Санкт-Петербургский государственный университет
Факультет прикладной математики и процессов управления

Иследование принципов работы метода отжига

Лабараторная работа №5

Аннотация

В данной работе представлено описание принципов работы метода отжтига, подкреплённое програмной и графической реализацией.

Ключевые слова: Метод отжига, Python, Tkinter

Автор работы: Шайдуров В.Д.
Группа: 21.Б15-пу
Научный руководитель: Дик А.Г.

Санкт-Петербург
2023 г.

Содержание

1 Вступление

В данной лабораторной работе мы будем изучать метод отжига, который является метаэвристическим алгоритмом для решения задач оптимизации. Метод отжига основан на имитации процесса охлаждения металла и применяется для решения задачи о коммивояжёре, т.е. нахождении минималного пути. Он способен находить оптимальное решение в условиях сильной локализации и существенных флуктуациях, которые могут снижать эффективность других методов оптимизации. Мы научимся реализовывать этот метод и проведем его тестирование на различных графах, чтобы оценить его эффективность и возможности применения.

2 Цель работы

Целью данной лабораторной работы является изучение основных принципов работы метода отжтга, а также его применение для задачи поиска минимального Гамильтонова пути.

3 Задача

Оценить эффективность метода отжига и реализовать его визуализацию.

4 Описание програмы

В данном разделе приводиться описание кода на уровне идеи. Для более подробного понимания, рекомендуется самостоятельно ознакомиться с кодом по ссылке (github). Каждое дейсвие в программе сопровожденно коментариями, если возникают вопросы по терменалогии или по общему устройству программы обращайтесь к этому разделу.

Програмная реализация написана на языке python 3.10, с использованием популярных и общеупотребимых пакетов.

4.1 Теоретические сведения

Необходимые термины:

Начальная температура в методе отжига - это значение, которое определяет начальную скорость охлаждения системы. Чем выше начальная температура, тем медленне будет происходить охлаждение, и тем больший шанс у алгоритма нахождения глобального оптимума.

Конечная температура в методе отжига - это значение, при достижении которого охлаждение системы прекращается. Это также является критерием остановки работы алгоритма, который гарантирует, что решение будет найдено. Как правило, конечная температура выбирается таким образом, чтобы достичь достаточно низкой температуры, которая гарантирует, что все возможные конфигурации будут рассмотрены, но не такой низкой, чтобы потребовалось слишком много времени для выполнения алгоритма.

Охлаждающий коэффициент - это значение, которое определяет скорость изменения температуры системы на каждом шаге алгоритма. Другими словами, скорость с которой начальная температура сходиться к конечной.

4.2 Общий ход программы

1. Вначале вызывается класс *class SimulatedAnn* в который передаётся, по усмотрению пользовался: **веса путей полного графа, начальная и конечная температуры, охлождающий коэффициент и количесво поколений**. Также в начале генерируется случайный путь, который является единственным и лучшим на данный момент.

2. Далее вызывается метод *findbestway* которой подолжает своё выполнение пока **начальная температура больше конечной**. Под **количеством поколений** понимается, количество перестоновок в родительском маршруте без изменения **начальной температуры**. Таким образом для последнего лучшего существующего маршрута производиться *numberGeneration* перестановок без изменения температуры.

3. Когда все попытки не влияющие на температуру исчерпаны, мы переопределяем лучший маршрут, и происходит охлаждение системы (начальная температура понижается).

4.3 Описание используемой схемы метода отжига

Существут множество схем метода отжига, по которым можно задавать правило охдождения системы, наиболие популярные способы вы можите прочитать в статье А.С.Лопатина, прикреплённой в списке литературы.

В моей программе был реализован метод отжга по схеме тушения, это когда с каждой итерацией система охлождается на охлождающий коэффициент лежаший в интервале от (0, 1).
Таким образом новая температура = старая температура * охлождающий коэффициент.

4.4 Описание програмной структуры генетического алгоритма

В Листиге 1 привёдн испольняемый код для класса SimulatedAnn.

```python
class SimulatedAnn:
    """ Класс реализующий метод отжига """
    def __init__(self, graphPaths, start_T, stop_T, coolingRatio, numberGeneration):
        self.graphPaths = graphPaths  # веса путей заданных в графе
        self.vertexes = len(graphPaths) + 1  # кол-во вершин
        self.coolingRatio = coolingRatio  # коэффициент охлаждения
        self.start_T = start_T  # начальная температура
        self.stop_T = stop_T  # конечная температура
        self.numberGeneration = numberGeneration  # количество попыток отыскать новый маршрут
        # задаём начальные значения
        self.best_path = [x for x in range(self.vertexes)]
        random.shuffle(self.best_path)
        self.best_score = self.count_length(copy.deepcopy(self.best_path))

    def count_length(self, path) -> int:
        """ Находим длину пути """
        path.append(path[0])
        score = 0
        # Потом добавляем остальные маршруты
        for i in range(self.vertexes):
            from_ = min(path[i], path[i + 1])
            to_ = max(path[i], path[i + 1]) - from_ - 1
            score += self.graphPaths[from_][to_]
        return score

    def find_best_way(self) -> None:
        """ Находим лучший путь с наименьшей длиной """
        while self.start_T > self.stop_T:
            # Даём сделать несколько попыток найти лучшую траекторию без понижения температуры
            previous_path = copy.deepcopy(self.best_path)
            for _ in range(self.numberGeneration):
                # пробуем найти лучший путь меняя 2 позиции
                new_path = copy.deepcopy(previous_path)
                pos1 = random.randrange(self.vertexes)
                pos2 = random.randrange(self.vertexes)
                while pos1 == pos2:
                    pos2 = random.randrange(self.vertexes)
                new_path[pos1], new_path[pos2] = new_path[pos2], new_path[pos1]
                # находим длину нового пути
                new_path_score = self.count_length(copy.deepcopy(new_path))
                # считаем вероятность перейти к следующему шагу
                h = math.exp(-(new_path_score - self.best_score) / self.start_T)
                if random.random() < h:
                    self.best_path = new_path
                    self.best_score = new_path_score
            # понижаем вероятность перейти к следующему шагу
            self.start_T = self.start_T * self.coolingRatio
        self.best_path.append(self.best_path[0])
```

Listing 1: class SimulatedAnn

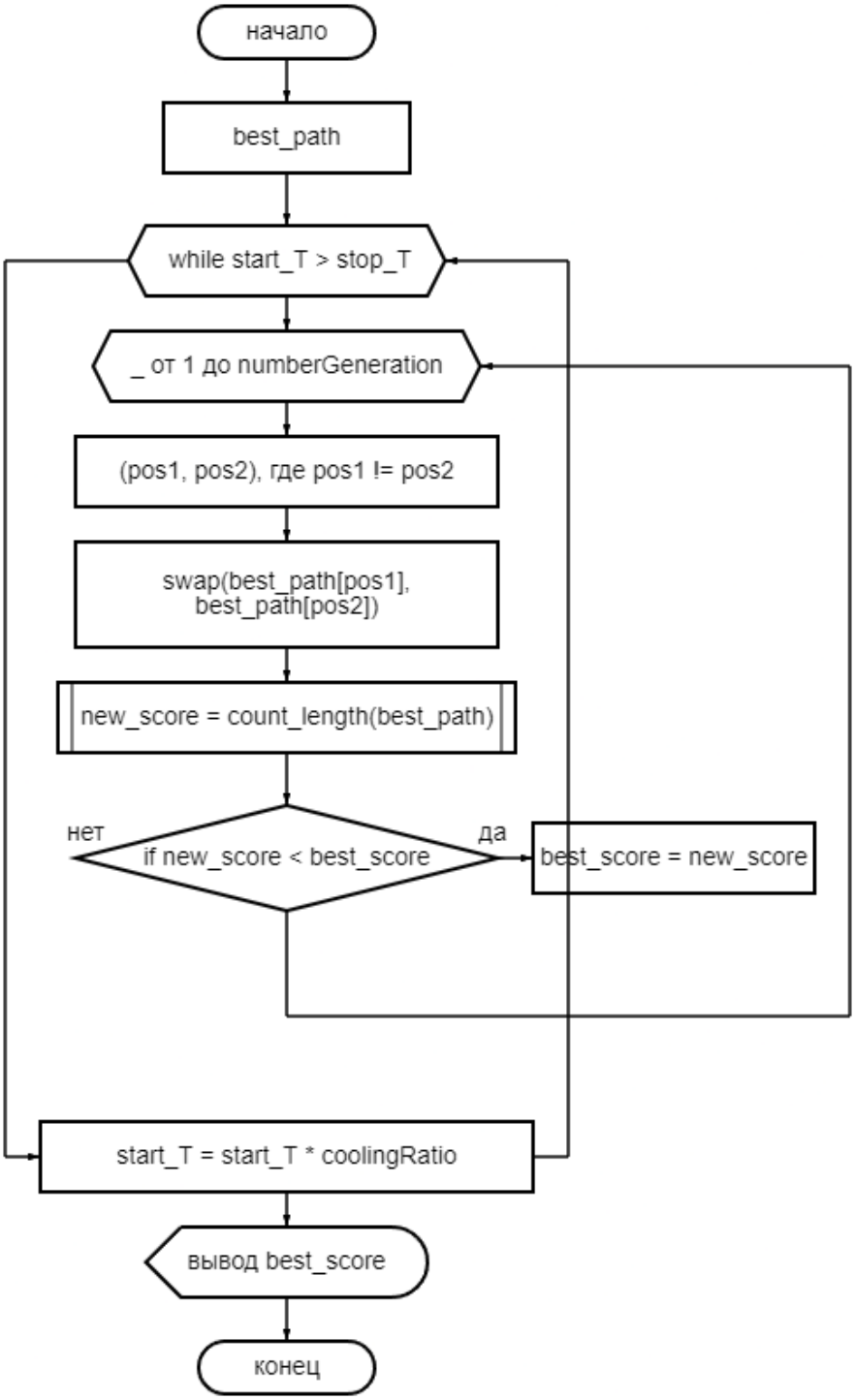

Рис. 1: Блоксхема программы.

5 Визуализация работы программы

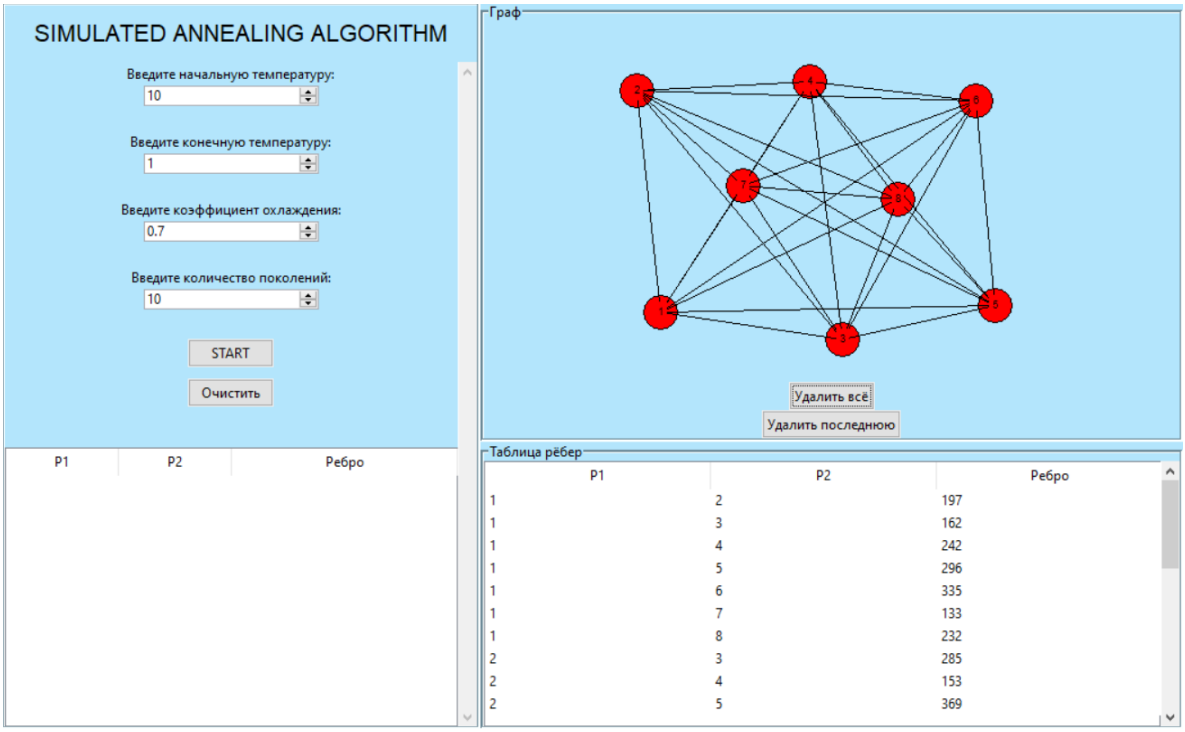

Рис. 2: Окно ввода.

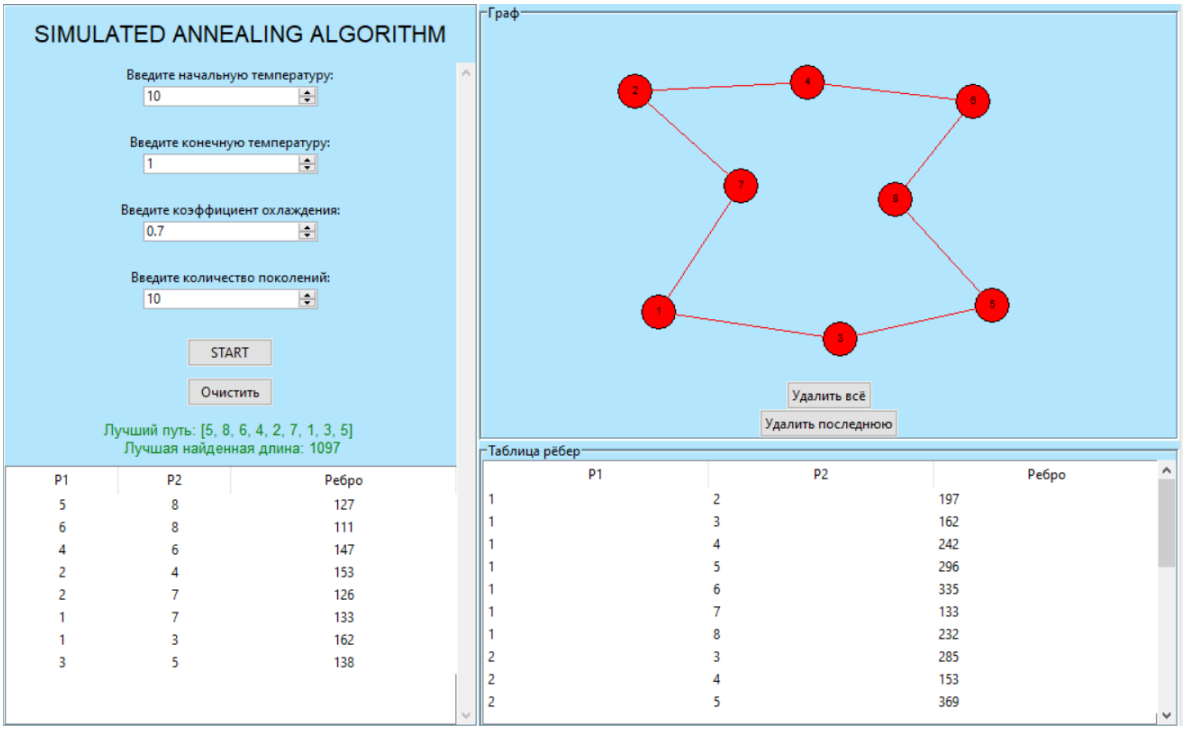

Рис. 3: Окно вывода.

Устройсво приложения:

В левой части окна можно вводить параметры влияющие на систему, такие как: **начальная и конечная температура, коэффициент охлаждения и кол-во поколений**.

Вверху справа графическое интерактивное окно, в котором можно добавлять точки одним щелчком мыши. Пути графа рисуются автоматически и добавляются в таблицу рёбер ниже. Весом ребра мо умолчанию является его длина в пикселях, которую можно легко изменить двойным нажатием мыши по нужной ячейке.

Нажимая на кнопку старт, производяться рассчёты, результат которых выводиться под кнопкой зелёным цветом, а таблица заполняется маршрутом в указанном алгоритмом порядке.

6 Оценка эффективности алгоритма

Оценка эффективности метода отжига в решении задачи коммивояжера показала, что он способен найти достаточно хорошие решения при правильном выборе начальных параметров и охлаждающего коэффициента. Так на личном опыте для графа состоящего из 20 вершин, метод отжига справился лучше, чем жадный алгоритм ближайшего соседа, однако не всегда удавалось добиться точности, близкой к полному перебору всех возможных вариантов.

Метод отжига является эффективным алгоритмом для задачи коммивояжера, особенно когда другие методы оптимизации не могут обеспечить достаточную точность результата. В то же время, для того чтобы получить наилучшие результаты, необходимо проводить тщательный анализ параметров алгоритма, таких как начальная температура, охлаждающий коэффициент и конечная температура.

Заключение

В данной работе был расмотрен метод отжига на поиск минимального по длине Гамельтона цикла. Были выявленны его преимущества в сравнение с методом ближайшего соседа. Была представленна его програмная визуализация и теоретическое растолкование.

Список литературы

[1] Лопатин А.С. Метод отжига, Санкт-Петербургский государственный университет.