МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ «МИСИС»

Институт информационных технологий и компьютерных наук Кафедра инженерной кибернетики

### КУРСОВАЯ РАБОТА

по дисциплине «Объектно-ориентированное программирование»

на тему “Решение задачи коммивояжера посредством алгоритма "Apple carving".”

Выполнил: студент 1-го курса,

гр. БПМ-22-4- Позоян Р.О.

Проверил: доцент, к.т.н. Полевой Д.В.

Москва, 2023

Оглавление

[КУРСОВАЯ РАБОТА 1](#_Toc148999177)

[Введение 3](#_Toc148999178)

[Описание задачи 3](#_Toc148999179)

[Пользовательское описание 4](#_Toc148999180)

[Пользовательская документация 4](#_Toc148999181)

[Инструкция по установке и запуску 5](#_Toc148999182)

[Тестирование 7](#_Toc148999183)

[Литература 8](#_Toc148999184)

# Введение

В данной курсовой работе рассматривается одна из самых известных задач комбинаторной оптимизации – задача Коммивояжер. Для работы с визуализацией хода работы библиотеки используется библиотека **CGAL**, которая позволяет сделать триангуляцию Делоне и пошагово показать работу кода.

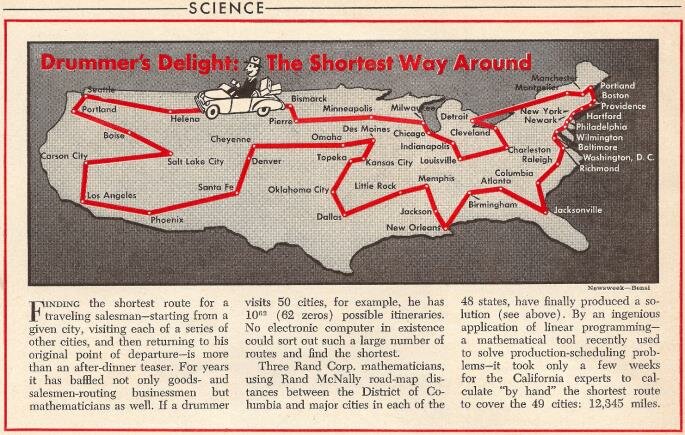
В общем виде нам дан набор городов и расстояние между ними(точек с координатами), которое нужно пройти между каждой парой из них, и тогда задача состоит в том, чтобы найти кратчайший маршрут для посещения каждого города и возвращения в исходную точку.

Реализация самого **Apple carving algorithm** и поэтапная визуализация через **CGAL** и **TeX Euklid** будут являться непосредственно частями библиотеки. Входные данные будут подаваться в приложении. (Координаты точек)

Задача коммивояжера применяется в логистике и навигаторах, так как она, собственно, и нацелена на вычисление наиболее рационального пути. А самым быстрым на данный момент является алгоритм **Apple carving**, его мы и реализуем в данной библиотеке.

# Описание задачи

Необходимо разработать библиотеку AppleCarving для реализации интерфейса между входными данными в виде чисел (координаты x и y) и наиболее быстрой обработки и преобразования данных в нужный нам формат, чтобы в последствии осуществить триангуляцию предварительно построенной кривой и при помощи алгоритма apple carving “вырезать” ненужные полигоны и получить наиболее рациональный маршрут из точки А в точку Б.



# Пользовательское описание

Для начала работы необходимо предоставить библиотеке текстовый файл с числами в виде координат с новой строки.

Далее, надо в самом коде прописать путь до самого файла с входными данными. Также можно воспользоваться тестовыми файлами с необходимым форматом данных, что предоставит пользователю пример работы проекта и правила оформления файла с координатами.

Сам файл необходимо предоставить в формате txt.

# Пользовательская документация

Список файлов

Объявления и описания членов классов находятся в файле:

-…\include\AppleCarving\AppleCarving.hpp

Класс Carving

Класс carving предназначен для реализации алгоритма Apple Carving.

#include <Apple\_Carving.hpp>

Открытые члены:

* Carving()

Конструктор класса

* ∼Carving()

Деструктор класса

* void MergeFaces(Cities& cities)
* void Generate\_path(Cities& cities, int begin\_trap, int end\_trap)

Открытые атрибуты:

* std::vector<Traping> traping

Вектор, содержащий все варианты путей

Методы:

* void MergeFaces(Cities& cities)

Метод для вырезания триангуляции

Аргумент:

|  |  |
| --- | --- |
| cities | Массив координат городов |

# void Generate\_path(Cities& cities, int begin\_trap, int end\_trap)

Метод для поиска оптимального пути

Аргумент:

|  |  |
| --- | --- |
| cities | Массив координат городов |
| begin\_trap |  |
| end\_trap |  |

Класс Cities

Открытые члены:

* Cities()
* ~Cities()
* void getCount(std::string path)
* void getCoordinate(std::string path)
* void getWeight()
* void InitializationPoints(std::string path, int argc, char\* argv[])
* void DrawBeforeCarving()
* void DrawAfterCarving()

Открытые атрибуты:

* int count

Количество городов

* std::string path

Путь к файлу

* std::vector <Coordinate> coordinate

Координаты точек триангуляции

* std::vector <std::vector <int>> weight

Расстояние между городами

* Triangulation TBeforeCarving

Триангуляция перед применением алгоритма Apple carving

* Triangulation TAfterCarving

Триангуляция после применения алгоритма Apple carving

* std::vector <Vertex\_handle> Vertex

Вершины триангуляции

Методы:

* void getCount(std::string path)

Установка количества городов

Аргумент:

|  |  |
| --- | --- |
| path | Путь до файла с входными данными(…\AppleCarving\project\test\data\dataX.txt) |

* void getCoordinate(std::string path)

Получение координат

Аргумент:

|  |  |
| --- | --- |
| path | Путь до файла с входными данными(…\AppleCarving\project\test\data\dataX.txt) |

* void getWeight()

Установка расстояния между городами

Аргумент:

|  |  |
| --- | --- |
| x, y | Города, между которыми необходимо найти расстояние |

* void InitializationPoints(std::string path, int argc, char\* argv[])

Триангуляция между городами

Аргумент:

|  |  |
| --- | --- |
| path | Путь до файла с входными данными(…\AppleCarving\project\test\data\dataX.txt) |
| argc | Индекс начала считывания файла |
| argv | Начало считывания файла |

Шаблон класса My\_vertex\_base

# Инструкция по установке и запуску

Для работы библиотеки Apple\_Carving и генерации документации необходима система генерации Doxygen и библиотека CGAL.

Пошаговая иструкция:

1. Скачайте исходные код библиотеки с репозитория по адресу: <https://github.com/RafaelPozoyan/oop_pozoyan_r_o/tree/main/pozoyan_r_o>
2. Откройте проект, и в сборочном файле **.build.bat** впишите адрес директории, содержащей vcpkg.cmake. (…\vcpkg\scripts\buildsystems\vcpkg.cmake)
3. Выполните команду .build.bat через командную строку для сборки проекта и создания папки build\_my\_project .
4. Запустите файл **libraries.sln** (…lib\_apple\build\_my\_project\libraries.sln), где в свойствах проекта надо выбрать запускаемым файлом **apple**.
5. После запуска проекта у вас появится визуализация работы алгоритма, также можно будет сгенерировать latex-файл или html-документацию через Doxywizard.

# Тестирование

При получении выходных данных и выполнении кодом некоторых условий, можно сказать, что библиотека выполнила все поставленные перед ней задачи.

Для тестирования используются текстовые файлы, которые находятся по пути **…\lib\_apple\AppleCarving\project\test\data**. Данные файлы ссодержат координаты городов, между которыми передвигается продавец. Необходимо вписать адрес тестового приложения в main.cpp и запустить его.

# Литература

* Задача Коммивояжера - <https://en.wikipedia.org/wiki/Travelling_salesman_problem>
* Описание и объяснение алгоритма ***apple carving*** - <https://www.researchgate.net/publication/340401264_Apple_Carving_Algorithm_to_Approximate_Traveling_Salesman_Problem_from_Compact_Triangulation_of_Planar_Point_Sets>
* Необходимая информация для работы с библиотекой CGAL -https://doc.cgal.org/latest/Triangulation\_2/index.html