0.0	10	112
0.7	0.0	50	112
0.7	0.0	100	0
0.7	0.0	500	0
0.7	0.0	1000	0
0.7	0.1	1	1726
0.7	0.1	5	1771
0.7	0.1	10	546
0.7	0.1	50	112
0.7	0.1	100	0

Таблица Б.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.7	0.1	500	0
0.7	0.1	1000	0
0.7	0.2	1	4130
0.7	0.2	5	509
0.7	0.2	10	339
0.7	0.2	50	112
0.7	0.2	100	0
0.7	0.2	500	0
0.7	0.2	1000	0
0.7	0.3	1	1900
0.7	0.3	5	1618
0.7	0.3	10	801
0.7	0.3	50	0
0.7	0.3	100	0
0.7	0.3	500	0
0.7	0.3	1000	0
0.7	0.4	1	905
0.7	0.4	5	546
0.7	0.4	10	919
0.7	0.4	50	112
0.7	0.4	100	0
0.7	0.4	500	0
0.7	0.4	1000	0
0.7	0.5	1	3543
0.7	0.5	5	1674
0.7	0.5	10	782
0.7	0.5	50	0
0.7	0.5	100	0
0.7	0.5	500	0
0.7	0.5	1000	0
0.7	0.6	1	5073

Таблица Б.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.7	0.6	5	339
0.7	0.6	10	919
0.7	0.6	50	112
0.7	0.6	100	112
0.7	0.6	500	0
0.7	0.6	1000	0
0.7	0.7	1	4409
0.7	0.7	5	1260
0.7	0.7	10	469
0.7	0.7	50	112
0.7	0.7	100	0
0.7	0.7	500	0
0.7	0.7	1000	0
0.7	0.8	1	2365
0.7	0.8	5	1726
0.7	0.8	10	112
0.7	0.8	50	112
0.7	0.8	100	112
0.7	0.8	500	0
0.7	0.8	1000	0
0.7	0.9	1	2159
0.7	0.9	5	1989
0.7	0.9	10	112
0.7	0.9	50	0
0.7	0.9	100	0
0.7	0.9	500	0
0.7	0.9	1000	0
0.7	1.0	1	3180
0.7	1.0	5	112
0.7	1.0	10	795
0.7	1.0	50	112

Таблица Б.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.7	1.0	100	0
0.7	1.0	500	0
0.7	1.0	1000	0
0.8	0.0	1	3841
0.8	0.0	5	509
0.8	0.0	10	1089
0.8	0.0	50	112
0.8	0.0	100	112
0.8	0.0	500	0
0.8	0.0	1000	0
0.8	0.1	1	2716
0.8	0.1	5	464
0.8	0.1	10	2067
0.8	0.1	50	112
0.8	0.1	100	0
0.8	0.1	500	0
0.8	0.1	1000	0
0.8	0.2	1	5336
0.8	0.2	5	669
0.8	0.2	10	296
0.8	0.2	50	296
0.8	0.2	100	0
0.8	0.2	500	0
0.8	0.2	1000	0
0.8	0.3	1	3363
0.8	0.3	5	2648
0.8	0.3	10	509
0.8	0.3	50	0
0.8	0.3	100	0
0.8	0.3	500	0
0.8	0.3	1000	0

Таблица Б.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.8	0.4	1	524
0.8	0.4	5	669
0.8	0.4	10	509
0.8	0.4	50	469
0.8	0.4	100	0
0.8	0.4	500	0
0.8	0.4	1000	0
0.8	0.5	1	1306
0.8	0.5	5	112
0.8	0.5	10	469
0.8	0.5	50	112
0.8	0.5	100	112
0.8	0.5	500	0
0.8	0.5	1000	0
0.8	0.6	1	3386
0.8	0.6	5	801
0.8	0.6	10	469
0.8	0.6	50	112
0.8	0.6	100	112
0.8	0.6	500	0
0.8	0.6	1000	0
0.8	0.7	1	4463
0.8	0.7	5	1678
0.8	0.7	10	851
0.8	0.7	50	112
0.8	0.7	100	112
0.8	0.7	500	0
0.8	0.7	1000	0
0.8	0.8	1	2711
0.8	0.8	5	905
0.8	0.8	10	296

Таблица Б.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.8	0.8	50	112
0.8	0.8	100	112
0.8	0.8	500	0
0.8	0.8	1000	0
0.8	0.9	1	6029
0.8	0.9	5	1847
0.8	0.9	10	801
0.8	0.9	50	0
0.8	0.9	100	112
0.8	0.9	500	112
0.8	0.9	1000	0
0.8	1.0	1	2096
0.8	1.0	5	870
0.8	1.0	10	112
0.8	1.0	50	0
0.8	1.0	100	0
0.8	1.0	500	112
0.8	1.0	1000	0
0.9	0.0	1	4575
0.9	0.0	5	2088
0.9	0.0	10	1192
0.9	0.0	50	112
0.9	0.0	100	509
0.9	0.0	500	0
0.9	0.0	1000	0
0.9	0.1	1	3346
0.9	0.1	5	3017
0.9	0.1	10	1547
0.9	0.1	50	112
0.9	0.1	100	296
0.9	0.1	500	0

Таблица Б.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.9	0.1	1000	0
0.9	0.2	1	3763
0.9	0.2	5	2131
0.9	0.2	10	339
0.9	0.2	50	0
0.9	0.2	100	296
0.9	0.2	500	0
0.9	0.2	1000	0
0.9	0.3	1	6288
0.9	0.3	5	1192
0.9	0.3	10	919
0.9	0.3	50	112
0.9	0.3	100	112
0.9	0.3	500	0
0.9	0.3	1000	0
0.9	0.4	1	2469
0.9	0.4	5	1408
0.9	0.4	10	524
0.9	0.4	50	339
0.9	0.4	100	112
0.9	0.4	500	0
0.9	0.4	1000	0
0.9	0.5	1	4236
0.9	0.5	5	857
0.9	0.5	10	2136
0.9	0.5	50	112
0.9	0.5	100	0
0.9	0.5	500	0
0.9	0.5	1000	0
0.9	0.6	1	5367
0.9	0.6	5	3184

Таблица Б.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.9	0.6	10	1717
0.9	0.6	50	112
0.9	0.6	100	112
0.9	0.6	500	0
0.9	0.6	1000	0
0.9	0.7	1	3913
0.9	0.7	5	1245
0.9	0.7	10	1768
0.9	0.7	50	112
0.9	0.7	100	112
0.9	0.7	500	0
0.9	0.7	1000	0
0.9	0.8	1	1674
0.9	0.8	5	1920
0.9	0.8	10	795
0.9	0.8	50	726
0.9	0.8	100	0
0.9	0.8	500	0
0.9	0.8	1000	0
0.9	0.9	1	3964
0.9	0.9	5	801
0.9	0.9	10	976
0.9	0.9	50	339
0.9	0.9	100	0
0.9	0.9	500	0
0.9	0.9	1000	0
0.9	1.0	1	2309
0.9	1.0	5	3347
0.9	1.0	10	2136
0.9	1.0	50	112
0.9	1.0	100	112

Таблица Б.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
0.9	1.0	500	112
0.9	1.0	1000	0
1.0	0.0	1	4095
1.0	0.0	5	296
1.0	0.0	10	3543
1.0	0.0	50	1934
1.0	0.0	100	112
1.0	0.0	500	0
1.0	0.0	1000	0
1.0	0.1	1	4623
1.0	0.1	5	2060
1.0	0.1	10	2807
1.0	0.1	50	524
1.0	0.1	100	1059
1.0	0.1	500	0
1.0	0.1	1000	0
1.0	0.2	1	5783
1.0	0.2	5	1155
1.0	0.2	10	1636
1.0	0.2	50	2178
1.0	0.2	100	509
1.0	0.2	500	112
1.0	0.2	1000	0
1.0	0.3	1	6862
1.0	0.3	5	4741
1.0	0.3	10	2127
1.0	0.3	50	509
1.0	0.3	100	296
1.0	0.3	500	0
1.0	0.3	1000	0
1.0	0.4	1	2034

Таблица Б.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
1.0	0.4	5	2266
1.0	0.4	10	1951
1.0	0.4	50	1952
1.0	0.4	100	0
1.0	0.4	500	0
1.0	0.4	1000	112
1.0	0.5	1	5142
1.0	0.5	5	1744
1.0	0.5	10	296
1.0	0.5	50	112
1.0	0.5	100	726
1.0	0.5	500	0
1.0	0.5	1000	0
1.0	0.6	1	5145
1.0	0.6	5	2119
1.0	0.6	10	1089
1.0	0.6	50	1192
1.0	0.6	100	509
1.0	0.6	500	112
1.0	0.6	1000	0
1.0	0.7	1	3180
1.0	0.7	5	2366
1.0	0.7	10	2561
1.0	0.7	50	1573
1.0	0.7	100	339
1.0	0.7	500	0
1.0	0.7	1000	0
1.0	0.8	1	4301
1.0	0.8	5	3529
1.0	0.8	10	2636
1.0	0.8	50	1673

Таблица Б.1 (продолжение)

$\alpha$	$p$	Число итераций	Ошибка
1.0	0.8	100	524
1.0	0.8	500	0
1.0	0.8	1000	0
1.0	0.9	1	4161
1.0	0.9	5	2191
1.0	0.9	10	2708
1.0	0.9	50	0
1.0	0.9	100	0
1.0	0.9	500	112
1.0	0.9	1000	0
1.0	1.0	1	3881
1.0	1.0	5	2944
1.0	1.0	10	1112
1.0	1.0	50	857
1.0	1.0	100	795
1.0	1.0	500	546
1.0	1.0	1000	0