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 №7 по курсу "Анализ алгоритмов"

Тема Методы решения задачи коммивояжера

Студент Котляров Н.А.

Группа ИУ7-51Б

Оценка (баллы)

Преподаватель Волкова Л.Л.

Москва — 2022 г.

Оглавление

Введение

Целью данной лабораторной работы является описание методов решения задачи коммивояжера на основе муравьиного алгоритма. Для достижения поставленной цели необходимо выполнить следующие задачи:

– описать задачу коммивояжера;

– описать методы решения задачи коммивояжера — метод полного перебора и метод на основе муравьиного алгоритма;

– реализовать выбранные алгоритмы;

– провести функциональное тестирование разработанных алгоритмов;

– исследовать время работы реализаций алгоритмов.

1 Аналитическая часть

В этом разделе представляется описание задачи коммивояжера, а также способы их решения.

1.1 Задача коммивояжера

Задача коммивояжёра (англ. *traveling salesman problem*) — (задача о бродячем торговце) одна из самых важных задач всей транспортной логистики, в которой рассматриваются вершины графа, а также матрица смежности (для расстояния между вершинами) [1]. Задача заключается в том, чтобы найти такой порядок посещения вершин графа, при котором путь будет минимален, каждая вершина будет посещена лишь один раз, а возврат произойдет в начальную вершину.

1.2 Метод полного перебора для решения задачи коммивояжёра

Полный перебор для задачи коммивояжёра [2] имеет высокую сложность алгоритма ($n!$), где n — количество городов. Суть в полном переборе всех возможных путей в графе и выбор наименьшего из них. Решение будет получено, но имеются большие затраты по времени выполнения при уже небольшом количестве вершин в графе.

1.3 Метод на основе муравьиного алгоритма

Муравьиный алгоритм (англ. *ant colony optimization*) [2] — метод решения задачи оптимизации, основаный на принципе поведения колонии муравьев.

Муравьи действуют, руководствуясь органами чувств. Каждый муравей оставляет на своём пути феромоны, чтобы другие могли ориентироваться. При большом количестве муравьев наибольшее количество феромона остаётся на наиболее посещаемом пути, посещаемость же может быть связана с длинами рёбер.

Суть в том, что отдельно взятый муравей мало что может, поскольку он способен выполнять только максимально простые задачи. Но при большом числе других таких муравьев они могут выступать самостоятельными вычислительными единицами. Муравьи используют непрямой обмен информацией через окружающую среду посредством феромона.

Пусть муравей имеет следующие характеристики:

1. зрение — способность определить длину ребра;
2. память — способность запомнить пройденный маршрут;
3. обоняние — способность чуять феромон.

Также введем целевую функцию (1.1), характеризующую привлекательность ребра, определяемую благодаря зрению.

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

где D_{ij} — расстояние от текущего пункта i до заданного пункта j.

Также понадобится формула вычисления вероятности перехода в заданную точку (1.2).

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

где a — параметр влияния длины пути, b — параметр влияния феромона, τ_{ij} — количество феромонов на ребре ij, η_{ij} — привлекательность ребра ij, J_k — список посещённых за текущий день городов.

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

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

При этом

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

где

$$\Delta\tau_{ij}^{k}(t) = \begin{cases} Q/L_k, \text{ребро посещено муравьем } k \text{ в текущий день } t, \\ 0, \text{иначе} \end{cases} \tag{1.5}$$

Поскольку вероятность перехода в заданную точку 1.2 не должна быть равна нулю, необходимо обеспечить неравенство $\tau_{ij}(t)$ нулю посредством введения дополительного минимально возможного значения феромона τ_{min} и в случае, если $\tau_{ij}(t+1)$ принимает значение, меньшее τ_{min}, откатывать феромон до этой величины.

Путь выбирается по следующей схеме.

1. Каждый муравей имеет список запретов — список уже посещенных городов (вершин графа).

2. Муравьиное зрение отвечает за эвристическое желание посетить вершину.

3. Муравьиное обоняние отвечает за ощущение феромона на определенном пути (ребре). При этом количество феромона на пути (ребре) в день t обозначается как $\tau_{i,j}(t)$.

4. После прохождения определенного ребра муравей откладывает на нем некотрое количество феромона, которое показывает оптимальность сделанного выбора, это количество вычисляется по формуле (1.5).

Вывод

В данном разделе была рассмотрена задача коммивояжёра, а также полный перебор для её решения и муравьиный алгоритм.

2 Конструкторская часть

В этом разделе будут приведены схемы алгоритмов и описаны используемые типы данных.

2.1 Описание используемых типов данных

При реализации алгоритмов будут использоваться следующие типы данных:

- размер матрицы смежности — целое число;
- имя файла — строка;
- коэффициенты α, β, $k\_evaporation$ — действительные числа;
- матрица смежности — матрица целых чисел.

2.2 Разработка алгоритмов

На рисунке 2.1 представлена схема алгоритма полного перебора.

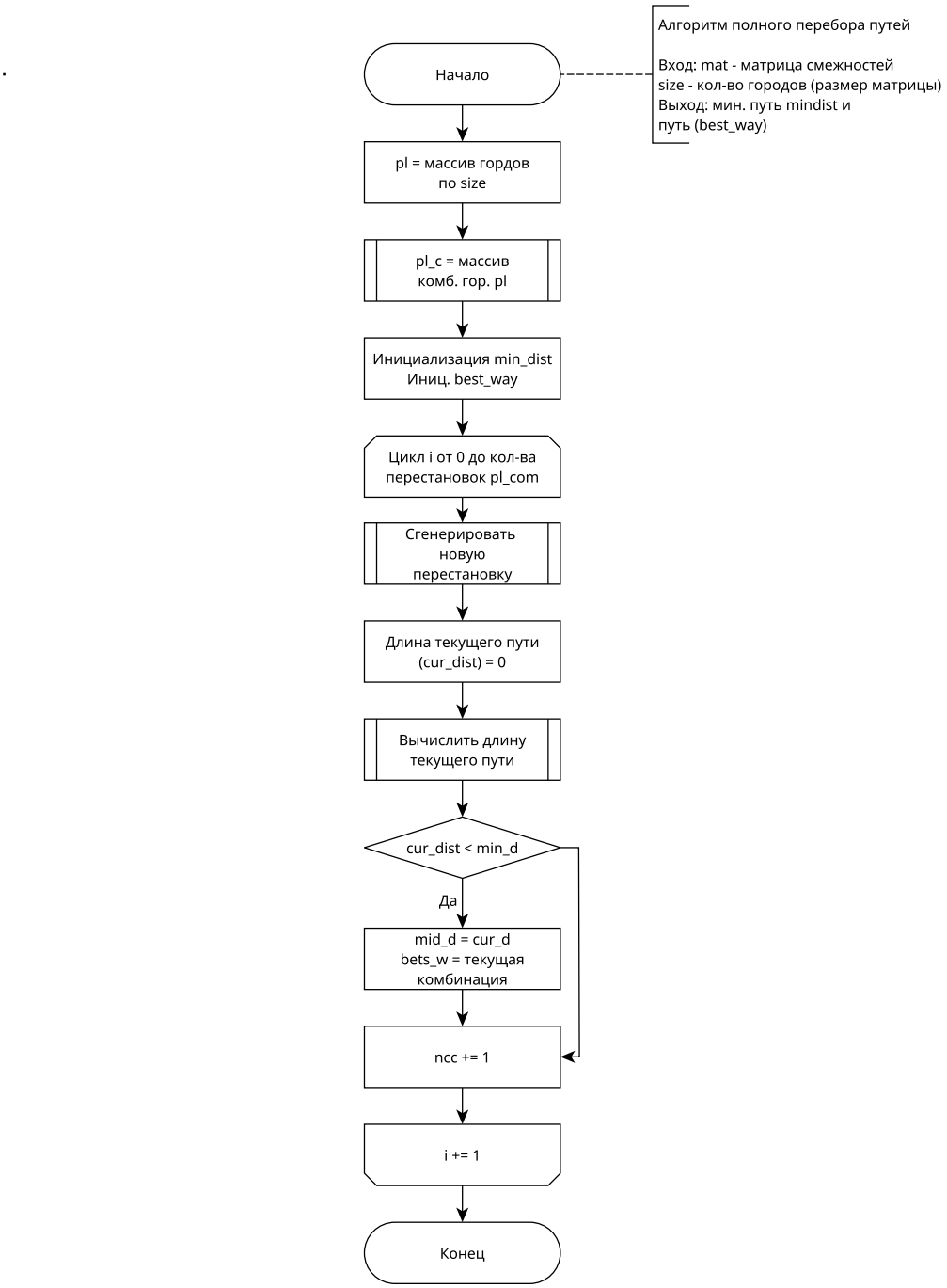

Рисунок 2.1 – Схема алгоритма полного перебора поиска путей

На рисунке 2.2 представлена схема муравьиного алгоритма.

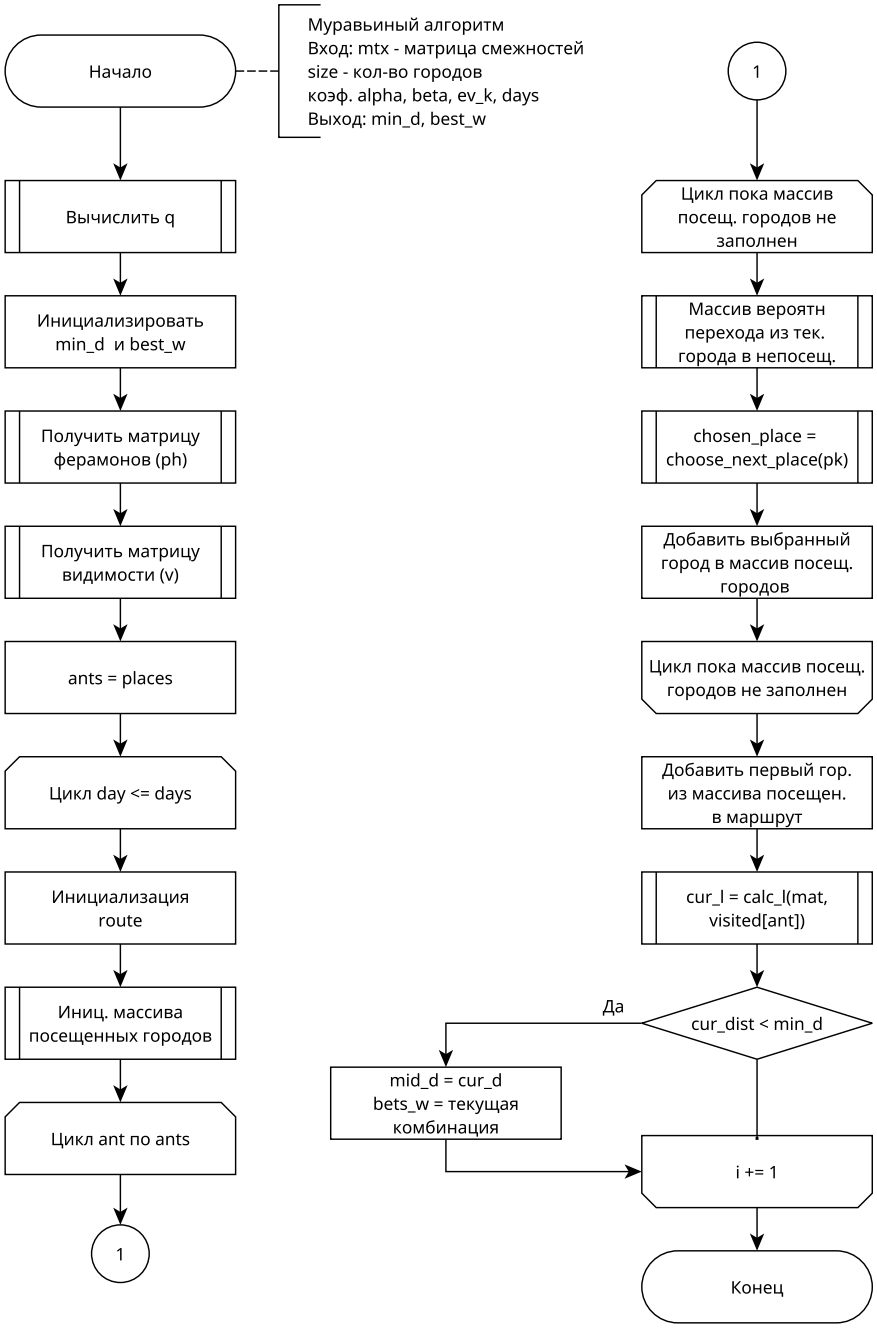

Рисунок 2.2 – Схема муравьиного алгоритма умножения поиска путей

Вывод

Были разработаны схемы алгоритмов поиска путей (полный перебор и муравьиный).

3 Технологическая часть

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

3.1 Требования к программному обеспечению

К программе предъявляется ряд требований:

– на вход подается имя файла с матрицей смежности городов;

– программа определяет количество дней и коэффициенты;

– возможно измерение реального времени;

3.2 Средства реализации

В качестве языка программирования для реализации данной лабораторной работы был выбран язык программирования Python [3].

В данном языке есть все требующиеся для данной лабораторной инструменты разработки.

Время работы реализаций алгоритмов было замерено с помощью функции process_time() из библиотеки time [4]

3.3 Реализация алгоритмов

В листинге 3.1 представлена реализация алгоритма полного перебора путей.

Листинг 3.1 – Реализация последовательного алгоритма умножения разреженных матриц

```
def fullCombinationAlg(matrix, size):
    places = np.arange(size)
    placesCombinations = list()

    for combination in it.permutations(places):
        combArr = list(combination)
        placesCombinations.append(combArr)

    minDist = float("inf")

    for i in range(len(placesCombinations)):
        placesCombinations[i].append(placesCombinations[i][0])
        curDist = 0
        for j in range(size):
            startCity = placesCombinations[i][j]
            endCity = placesCombinations[i][j + 1]
            curDist += matrix[startCity][endCity]

        if (curDist < minDist):
            minDist = curDist
            bestWay = placesCombinations[i]

    return minDist, bestWay
```

В листинге 3.2 представлена реализация муравьиного алгоритма поиска путей.

Листинг 3.2 – Реализация муравьиного алгоритма поиска путей

```
def calcQ(matrix, size):
    q = 0
    count = 0
    for i in range(size):
        for j in range(size):
            if (i != j):
                q += matrix[i][j]
                count += 1
    return q / count

def calcPheromones(size):
    min_phero = 1
    pheromones = [[min_phero for i in range(size)] for j in
```

```
        range(size)]
14      return pheromones
15
16
17  def calcVisibility(matrix, size):
18      visibility = [[(1.0 / matrix[i][j] if (i != j) else 0) for j
            in range(size)] for i in range(size)]
19      return visibility
20
21
22  def calcVisitedPlaces(route, ants):
23      visited = [list() for _ in range(ants)]
24      for ant in range(ants):
25          visited[ant].append(route[ant])
26      return visited
27
28
29  def calcLength(matrix, route):
30      length = 0
31      for way_len in range(1, len(route)):
32          length += matrix[route[way_len - 1], route[way_len]]
33      return length
34
35
36  def updatePheromones(matrix, places, visited, pheromones, q,
        k_evaporation):
37      ants = places
38      for i in range(places):
39          for j in range(places):
40              delta = 0
41              for ant in range(ants):
42                  length = calcLength(matrix, visited[ant])
43                  delta += q / length
44              pheromones[i][j] *= (1 - k_evaporation)
45              pheromones[i][j] += delta
46              if (pheromones[i][j] < MIN_PHEROMONE):
47                  pheromones[i][j] = MIN_PHEROMONE
48      return pheromones
49
50
51  def findWays(pheromones, visibility, visited, places, ant, alpha,
```

```
    beta):
52      pk = [0] * places
53      for place in range(places):
54          if place not in visited[ant]:
55              ant_place = visited[ant][-1]
56              pk[place] = pow(pheromones[ant_place][place], alpha)
                    * \
57                  pow(visibility[ant_place][place], beta)
58          else:
59              pk[place] = 0
60      sum_pk = sum(pk)
61      for place in range(places):
62          pk[place] /= sum_pk
63      return pk
64
65
66  def chooseNextPlaceByPosibility(pk):
67      posibility = random()
68      choice = 0
69      chosenPlace = 0
70      while ((choice < posibility) and (chosenPlace < len(pk))):
71          choice += pk[chosenPlace]
72          chosenPlace += 1
73      return chosenPlace
74
75
76
77  def antAlgorithm(matrix, places, alpha, beta, k_evaporation,
        days):
78      q = calcQ(matrix, places)
79      bestWay = []
80      minDist = float("inf")
81      pheromones = calcPheromones(places)
82      visibility = calcVisibility(matrix, places)
83      ants = places
84      for day in range(days):
85          route = np.arange(places)
86          visited = calcVisitedPlaces(route, ants)
87          for ant in range(ants):
88              while (len(visited[ant]) != ants):
89                  pk = findWays(pheromones, visibility, visited,
```

```
                    places, ant, alpha, beta)
90                  chosenPlace = chooseNextPlaceByPosibility(pk)
91                  visited[ant].append(chosenPlace - 1)
92              visited[ant].append(visited[ant][0])
93              curLength = calcLength(matrix, visited[ant])
94              if (curLength < minDist):
95                  minDist = curLength
96                  bestWay = visited[ant]
97          pheromones = updatePheromones(matrix, places, visited,
                pheromones, q, k_evaporation)
98      return minDist, bestWay
```

3.4 Функциональные тесты

В таблице 3.1 приведены тесты для функций, реализующих алгоритмы сортировки.

Матрица смежности	Ожидаемый результат	Результат программы
$\begin{pmatrix} 0 & 1 & 2 \\ 1 & 0 & 1 \\ 2 & 1 & 0 \end{pmatrix}$	4, [0, 1, 2, 0]	4, [0, 1, 2, 0]
$\begin{pmatrix} 0 & 15 & 19 & 20 \\ 15 & 0 & 12 & 13 \\ 19 & 12 & 0 & 17 \\ 20 & 13 & 17 & 0 \end{pmatrix}$	64, [0, 1, 2, 3, 0]	64, [0, 1, 2, 3, 0]
$\begin{pmatrix} 0 & 4 & 2 & 1 & 7 \\ 4 & 0 & 3 & 7 & 2 \\ 2 & 3 & 0 & 10 & 3 \\ 1 & 7 & 10 & 0 & 9 \\ 7 & 2 & 3 & 9 & 0 \end{pmatrix}$	15, [0, 2, 4, 1, 3, 0]	15, [0, 2, 4, 1, 3, 0]

Таблица 3.1 – Функциональные тесты

Вывод

Написано и протестировано программное обеспечение для поставленной задачи.

4 Исследовательская часть

В данном разделе будут приведены примеры работы программ, постановка эксперимента и сравнительный анализ алгоритмов на основе полученных данных.

4.1 Технические характеристики

Технические характеристики устройства, на котором выполнялось тестирование:

- операционная система: Manjaro xfce [5] Linux [6] x86_64;
- память — 8 Гб;
- мобильный процессор AMD Ryzen™ 7 3700U @ 2.3 ГГц [7].

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

4.2 Демонстрация работы программы

На рисунке 4.1 представлен результат работы программы.

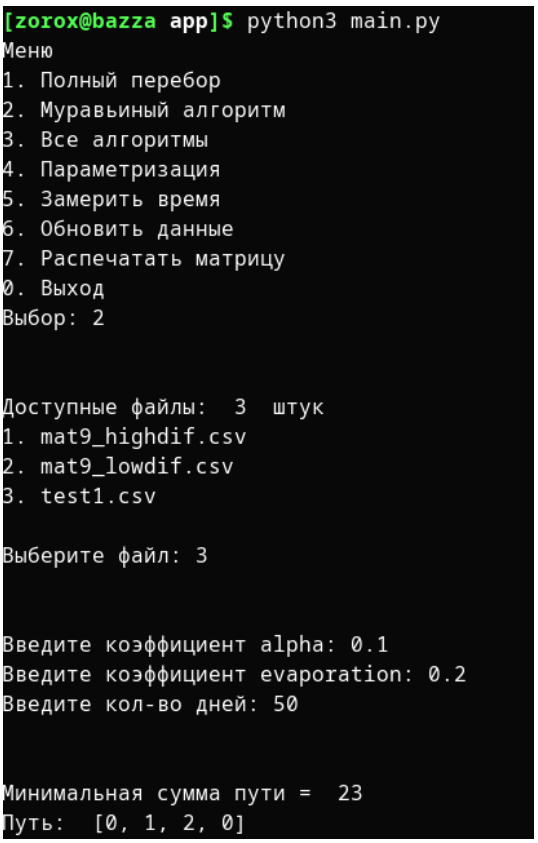

Рисунок 4.1 – Пример работы программы

4.3 Время выполнения реализации алгоритмов

Для замера процессорного времени используется функция *process_ time()* из библиотеки *time* на *Python*. Функция возвращает процессорное время типа float в секундах.

Использовать функцию приходится дважды, затем из конечного времени нужно вычесть начальное, чтобы получить результат.

Замеры проводились для разного размера матриц, чтобы определить, когда наиболее эффективно использовать муравьиный алгоритм.

Результаты замеров приведены в таблице 4.1 (время в с).

Таблица 4.1 – Результаты замеров времени

Размер	Полный перебор	Муравьиный
2	0.000130	0.019932
3	0.000138	0.031615
4	0.000104	0.044361
5	0.000420	0.089291
6	0.002390	0.152131
7	0.019703	0.254059
8	0.162850	0.398472
9	1.637611	0.594024
10	18.207853	0.857666

Также на рисунке 4.2 приведены графические результаты замеров.

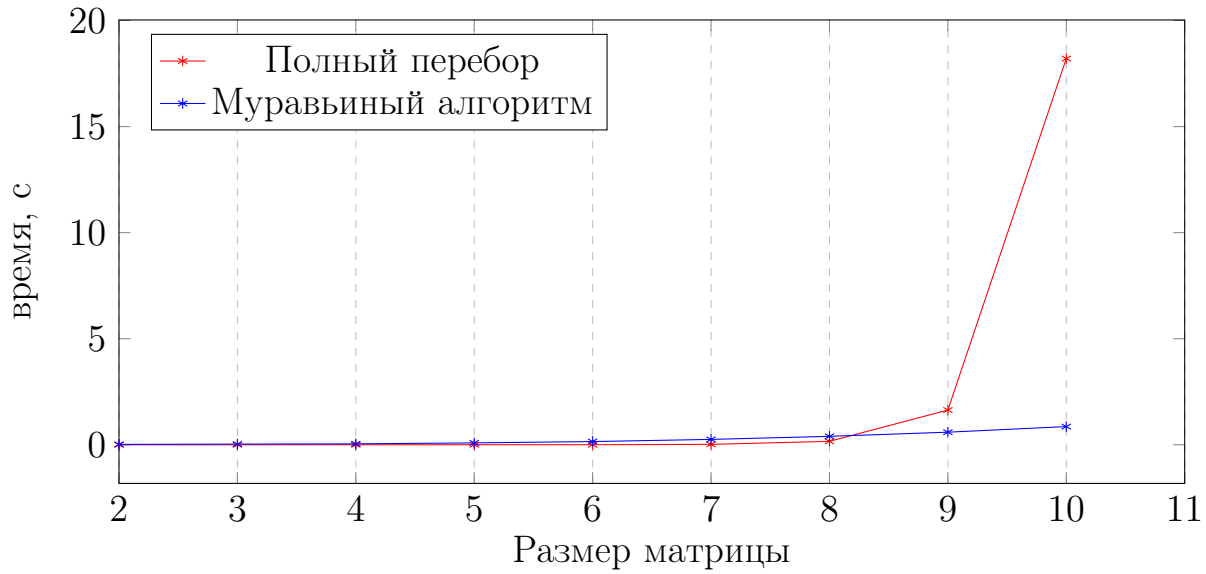

Рисунок 4.2 – Сравнение по времени алгоритмов полного перебора путей и муравьиного на разных размерах матриц

4.4 Постановка эксперимента

Автоматическая параметризация была проведена на двух классах данных — 4.4.1 и 4.4.2. Алгоритм будет запущен для набора значений $\alpha, \rho \in (0, 1)$.

Итоговая таблица значений параметризации будет состоять из следующих колонок:

— α — коэффициент жадности;

— ρ — коэффициент испарения;

— *days* — количество дней жизни колонии муравьёв;

— *Result* — эталонный результат, полученный методом полного перебора для проведения данного эксперимента;

— *Mistake* — разность полученного основаным на муравьином алгоритме методом значения и эталонного значения на данных значениях параметров, показатель качества решения.

Цель эксперимента — определить комбинацию параметров, которые позволяют решаить задачу наилучшим образом для выбранного класса данных. Качество решения зависит от количества дней и погрешности измерений.

4.4.1 Класс данных 1

Класс данных 1 представляет собой матрицу смежности размером 9 элементов (небольшой разброс значений — от 1 до 2), которая представлена далее.

$$K_1 = \begin{pmatrix} 0 & 1 & 1 & 2 & 2 & 1 & 1 & 1 & 2 \\ 1 & 0 & 1 & 2 & 1 & 1 & 2 & 1 & 1 \\ 1 & 1 & 0 & 2 & 2 & 1 & 1 & 2 & 2 \\ 2 & 2 & 2 & 0 & 1 & 2 & 1 & 2 & 2 \\ 2 & 1 & 2 & 1 & 0 & 2 & 2 & 1 & 1 \\ 1 & 1 & 1 & 2 & 2 & 0 & 1 & 1 & 2 \\ 1 & 2 & 1 & 1 & 2 & 1 & 0 & 2 & 2 \\ 1 & 1 & 2 & 2 & 1 & 1 & 2 & 0 & 2 \\ 2 & 1 & 2 & 2 & 1 & 2 & 2 & 2 & 0 \end{pmatrix} \tag{4.1}$$

Для данного класса данных приведена таблица 4.2 с выборкой параметров, которые наилучшим образом решают поставленную задачу, полные результаты параметризации приведены в приложении А. Использованы следующие обозначения: Days — количество дней, Result — результат работы, Mistake — ошибка как отклонение решения от эталонного .

Таблица 4.2 – Параметры для класса данных 1

α	ρ	Days	Result	Mistake
0.1	0.3	10	9	0
0.1	0.3	50	9	0
0.1	0.3	100	9	0
0.1	0.3	300	9	0
0.1	0.3	500	9	0
0.1	0.4	10	9	0
0.1	0.4	50	9	0
0.1	0.4	100	9	0
0.1	0.4	300	9	0
0.1	0.4	500	9	0
0.1	0.7	10	9	0
0.1	0.7	50	9	0
0.1	0.7	100	9	0
0.1	0.7	300	9	0
0.1	0.7	500	9	0
0.2	0.5	10	9	0

0.2	0.5	50	9	0
0.2	0.5	100	9	0
0.2	0.5	300	9	0
0.2	0.5	500	9	0
0.2	0.7	10	9	0
0.2	0.7	50	9	0
0.2	0.7	100	9	0
0.2	0.7	300	9	0
0.2	0.7	500	9	0
0.3	0.4	10	9	0
0.3	0.4	50	9	0
0.3	0.4	100	9	0
0.3	0.4	300	9	0
0.3	0.4	500	9	0
0.3	0.5	10	9	0
0.3	0.5	100	9	0
0.3	0.5	300	9	0
0.3	0.5	500	9	0
0.4	0.5	10	9	0
0.4	0.5	50	9	0
0.4	0.5	100	9	0
0.4	0.5	300	9	0
0.4	0.5	500	9	0
0.6	0.1	10	9	0
0.6	0.1	50	9	0
0.6	0.1	100	9	0
0.6	0.1	300	9	0
0.6	0.1	500	9	0

4.4.2 Класс данных 2

Класс данных 2 представляет собой матрицу смежности размером 9 элементов (большой разброс значений - от 1000 до 9999), которая представлена далее.

$$K_1 = \begin{pmatrix} 0 & 9271 & 8511 & 2010 & 1983 & 7296 & 7289 & 3024 & 1011 \\ 9271 & 0 & 7731 & 4865 & 5494 & 6812 & 4755 & 7780 & 7641 \\ 8511 & 7731 & 0 & 1515 & 9297 & 7506 & 5781 & 5804 & 7334 \\ 2010 & 4865 & 1515 & 0 & 3662 & 9597 & 2876 & 8188 & 9227 \\ 1983 & 5494 & 9297 & 3662 & 0 & 8700 & 4754 & 7445 & 3834 \\ 7296 & 6812 & 7506 & 9597 & 8700 & 0 & 4216 & 5553 & 8215 \\ 7289 & 4755 & 5781 & 2876 & 4754 & 4216 & 0 & 4001 & 4715 \\ 3024 & 7780 & 5804 & 8188 & 7445 & 5553 & 4001 & 0 & 9522 \\ 1011 & 7641 & 7334 & 9227 & 3834 & 8215 & 4715 & 9522 & 0 \end{pmatrix} \quad (4.2)$$

Для данного класса данных приведена таблица с выборкой параметров, которые наилучшим образом решают поставленную задачу Days — количество дней, Result — результат работы, Mistake — ошибочность полученного результата).

Таблица 4.3 – Параметры для класса данных 2

α	ρ	Days	Result	Mistake
0.1	0.3	100	34192	0
0.1	0.3	300	34192	0
0.1	0.3	500	34192	0
0.1	0.7	100	34192	0
0.1	0.7	300	34192	0
0.1	0.7	500	34192	0
0.2	0.1	100	34192	0
0.2	0.1	300	34192	0
0.2	0.1	500	34192	0
0.2	0.2	100	34192	0

0.2	0.2	300	34192	0
0.2	0.2	500	34192	0
0.2	0.5	100	34192	0
0.2	0.5	300	34192	0
0.2	0.5	500	34192	0
0.2	0.8	100	34192	0
0.2	0.8	300	34192	0
0.2	0.8	500	34192	0
0.3	0.1	100	34192	0
0.3	0.1	300	34192	0
0.3	0.1	500	34192	0
0.3	0.2	5	34192	0
0.3	0.2	50	34192	0
0.3	0.2	100	34192	0
0.3	0.2	300	34192	0
0.3	0.2	500	34192	0
0.4	0.5	50	34192	0
0.4	0.5	300	34192	0
0.4	0.5	500	34192	0
0.5	0.2	100	34192	0
0.5	0.2	300	34192	0
0.5	0.2	500	34192	0
0.6	0.2	100	34192	0
0.6	0.2	300	34192	0
0.6	0.2	500	34192	0
0.6	0.3	300	34192	0
0.6	0.3	500	34192	0
0.6	0.4	100	34192	0
0.6	0.4	500	34192	0
0.6	0.5	100	34192	0
0.6	0.5	300	34192	0
0.6	0.5	500	34192	0

Вывод

В результате эксперимента было получно, что использование муравьиного алгоритма наиболее эффективно при больших размерах матриц. Так, при размере матрицы, равном 2, муравьиный алгоритм медленее алгоритма полного перебора в 153 раза, а при размере матрицы, равном 9, муравьиный алгоритм быстрее алгоритма полного перебора в раз, а при размере в 10 – уже в 21 раз. Следовательно, при рамзерах матриц больше 8 следует использовать муравьиный алгоритм, но стоит учитывать, что он не гарантирует получения глобального оптимума при решении задачи.

Также при проведении эксперимента с классами данных было получено, что на первом классе данных (см. п. 4.4.1) муравьиный алгоритм лучше всего показывает себя при параметрах:

- $\alpha = 0.1, \rho = 0.3, 0.4, 0.7$;
- $\alpha = 0.2, \rho = 0.5, 0.7$;
- $\alpha = 0.3, \rho = 0.4, 0.5$;
- $\alpha = 0.4, \rho = 0.5$;
- $\alpha = 0.6, \rho = 0.1$.

Следовательно, для класса данных 1 рекомендуется использовать данные параметры.

Для класса данных 2 (см. п. 4.4.2) было получено, что наилучшим образом алгоритм работает на значениях параметров, которые представлены далее:

- $\alpha = 0.1, \rho = 0.3, 0.7$;
- $\alpha = 0.2, \rho = 0.1, 0.2, 0.5, 0.8$;
- $\alpha = 0.3, \rho = 0.1, 0.2$;
- $\alpha = 0.4, \rho = 0.5$;
- $\alpha = 0.5, \rho = 0.2$;

— $\alpha = 0.6, \rho = 0.2, 0.3, 0.4$.

Для второго класса данных 2 рекомендуется использовать данные параметры.

Также во время исллледования было замечено — чем меньше α, тем меньше погрешностей возникает. При этом число дней жизни колонии значительно влияет на качество решения: чем значение параметра $Days$ больше, тем меньше отклонение решения от эталонного.

Заключение

В ходе выполнения лабораторной работы были решены следующие задачи:

– описана задача коммивояжера;

– описаны методы решения задачи коммивояжера;

– реализованы выбранные алгоритмы;

– проведено функциональное тестирование разработанных алгоритмов;

– исследовано время работы реализаций алгоритмов.

В данном были сравнены алгоритмы по времени. Выявлено, что на больших матрицах муравьиный алгоритм показывает большую эффективность. Однако следует учесть, что, в отличие от алгоритма полного перебора, муравьиный алгоритм не гарантирует оптимального решения.

Поставленная цель достигнута: описан метод решения задачи коммивояжера на основе муравьиного алгоритма.

Список использованных источников

1. О. Борознов В. Исследование решения задачи коммивояжера. — АГТУ, Вестник Астраханского государственного технического университета. Серия: Управление, вычислительная техника и информатика [Электронный ресурс], 2022. URL: Режим доступа: https://cyberleninka.ru/article/n/issledovanie-resheniya-zadachi-kommivoyazhera/viewer (дата обращения: 18.11.2021).

2. Семёнов С. С. Педан А. В. Воловиков В. С. Климов И. С. Анализ трудоёмкости различных алгоритмических подходов для решения задачи коммивояжёра. — ООО «Корпорация «Интел Групп», Системы управления, связи и безопасности [Электронный ресурс], 2022. URL: Режим доступа: https://cyberleninka.ru/article/n/analiz-trudoemkosti-razlichnyh-algoritmicheskih-podhodov-dlya-\resheniya-zadachi-kommivoyazhera (дата обращения: 18.11.2021).

3. Welcome to Python [Электронный ресурс]. Режим доступа: https://www.python.org (дата обращения: 04.09.2021).

4. time — Time access and conversions [Электронный ресурс]. Режим доступа: https://docs.python.org/3/library/time.html#functions (дата обращения: 04.09.2021).

5. Manjaro [Электронный ресурс]. Режим доступа: https://manjaro.org/ (дата обращения: 03.10.2022).

6. Linux – Википедия [Электронный ресурс]. Режим доступа: https://ru.wikipedia.org/wiki/Linux (дата обращения: 04.10.2022).

7. Мобильный процессор AMD Ryzen™ 7 3700U [Электронный ресурс]. Режим доступа: https://www.amd.com/ru/products/apu/amd-ryzen-7-3700u (дата обращения: 04.10.2022).

Приложение А

Таблица 4.4 – Параметризация для класса данных 1, Days — количество дней, Result — результат работы, Mistake — ошибочность полученного результата

α	ρ	Days	Result	Mistake
0.1	0.1	3	9	1
0.1	0.1	5	9	0
0.1	0.1	10	9	1
0.1	0.1	50	9	0
0.1	0.1	100	9	0
0.1	0.1	300	9	0
0.1	0.1	500	9	0
0.1	0.2	1	9	2
0.1	0.2	3	9	0
0.1	0.2	5	9	1
0.1	0.2	10	9	1
0.1	0.2	50	9	0
0.1	0.2	100	9	0
0.1	0.2	300	9	0
0.1	0.2	500	9	0
0.1	0.3	1	9	2
0.1	0.3	3	9	0
0.1	0.3	5	9	1
0.1	0.3	10	9	2
0.1	0.3	50	9	1
0.1	0.3	100	9	1
0.1	0.3	300	9	0
0.1	0.3	500	9	0
0.1	0.4	1	9	3
0.1	0.4	3	9	2
0.1	0.4	5	9	2
0.1	0.4	10	9	1

0.1	0.4	50	9	0
0.1	0.4	100	9	0
0.1	0.4	300	9	0
0.1	0.4	500	9	0
0.1	0.5	1	9	3
0.1	0.5	3	9	1
0.1	0.5	5	9	1
0.1	0.5	10	9	1
0.1	0.5	50	9	0
0.1	0.5	100	9	1
0.1	0.5	300	9	0
0.1	0.5	500	9	0
0.1	0.6	1	9	1
0.1	0.6	3	9	2
0.1	0.6	5	9	2
0.1	0.6	10	9	1
0.1	0.6	50	9	0
0.1	0.6	100	9	0
0.1	0.6	300	9	0
0.1	0.6	500	9	0
0.1	0.7	1	9	2
0.1	0.7	3	9	1
0.1	0.7	5	9	1
0.1	0.7	10	9	1
0.1	0.7	50	9	0
0.1	0.7	100	9	0
0.1	0.7	300	9	0
0.1	0.7	500	9	0
0.1	0.8	1	9	3
0.1	0.8	3	9	1
0.1	0.8	5	9	2
0.1	0.8	10	9	1
0.1	0.8	50	9	0

0.1	0.8	100	9	0
0.1	0.8	300	9	0
0.1	0.8	500	9	0
0.2	0.1	1	9	2
0.2	0.1	3	9	1
0.2	0.1	5	9	1
0.2	0.1	10	9	1
0.2	0.1	50	9	0
0.2	0.1	100	9	0
0.2	0.1	300	9	0
0.2	0.1	500	9	0
0.2	0.2	1	9	3
0.2	0.2	3	9	1
0.2	0.2	5	9	1
0.2	0.2	10	9	1
0.2	0.2	50	9	0
0.2	0.2	100	9	0
0.2	0.2	300	9	0
0.2	0.2	500	9	0
0.2	0.3	1	9	2
0.2	0.3	3	9	1
0.2	0.3	5	9	1
0.2	0.3	10	9	1
0.2	0.3	50	9	0
0.2	0.3	100	9	0
0.2	0.3	300	9	0
0.2	0.3	500	9	0
0.2	0.4	1	9	2
0.2	0.4	3	9	1
0.2	0.4	5	9	1
0.2	0.4	10	9	1
0.2	0.4	50	9	0
0.2	0.4	100	9	0

0.2	0.4	300	9	0
0.2	0.4	500	9	0
0.2	0.5	1	9	2
0.2	0.5	3	9	1
0.2	0.5	5	9	1
0.2	0.5	10	9	1
0.2	0.5	50	9	1
0.2	0.5	100	9	0
0.2	0.5	300	9	0
0.2	0.5	500	9	0
0.2	0.6	1	9	3
0.2	0.6	3	9	1
0.2	0.6	5	9	1
0.2	0.6	10	9	1
0.2	0.6	50	9	0
0.2	0.6	100	9	0
0.2	0.6	300	9	0
0.2	0.6	500	9	0
0.2	0.7	1	9	2
0.2	0.7	3	9	1
0.2	0.7	5	9	1
0.2	0.7	10	9	0
0.2	0.7	50	9	0
0.2	0.7	100	9	0
0.2	0.7	300	9	0
0.2	0.7	500	9	0
0.2	0.8	1	9	2
0.2	0.8	3	9	1
0.2	0.8	5	9	2
0.2	0.8	10	9	1
0.2	0.8	50	9	0
0.2	0.8	100	9	0
0.2	0.8	300	9	0

0.2	0.8	500	9	0
0.3	0.1	1	9	2
0.3	0.1	3	9	2
0.3	0.1	5	9	1
0.3	0.1	10	9	1
0.3	0.1	50	9	1
0.3	0.1	100	9	0
0.3	0.1	300	9	0
0.3	0.1	500	9	0
0.3	0.2	1	9	2
0.3	0.2	3	9	1
0.3	0.2	5	9	1
0.3	0.2	10	9	1
0.3	0.2	50	9	1
0.3	0.2	100	9	0
0.3	0.2	300	9	0
0.3	0.2	500	9	0
0.3	0.3	1	9	3
0.3	0.3	3	9	1
0.3	0.3	5	9	2
0.3	0.3	10	9	1
0.3	0.3	50	9	1
0.3	0.3	100	9	0
0.3	0.3	300	9	0
0.3	0.3	500	9	0
0.3	0.4	1	9	2
0.3	0.4	3	9	1
0.3	0.4	5	9	1
0.3	0.4	10	9	1
0.3	0.4	50	9	1
0.3	0.4	100	9	1
0.3	0.4	300	9	0
0.3	0.4	500	9	0

0.3	0.5	1	9	3
0.3	0.5	3	9	1
0.3	0.5	5	9	2
0.3	0.5	10	9	1
0.3	0.5	50	9	1
0.3	0.5	100	9	0
0.3	0.5	300	9	0
0.3	0.5	500	9	0
0.3	0.6	1	9	2
0.3	0.6	3	9	1
0.3	0.6	5	9	1
0.3	0.6	10	9	1
0.3	0.6	50	9	0
0.3	0.6	100	9	0
0.3	0.6	300	9	0
0.3	0.6	500	9	0
0.3	0.7	1	9	4
0.3	0.7	3	9	2
0.3	0.7	5	9	1
0.3	0.7	10	9	1
0.3	0.7	50	9	0
0.3	0.7	100	9	1
0.3	0.7	300	9	0
0.3	0.7	500	9	0
0.3	0.8	1	9	4
0.3	0.8	3	9	2
0.3	0.8	5	9	1
0.3	0.8	10	9	1
0.3	0.8	50	9	0
0.3	0.8	100	9	0
0.3	0.8	300	9	0
0.3	0.8	500	9	0
0.4	0.1	1	9	1

0.4	0.1	3	9	2
0.4	0.1	5	9	2
0.4	0.1	10	9	1
0.4	0.1	50	9	0
0.4	0.1	100	9	0
0.4	0.1	300	9	0
0.4	0.1	500	9	0
0.4	0.2	1	9	0
0.4	0.2	3	9	2
0.4	0.2	5	9	1
0.4	0.2	10	9	0
0.4	0.2	50	9	1
0.4	0.2	100	9	1
0.4	0.2	300	9	0
0.4	0.2	500	9	0
0.4	0.3	1	9	1
0.4	0.3	3	9	2
0.4	0.3	5	9	0
0.4	0.3	10	9	2
0.4	0.3	50	9	0
0.4	0.3	100	9	0
0.4	0.3	300	9	0
0.4	0.3	500	9	0
0.4	0.4	1	9	2
0.4	0.4	3	9	2
0.4	0.4	5	9	2
0.4	0.4	10	9	1
0.4	0.4	50	9	1
0.4	0.4	100	9	0
0.4	0.4	300	9	0
0.4	0.4	500	9	0
0.4	0.5	1	9	2
0.4	0.5	3	9	1

0.4	0.5	5	9	1
0.4	0.5	10	9	0
0.4	0.5	50	9	1
0.4	0.5	100	9	0
0.4	0.5	300	9	0
0.4	0.5	500	9	0
0.4	0.6	1	9	2
0.4	0.6	3	9	1
0.4	0.6	5	9	1
0.4	0.6	10	9	1
0.4	0.6	50	9	1
0.4	0.6	100	9	0
0.4	0.6	300	9	0
0.4	0.6	500	9	0
0.4	0.7	1	9	1
0.4	0.7	3	9	2
0.4	0.7	5	9	1
0.4	0.7	10	9	1
0.4	0.7	50	9	1
0.4	0.7	100	9	0
0.4	0.7	300	9	0
0.4	0.7	500	9	0
0.4	0.8	1	9	2
0.4	0.8	3	9	2
0.4	0.8	5	9	2
0.4	0.8	10	9	2
0.4	0.8	50	9	1
0.4	0.8	100	9	0
0.4	0.8	300	9	0
0.4	0.8	500	9	0
0.5	0.1	1	9	2
0.5	0.1	3	9	1
0.5	0.1	5	9	1

0.5	0.1	10	9	1
0.5	0.1	50	9	1
0.5	0.1	100	9	0
0.5	0.1	300	9	0
0.5	0.1	500	9	0
0.5	0.2	1	9	2
0.5	0.2	3	9	2
0.5	0.2	5	9	1
0.5	0.2	10	9	1
0.5	0.2	50	9	1
0.5	0.2	100	9	0
0.5	0.2	300	9	0
0.5	0.2	500	9	0
0.5	0.3	1	9	3
0.5	0.3	3	9	2
0.5	0.3	5	9	1
0.5	0.3	10	9	1
0.5	0.3	50	9	1
0.5	0.3	100	9	0
0.5	0.3	300	9	0
0.5	0.3	500	9	0
0.5	0.4	1	9	2
0.5	0.4	3	9	2
0.5	0.4	5	9	2
0.5	0.4	10	9	1
0.5	0.4	50	9	1
0.5	0.4	100	9	0
0.5	0.4	300	9	0
0.5	0.4	500	9	0
0.5	0.5	1	9	3
0.5	0.5	3	9	2
0.5	0.5	5	9	3
0.5	0.5	10	9	1

0.5	0.5	50	9	1
0.5	0.5	100	9	1
0.5	0.5	300	9	0
0.5	0.5	500	9	0
0.5	0.6	1	9	2
0.5	0.6	3	9	2
0.5	0.6	5	9	2
0.5	0.6	10	9	1
0.5	0.6	50	9	1
0.5	0.6	100	9	0
0.5	0.6	300	9	0
0.5	0.6	500	9	0
0.5	0.7	1	9	2
0.5	0.7	3	9	2
0.5	0.7	5	9	2
0.5	0.7	10	9	1
0.5	0.7	50	9	0
0.5	0.7	100	9	1
0.5	0.7	300	9	0
0.5	0.7	500	9	0
0.5	0.8	1	9	2
0.5	0.8	3	9	1
0.5	0.8	5	9	2
0.5	0.8	10	9	0
0.5	0.8	50	9	1
0.5	0.8	100	9	1
0.5	0.8	300	9	0
0.5	0.8	500	9	0
0.6	0.1	1	9	2
0.6	0.1	3	9	1
0.6	0.1	5	9	2
0.6	0.1	10	9	1
0.6	0.1	50	9	1

0.6	0.1	100	9	1
0.6	0.1	300	9	0
0.6	0.1	500	9	0
0.6	0.2	1	9	3
0.6	0.2	3	9	2
0.6	0.2	5	9	1
0.6	0.2	10	9	1
0.6	0.2	50	9	1
0.6	0.2	100	9	0
0.6	0.2	300	9	0
0.6	0.2	500	9	0
0.6	0.3	1	9	2
0.6	0.3	3	9	2
0.6	0.3	5	9	1
0.6	0.3	10	9	2
0.6	0.3	50	9	0
0.6	0.3	100	9	0
0.6	0.3	300	9	0
0.6	0.3	500	9	0
0.6	0.4	1	9	3
0.6	0.4	3	9	1
0.6	0.4	5	9	1
0.6	0.4	10	9	1
0.6	0.4	50	9	1
0.6	0.4	100	9	0
0.6	0.4	300	9	0
0.6	0.4	500	9	0
0.6	0.5	1	9	2
0.6	0.5	3	9	2
0.6	0.5	5	9	2
0.6	0.5	10	9	1
0.6	0.5	50	9	1
0.6	0.5	100	9	0

0.6	0.5	300	9	0
0.6	0.5	500	9	0
0.6	0.6	1	9	2
0.6	0.6	3	9	0
0.6	0.6	5	9	2
0.6	0.6	10	9	2
0.6	0.6	50	9	1
0.6	0.6	100	9	0
0.6	0.6	300	9	0
0.6	0.6	500	9	0
0.6	0.7	1	9	2
0.6	0.7	3	9	2
0.6	0.7	5	9	1
0.6	0.7	10	9	2
0.6	0.7	50	9	1
0.6	0.7	100	9	1
0.6	0.7	300	9	0
0.6	0.7	500	9	0
0.6	0.8	1	9	2
0.6	0.8	3	9	3
0.6	0.8	5	9	1
0.6	0.8	10	9	1
0.6	0.8	50	9	1
0.6	0.8	100	9	0
0.6	0.8	300	9	0
0.6	0.8	500	9	0
0.7	0.1	1	9	4
0.7	0.1	3	9	3
0.7	0.1	5	9	1
0.7	0.1	10	9	1
0.7	0.1	50	9	1
0.7	0.1	100	9	0
0.7	0.1	300	9	0

0.7	0.1	500	9	0
0.7	0.2	1	9	2
0.7	0.2	3	9	1
0.7	0.2	5	9	2
0.7	0.2	10	9	1
0.7	0.2	50	9	1
0.7	0.2	100	9	0
0.7	0.2	300	9	0
0.7	0.2	500	9	0
0.7	0.3	1	9	1
0.7	0.3	3	9	1
0.7	0.3	5	9	2
0.7	0.3	10	9	1
0.7	0.3	50	9	1
0.7	0.3	100	9	1
0.7	0.3	300	9	0
0.7	0.3	500	9	0
0.7	0.4	1	9	3
0.7	0.4	3	9	2
0.7	0.4	5	9	1
0.7	0.4	10	9	1
0.7	0.4	50	9	0
0.7	0.4	100	9	1
0.7	0.4	300	9	0
0.7	0.4	500	9	0
0.7	0.5	1	9	2
0.7	0.5	3	9	3
0.7	0.5	5	9	1
0.7	0.5	10	9	1
0.7	0.5	50	9	0
0.7	0.5	100	9	1
0.7	0.5	300	9	1
0.7	0.5	500	9	0

0.7	0.6	1	9	4
0.7	0.6	3	9	3
0.7	0.6	5	9	1
0.7	0.6	10	9	0
0.7	0.6	50	9	0
0.7	0.6	100	9	0
0.7	0.6	300	9	0
0.7	0.6	500	9	0
0.7	0.7	1	9	2
0.7	0.7	3	9	2
0.7	0.7	5	9	1
0.7	0.7	10	9	0
0.7	0.7	50	9	0
0.7	0.7	100	9	0
0.7	0.7	300	9	0
0.7	0.7	500	9	0
0.7	0.8	1	9	2
0.7	0.8	3	9	2
0.7	0.8	5	9	2
0.7	0.8	10	9	1
0.7	0.8	50	9	1
0.7	0.8	100	9	1
0.7	0.8	300	9	0
0.7	0.8	500	9	0
0.8	0.1	1	9	3
0.8	0.1	3	9	2
0.8	0.1	5	9	1
0.8	0.1	10	9	2
0.8	0.1	50	9	0
0.8	0.1	100	9	1
0.8	0.1	300	9	0
0.8	0.1	500	9	0
0.8	0.2	1	9	2

0.8	0.2	3	9	2
0.8	0.2	5	9	2
0.8	0.2	10	9	1
0.8	0.2	50	9	1
0.8	0.2	100	9	1
0.8	0.2	300	9	0
0.8	0.2	500	9	0
0.8	0.3	1	9	3
0.8	0.3	3	9	3
0.8	0.3	5	9	1
0.8	0.3	10	9	1
0.8	0.3	50	9	0
0.8	0.3	100	9	1
0.8	0.3	300	9	0
0.8	0.3	500	9	0
0.8	0.4	1	9	2
0.8	0.4	3	9	2
0.8	0.4	5	9	2
0.8	0.4	10	9	2
0.8	0.4	50	9	0
0.8	0.4	100	9	0
0.8	0.4	300	9	1
0.8	0.4	500	9	0
0.8	0.5	1	9	2
0.8	0.5	3	9	2
0.8	0.5	5	9	1
0.8	0.5	10	9	1
0.8	0.5	50	9	1
0.8	0.5	100	9	1
0.8	0.5	300	9	0
0.8	0.5	500	9	0
0.8	0.6	1	9	3
0.8	0.6	3	9	2

0.8	0.6	5	9	2
0.8	0.6	10	9	1
0.8	0.6	50	9	1
0.8	0.6	100	9	1
0.8	0.6	300	9	0
0.8	0.6	500	9	0
0.8	0.7	1	9	3
0.8	0.7	3	9	1
0.8	0.7	5	9	0
0.8	0.7	10	9	1
0.8	0.7	50	9	1
0.8	0.7	100	9	1
0.8	0.7	300	9	0
0.8	0.7	500	9	0
0.8	0.8	1	9	3
0.8	0.8	3	9	3
0.8	0.8	5	9	2
0.8	0.8	10	9	2
0.8	0.8	50	9	0
0.8	0.8	100	9	0
0.8	0.8	300	9	0
0.8	0.8	500	9	0
0.9	0.1	1	9	3
0.9	0.1	3	9	2
0.9	0.1	5	9	0
0.9	0.1	10	9	2
0.9	0.1	50	9	0
0.9	0.1	100	9	1
0.9	0.1	300	9	0
0.9	0.1	500	9	0
0.9	0.2	1	9	3
0.9	0.2	3	9	2
0.9	0.2	5	9	2

0.9	0.2	10	9	1
0.9	0.2	50	9	1
0.9	0.2	100	9	0
0.9	0.2	300	9	0
0.9	0.2	500	9	0
0.9	0.3	1	9	3
0.9	0.3	3	9	1
0.9	0.3	5	9	2
0.9	0.3	10	9	2
0.9	0.3	50	9	1
0.9	0.3	100	9	1
0.9	0.3	300	9	0
0.9	0.3	500	9	0
0.9	0.4	1	9	3
0.9	0.4	3	9	1
0.9	0.4	5	9	2
0.9	0.4	10	9	1
0.9	0.4	50	9	1
0.9	0.4	100	9	1
0.9	0.4	300	9	0
0.9	0.4	500	9	0
0.9	0.5	1	9	2
0.9	0.5	3	9	2
0.9	0.5	5	9	2
0.9	0.5	10	9	2
0.9	0.5	50	9	1
0.9	0.5	100	9	1
0.9	0.5	300	9	0
0.9	0.5	500	9	0
0.9	0.6	1	9	3
0.9	0.6	3	9	2
0.9	0.6	5	9	2
0.9	0.6	10	9	0

0.9	0.6	50	9	1
0.9	0.6	100	9	0
0.9	0.6	300	9	1
0.9	0.6	500	9	0
0.9	0.7	1	9	3
0.9	0.7	3	9	2
0.9	0.7	5	9	2
0.9	0.7	10	9	1
0.9	0.7	50	9	1
0.9	0.7	100	9	0
0.9	0.7	300	9	0
0.9	0.7	500	9	0
0.9	0.8	1	9	3
0.9	0.8	3	9	2
0.9	0.8	5	9	1
0.9	0.8	10	9	1
0.9	0.8	50	9	1
0.9	0.8	100	9	1
0.9	0.8	300	9	0
0.9	0.8	500	9	0

Приложение Б

Таблица 4.5 – Параметризация для класса данных 2, Days — количество дней, Result — результат работы, Mistake — ошибочность полученного результата

α	ρ	Days	Result	Mistake
0.1	0.1	3	34192	2576
0.1	0.1	5	34192	1618
0.1	0.1	10	34192	3029
0.1	0.1	50	34192	0
0.1	0.1	100	34192	0
0.1	0.1	300	34192	0
0.1	0.1	500	34192	0
0.1	0.2	1	34192	3767
0.1	0.2	3	34192	4709
0.1	0.2	5	34192	3476
0.1	0.2	10	34192	2918
0.1	0.2	50	34192	394
0.1	0.2	100	34192	0
0.1	0.2	300	34192	0
0.1	0.2	500	34192	0
0.1	0.3	1	34192	1376
0.1	0.3	3	34192	2453
0.1	0.3	5	34192	3043
0.1	0.3	10	34192	3532
0.1	0.3	50	34192	1062
0.1	0.3	100	34192	0
0.1	0.3	300	34192	0
0.1	0.3	500	34192	0
0.1	0.4	1	34192	5166
0.1	0.4	3	34192	3014
0.1	0.4	5	34192	4212
0.1	0.4	10	34192	4226

0.1	0.4	50	34192	1614
0.1	0.4	100	34192	394
0.1	0.4	300	34192	505
0.1	0.4	500	34192	0
0.1	0.5	1	34192	8179
0.1	0.5	3	34192	6474
0.1	0.5	5	34192	3878
0.1	0.5	10	34192	2791
0.1	0.5	50	34192	505
0.1	0.5	100	34192	505
0.1	0.5	300	34192	0
0.1	0.5	500	34192	0
0.1	0.6	1	34192	8788
0.1	0.6	3	34192	3890
0.1	0.6	5	34192	1376
0.1	0.6	10	34192	964
0.1	0.6	50	34192	1755
0.1	0.6	100	34192	0
0.1	0.6	300	34192	0
0.1	0.6	500	34192	0
0.1	0.7	1	34192	7276
0.1	0.7	3	34192	2957
0.1	0.7	5	34192	1571
0.1	0.7	10	34192	1948
0.1	0.7	50	34192	1101
0.1	0.7	100	34192	394
0.1	0.7	300	34192	0
0.1	0.7	500	34192	0
0.1	0.8	1	34192	10326
0.1	0.8	3	34192	5156
0.1	0.8	5	34192	3029
0.1	0.8	10	34192	505
0.1	0.8	50	34192	394

0.1	0.8	100	34192	394
0.1	0.8	300	34192	0
0.1	0.8	500	34192	0
0.2	0.1	1	34192	6093
0.2	0.1	3	34192	1618
0.2	0.1	5	34192	4950
0.2	0.1	10	34192	3476
0.2	0.1	50	34192	505
0.2	0.1	100	34192	964
0.2	0.1	300	34192	394
0.2	0.1	500	34192	0
0.2	0.2	1	34192	8604
0.2	0.2	3	34192	3844
0.2	0.2	5	34192	3532
0.2	0.2	10	34192	1109
0.2	0.2	50	34192	0
0.2	0.2	100	34192	0
0.2	0.2	300	34192	0
0.2	0.2	500	34192	394
0.2	0.3	1	34192	3878
0.2	0.3	3	34192	7319
0.2	0.3	5	34192	4626
0.2	0.3	10	34192	5000
0.2	0.3	50	34192	1101
0.2	0.3	100	34192	505
0.2	0.3	300	34192	0
0.2	0.3	500	34192	0
0.2	0.4	1	34192	7604
0.2	0.4	3	34192	5525
0.2	0.4	5	34192	1643
0.2	0.4	10	34192	1881
0.2	0.4	50	34192	1101
0.2	0.4	100	34192	1062

0.2	0.4	300	34192	0
0.2	0.4	500	34192	0
0.2	0.5	1	34192	8018
0.2	0.5	3	34192	3435
0.2	0.5	5	34192	1614
0.2	0.5	10	34192	1062
0.2	0.5	50	34192	505
0.2	0.5	100	34192	394
0.2	0.5	300	34192	505
0.2	0.5	500	34192	0
0.2	0.6	1	34192	4181
0.2	0.6	3	34192	1274
0.2	0.6	5	34192	6212
0.2	0.6	10	34192	1571
0.2	0.6	50	34192	1062
0.2	0.6	100	34192	505
0.2	0.6	300	34192	0
0.2	0.6	500	34192	0
0.2	0.7	1	34192	5092
0.2	0.7	3	34192	6232
0.2	0.7	5	34192	3757
0.2	0.7	10	34192	2789
0.2	0.7	50	34192	0
0.2	0.7	100	34192	394
0.2	0.7	300	34192	0
0.2	0.7	500	34192	0
0.2	0.8	1	34192	8587
0.2	0.8	3	34192	5041
0.2	0.8	5	34192	3150
0.2	0.8	10	34192	0
0.2	0.8	50	34192	0
0.2	0.8	100	34192	0
0.2	0.8	300	34192	0

0.2	0.8	500	34192	0
0.3	0.1	1	34192	6858
0.3	0.1	3	34192	5996
0.3	0.1	5	34192	1819
0.3	0.1	10	34192	3002
0.3	0.1	50	34192	1062
0.3	0.1	100	34192	0
0.3	0.1	300	34192	0
0.3	0.1	500	34192	0
0.3	0.2	1	34192	5756
0.3	0.2	3	34192	1101
0.3	0.2	5	34192	4245
0.3	0.2	10	34192	3806
0.3	0.2	50	34192	3504
0.3	0.2	100	34192	0
0.3	0.2	300	34192	0
0.3	0.2	500	34192	0
0.3	0.3	1	34192	5214
0.3	0.3	3	34192	5000
0.3	0.3	5	34192	0
0.3	0.3	10	34192	0
0.3	0.3	50	34192	1376
0.3	0.3	100	34192	0
0.3	0.3	300	34192	394
0.3	0.3	500	34192	505
0.3	0.4	1	34192	6916
0.3	0.4	3	34192	5525
0.3	0.4	5	34192	1101
0.3	0.4	10	34192	505
0.3	0.4	50	34192	505
0.3	0.4	100	34192	1274
0.3	0.4	300	34192	394
0.3	0.4	500	34192	0

0.3	0.5	1	34192	1881
0.3	0.5	3	34192	4234
0.3	0.5	5	34192	1618
0.3	0.5	10	34192	3566
0.3	0.5	50	34192	505
0.3	0.5	100	34192	1101
0.3	0.5	300	34192	0
0.3	0.5	500	34192	394
0.3	0.6	1	34192	10168
0.3	0.6	3	34192	4226
0.3	0.6	5	34192	2957
0.3	0.6	10	34192	964
0.3	0.6	50	34192	1755
0.3	0.6	100	34192	0
0.3	0.6	300	34192	0
0.3	0.6	500	34192	0
0.3	0.7	1	34192	3111
0.3	0.7	3	34192	3570
0.3	0.7	5	34192	1819
0.3	0.7	10	34192	4313
0.3	0.7	50	34192	1109
0.3	0.7	100	34192	394
0.3	0.7	300	34192	964
0.3	0.7	500	34192	394
0.3	0.8	1	34192	4288
0.3	0.8	3	34192	0
0.3	0.8	5	34192	5613
0.3	0.8	10	34192	4181
0.3	0.8	50	34192	1062
0.3	0.8	100	34192	394
0.3	0.8	300	34192	0
0.3	0.8	500	34192	0
0.4	0.1	1	34192	1819

0.4	0.1	3	34192	1376
0.4	0.1	5	34192	4263
0.4	0.1	10	34192	3494
0.4	0.1	50	34192	394
0.4	0.1	100	34192	394
0.4	0.1	300	34192	0
0.4	0.1	500	34192	0
0.4	0.2	1	34192	16069
0.4	0.2	3	34192	8584
0.4	0.2	5	34192	5389
0.4	0.2	10	34192	3558
0.4	0.2	50	34192	1109
0.4	0.2	100	34192	0
0.4	0.2	300	34192	0
0.4	0.2	500	34192	0
0.4	0.3	1	34192	6849
0.4	0.3	3	34192	5158
0.4	0.3	5	34192	3900
0.4	0.3	10	34192	3878
0.4	0.3	50	34192	1062
0.4	0.3	100	34192	505
0.4	0.3	300	34192	0
0.4	0.3	500	34192	0
0.4	0.4	1	34192	4498
0.4	0.4	3	34192	505
0.4	0.4	5	34192	4234
0.4	0.4	10	34192	1928
0.4	0.4	50	34192	1928
0.4	0.4	100	34192	394
0.4	0.4	300	34192	0
0.4	0.4	500	34192	0
0.4	0.5	1	34192	5323
0.4	0.5	3	34192	1376

0.4	0.5	5	34192	4507
0.4	0.5	10	34192	3002
0.4	0.5	50	34192	1571
0.4	0.5	100	34192	1487
0.4	0.5	300	34192	394
0.4	0.5	500	34192	0
0.4	0.6	1	34192	4187
0.4	0.6	3	34192	5019
0.4	0.6	5	34192	3851
0.4	0.6	10	34192	3878
0.4	0.6	50	34192	1643
0.4	0.6	100	34192	964
0.4	0.6	300	34192	0
0.4	0.6	500	34192	394
0.4	0.7	1	34192	1487
0.4	0.7	3	34192	3544
0.4	0.7	5	34192	3962
0.4	0.7	10	34192	1109
0.4	0.7	50	34192	1062
0.4	0.7	100	34192	0
0.4	0.7	300	34192	0
0.4	0.7	500	34192	0
0.4	0.8	1	34192	5996
0.4	0.8	3	34192	5323
0.4	0.8	5	34192	1881
0.4	0.8	10	34192	3504
0.4	0.8	50	34192	0
0.4	0.8	100	34192	1274
0.4	0.8	300	34192	394
0.4	0.8	500	34192	0
0.5	0.1	1	34192	9525
0.5	0.1	3	34192	3119
0.5	0.1	5	34192	1819

0.5	0.1	10	34192	4706
0.5	0.1	50	34192	964
0.5	0.1	100	34192	1487
0.5	0.1	300	34192	0
0.5	0.1	500	34192	0
0.5	0.2	1	34192	9710
0.5	0.2	3	34192	8086
0.5	0.2	5	34192	3878
0.5	0.2	10	34192	3043
0.5	0.2	50	34192	1109
0.5	0.2	100	34192	964
0.5	0.2	300	34192	0
0.5	0.2	500	34192	0
0.5	0.3	1	34192	5916
0.5	0.3	3	34192	5225
0.5	0.3	5	34192	964
0.5	0.3	10	34192	4245
0.5	0.3	50	34192	1101
0.5	0.3	100	34192	1928
0.5	0.3	300	34192	0
0.5	0.3	500	34192	0
0.5	0.4	1	34192	7563
0.5	0.4	3	34192	5634
0.5	0.4	5	34192	6743
0.5	0.4	10	34192	2061
0.5	0.4	50	34192	1643
0.5	0.4	100	34192	394
0.5	0.4	300	34192	394
0.5	0.4	500	34192	0
0.5	0.5	1	34192	2387
0.5	0.5	3	34192	4212
0.5	0.5	5	34192	3767
0.5	0.5	10	34192	2918

0.5	0.5	50	34192	1614
0.5	0.5	100	34192	1062
0.5	0.5	300	34192	0
0.5	0.5	500	34192	394
0.5	0.6	1	34192	9356
0.5	0.6	3	34192	5848
0.5	0.6	5	34192	1819
0.5	0.6	10	34192	2912
0.5	0.6	50	34192	1618
0.5	0.6	100	34192	2387
0.5	0.6	300	34192	394
0.5	0.6	500	34192	0
0.5	0.7	1	34192	6914
0.5	0.7	3	34192	6092
0.5	0.7	5	34192	4606
0.5	0.7	10	34192	3029
0.5	0.7	50	34192	1109
0.5	0.7	100	34192	0
0.5	0.7	300	34192	505
0.5	0.7	500	34192	0
0.5	0.8	1	34192	17787
0.5	0.8	3	34192	3306
0.5	0.8	5	34192	4749
0.5	0.8	10	34192	6089
0.5	0.8	50	34192	964
0.5	0.8	100	34192	505
0.5	0.8	300	34192	0
0.5	0.8	500	34192	505
0.6	0.1	1	34192	3504
0.6	0.1	3	34192	7003
0.6	0.1	5	34192	5156
0.6	0.1	10	34192	4220
0.6	0.1	50	34192	964

0.6	0.1	100	34192	0
0.6	0.1	300	34192	0
0.6	0.1	500	34192	964
0.6	0.2	1	34192	394
0.6	0.2	3	34192	1819
0.6	0.2	5	34192	2877
0.6	0.2	10	34192	5535
0.6	0.2	50	34192	1062
0.6	0.2	100	34192	0
0.6	0.2	300	34192	394
0.6	0.2	500	34192	394
0.6	0.3	1	34192	5646
0.6	0.3	3	34192	1856
0.6	0.3	5	34192	4485
0.6	0.3	10	34192	4731
0.6	0.3	50	34192	1274
0.6	0.3	100	34192	1274
0.6	0.3	300	34192	1062
0.6	0.3	500	34192	394
0.6	0.4	1	34192	13645
0.6	0.4	3	34192	3029
0.6	0.4	5	34192	5921
0.6	0.4	10	34192	2061
0.6	0.4	50	34192	3757
0.6	0.4	100	34192	505
0.6	0.4	300	34192	0
0.6	0.4	500	34192	0
0.6	0.5	1	34192	13753
0.6	0.5	3	34192	4139
0.6	0.5	5	34192	5799
0.6	0.5	10	34192	3705
0.6	0.5	50	34192	3504
0.6	0.5	100	34192	1643

0.6	0.5	300	34192	394
0.6	0.5	500	34192	394
0.6	0.6	1	34192	6633
0.6	0.6	3	34192	3014
0.6	0.6	5	34192	4383
0.6	0.6	10	34192	4495
0.6	0.6	50	34192	505
0.6	0.6	100	34192	1274
0.6	0.6	300	34192	394
0.6	0.6	500	34192	0
0.6	0.7	1	34192	9592
0.6	0.7	3	34192	5617
0.6	0.7	5	34192	1376
0.6	0.7	10	34192	2372
0.6	0.7	50	34192	1487
0.6	0.7	100	34192	394
0.6	0.7	300	34192	0
0.6	0.7	500	34192	505
0.6	0.8	1	34192	8427
0.6	0.8	3	34192	4438
0.6	0.8	5	34192	3111
0.6	0.8	10	34192	2877
0.6	0.8	50	34192	1274
0.6	0.8	100	34192	1376
0.6	0.8	300	34192	0
0.6	0.8	500	34192	1101
0.7	0.1	1	34192	11020
0.7	0.1	3	34192	5424
0.7	0.1	5	34192	1614
0.7	0.1	10	34192	3150
0.7	0.1	50	34192	2061
0.7	0.1	100	34192	1062
0.7	0.1	300	34192	0

0.7	0.1	500	34192	1062
0.7	0.2	1	34192	8093
0.7	0.2	3	34192	9703
0.7	0.2	5	34192	8043
0.7	0.2	10	34192	1948
0.7	0.2	50	34192	964
0.7	0.2	100	34192	1881
0.7	0.2	300	34192	0
0.7	0.2	500	34192	0
0.7	0.3	1	34192	4719
0.7	0.3	3	34192	7437
0.7	0.3	5	34192	4383
0.7	0.3	10	34192	4181
0.7	0.3	50	34192	2372
0.7	0.3	100	34192	0
0.7	0.3	300	34192	394
0.7	0.3	500	34192	505
0.7	0.4	1	34192	9401
0.7	0.4	3	34192	7573
0.7	0.4	5	34192	4181
0.7	0.4	10	34192	6092
0.7	0.4	50	34192	1948
0.7	0.4	100	34192	505
0.7	0.4	300	34192	0
0.7	0.4	500	34192	394
0.7	0.5	1	34192	11995
0.7	0.5	3	34192	4485
0.7	0.5	5	34192	6317
0.7	0.5	10	34192	2061
0.7	0.5	50	34192	1062
0.7	0.5	100	34192	1487
0.7	0.5	300	34192	1062
0.7	0.5	500	34192	394

0.7	0.6	1	34192	5123
0.7	0.6	3	34192	5365
0.7	0.6	5	34192	5838
0.7	0.6	10	34192	4181
0.7	0.6	50	34192	1881
0.7	0.6	100	34192	394
0.7	0.6	300	34192	0
0.7	0.6	500	34192	505
0.7	0.7	1	34192	9895
0.7	0.7	3	34192	7082
0.7	0.7	5	34192	2887
0.7	0.7	10	34192	5366
0.7	0.7	50	34192	2912
0.7	0.7	100	34192	1571
0.7	0.7	300	34192	0
0.7	0.7	500	34192	0
0.7	0.8	1	34192	6023
0.7	0.8	3	34192	5760
0.7	0.8	5	34192	6146
0.7	0.8	10	34192	3002
0.7	0.8	50	34192	2453
0.7	0.8	100	34192	0
0.7	0.8	300	34192	964
0.7	0.8	500	34192	1376
0.8	0.1	1	34192	13258
0.8	0.1	3	34192	4181
0.8	0.1	5	34192	5225
0.8	0.1	10	34192	5734
0.8	0.1	50	34192	964
0.8	0.1	100	34192	1487
0.8	0.1	300	34192	0
0.8	0.1	500	34192	394
0.8	0.2	1	34192	12519

0.8	0.2	3	34192	5348
0.8	0.2	5	34192	2877
0.8	0.2	10	34192	3566
0.8	0.2	50	34192	1101
0.8	0.2	100	34192	1571
0.8	0.2	300	34192	1109
0.8	0.2	500	34192	964
0.8	0.3	1	34192	13514
0.8	0.3	3	34192	3570
0.8	0.3	5	34192	5799
0.8	0.3	10	34192	3375
0.8	0.3	50	34192	1643
0.8	0.3	100	34192	1109
0.8	0.3	300	34192	964
0.8	0.3	500	34192	0
0.8	0.4	1	34192	7156
0.8	0.4	3	34192	5647
0.8	0.4	5	34192	5182
0.8	0.4	10	34192	5182
0.8	0.4	50	34192	964
0.8	0.4	100	34192	1274
0.8	0.4	300	34192	394
0.8	0.4	500	34192	394
0.8	0.5	1	34192	14770
0.8	0.5	3	34192	4485
0.8	0.5	5	34192	7495
0.8	0.5	10	34192	4526
0.8	0.5	50	34192	2791
0.8	0.5	100	34192	1618
0.8	0.5	300	34192	0
0.8	0.5	500	34192	0
0.8	0.6	1	34192	9025
0.8	0.6	3	34192	9574

0.8	0.6	5	34192	5119
0.8	0.6	10	34192	2387
0.8	0.6	50	34192	1062
0.8	0.6	100	34192	505
0.8	0.6	300	34192	0
0.8	0.6	500	34192	505
0.8	0.7	1	34192	7658
0.8	0.7	3	34192	964
0.8	0.7	5	34192	9732
0.8	0.7	10	34192	1819
0.8	0.7	50	34192	3111
0.8	0.7	100	34192	964
0.8	0.7	300	34192	1376
0.8	0.7	500	34192	964
0.8	0.8	1	34192	6427
0.8	0.8	3	34192	9428
0.8	0.8	5	34192	3757
0.8	0.8	10	34192	1487
0.8	0.8	50	34192	4187
0.8	0.8	100	34192	394
0.8	0.8	300	34192	394
0.8	0.8	500	34192	505
0.9	0.1	1	34192	9268
0.9	0.1	3	34192	4212
0.9	0.1	5	34192	5158
0.9	0.1	10	34192	5161
0.9	0.1	50	34192	394
0.9	0.1	100	34192	1618
0.9	0.1	300	34192	1376
0.9	0.1	500	34192	394
0.9	0.2	1	34192	11734
0.9	0.2	3	34192	9732
0.9	0.2	5	34192	8028

0.9	0.2	10	34192	3443
0.9	0.2	50	34192	1928
0.9	0.2	100	34192	505
0.9	0.2	300	34192	964
0.9	0.2	500	34192	1109
0.9	0.3	1	34192	8368
0.9	0.3	3	34192	4944
0.9	0.3	5	34192	10069
0.9	0.3	10	34192	6549
0.9	0.3	50	34192	3624
0.9	0.3	100	34192	1101
0.9	0.3	300	34192	1643
0.9	0.3	500	34192	964
0.9	0.4	1	34192	9443
0.9	0.4	3	34192	8292
0.9	0.4	5	34192	6086
0.9	0.4	10	34192	5871
0.9	0.4	50	34192	1755
0.9	0.4	100	34192	1643
0.9	0.4	300	34192	1487
0.9	0.4	500	34192	0
0.9	0.5	1	34192	13245
0.9	0.5	3	34192	5000
0.9	0.5	5	34192	7555
0.9	0.5	10	34192	4817
0.9	0.5	50	34192	2957
0.9	0.5	100	34192	3392
0.9	0.5	300	34192	1062
0.9	0.5	500	34192	964
0.9	0.6	1	34192	6916
0.9	0.6	3	34192	2791
0.9	0.6	5	34192	8220
0.9	0.6	10	34192	3890

0.9	0.6	50	34192	4739
0.9	0.6	100	34192	1819
0.9	0.6	300	34192	394
0.9	0.6	500	34192	1101
0.9	0.7	1	34192	12036
0.9	0.7	3	34192	8292
0.9	0.7	5	34192	6128
0.9	0.7	10	34192	5922
0.9	0.7	50	34192	394
0.9	0.7	100	34192	964
0.9	0.7	300	34192	1928
0.9	0.7	500	34192	0
0.9	0.8	1	34192	14992
0.9	0.8	3	34192	2957
0.9	0.8	5	34192	7008
0.9	0.8	10	34192	5182
0.9	0.8	50	34192	3494
0.9	0.8	100	34192	1062
0.9	0.8	300	34192	394
0.9	0.8	500	34192	0