ГУАП

КАФЕДРА № 51

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| канд. техн. наук |  |  |  | Е. М. Линский |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О КУРСОВОЙ РАБОТЕ |
| АЛГОРИТМ ХАФФМАНА |
| по курсу: ОСНОВЫ ПРОГРАММИРОВАНИЯ |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 5013 |  |  |  | А. В. Гончаров |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2021

Оглавление

[1 Постановка задачи 3](#_Toc89705300)

[Задача коммивояжёра 3](#_Toc89705301)

[2 Жадный Алгоритм 3](#_Toc89705302)

[Описание алгоритма 3](#_Toc89705303)

[3 Метод ветвей и границ 6](#_Toc89705304)

[Описание алгоритма 6](#_Toc89705305)

[Структуры данных, используемые в программе 9](#_Toc89705306)

[Псевдокод 10](#_Toc89705307)

[4 Инструкция пользователя 11](#_Toc89705308)

[Запуск программы 11](#_Toc89705309)

[Формат входного файла 12](#_Toc89705310)

[Формат выходного файла 12](#_Toc89705311)

[5 Тестовые примеры 12](#_Toc89705312)

[Тест 1 12](#_Toc89705313)

[Тест 2 13](#_Toc89705314)

[Тест 3 14](#_Toc89705315)

[Список литературы 15](#_Toc89705316)

# Постановка задачи

Задачей данной курсовой работы является разработка программы-архиватора английских букв, использующей для сжатия алгоритм Хаффмана.

# Алгоритм Хаффмана

## Описание алгоритма

Алгоритм Хаффмана - жадный алгоритм оптимального префиксного кодирования алфавита с минимальной избыточностью. Был разработан в 1952 году аспирантом Массачусетского технологического института Дэвидом Хаффманом при написании им курсовой работы. В настоящее время используется во многих программах сжатия данных.

Пример:

Найти кратчайший цикл в графе (Рисунок 1) матрица смежности для данного графа (Таблица 1).

Таблица - Исходная матрица смежности.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 |
| 1 | 999 | 2 | 35 | 5 |
| 2 | 11 | 999 | 20 | 25 |
| 3 | 18 | 19 | 999 | 3 |
| 4 | 2 | 20 | 34 | 999 |

Алгоритм начинается с приведения матрицы смежности (Таблица 2 Таблица 4)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Таблица - Приведение по строкам.   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | |  | 1 | 2 | 3 | 4 |  | | 1 | 999 | 2 | 35 | 5 | 2 | | 2 | 11 | 999 | 20 | 25 | 11 | | 3 | 18 | 19 | 999 | 3 | 3 | | 4 | 2 | 20 | 34 | 999 | 2 | | Таблица - Приведение по столбцам.   |  |  |  |  |  | | --- | --- | --- | --- | --- | |  | 1 | 2 | 3 | 4 | | 1 | 999 | 0 | 33 | 3 | | 2 | 0 | 999 | 9 | 14 | | 3 | 15 | 16 | 999 | 0 | | 4 | 0 | 18 | 32 | 999 | |  | 0 | 0 | 9 | 0 | |
| Таблица - Приведенная матрица.   |  |  |  |  |  | | --- | --- | --- | --- | --- | |  | 1 | 2 | 3 | 4 | | 1 | 999 | 0 | 24 | 3 | | 2 | 0 | 999 | 0 | 14 | | 3 | 15 | 16 | 999 | 0 | | 4 | 0 | 18 | 23 | 999 | | |

Сумма минимальных значений, использованных при приведении, даст нам нижнюю границу для данного графа 2 + 11 + 3 + 2 + 0 + 0 + 9 + 0 = 27. Так как верхняя и нижняя граница, полученная ЖА, равны можно сказать что цикл полученный в ЖА является решением поставленной задачи.

В полученной приведенной матрице (Таблица 4) у нас есть несколько нулевых путей: (1;2), (2;1), (2;3), (3;4), (4;1). Для каждого из них найдем сумму минимальных элементов строки и столбца. При поиске минимального значения мы исключаем элемент нулевого узла. Пример для узла (1;2) (Таблица 5):

Таблица - Поиск минимальных элементов в 1 строке и 2 столбце, исключая элемент (1;2).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 |  |
| 1 | 999 | 0 | 24 | 3 | 3 |
| 2 | 0 | 999 | 0 | 14 |  |
| 3 | 15 | 16 | 999 | 0 |  |
| 4 | 0 | 18 | 23 | 999 |  |
|  |  | 16 |  |  |  |

(1;2): 16 + 3 = 19, (2;1): 0 + 0 = 0 , (2;3): 23 + 0 = 23, (3;4): 15 + 3 = 18, (4;1): 18 + 0 = 18 .

Из полученных значений выбираем наибольшее (2;3): 23. Далее происходит выбор включать данное ребро в цикл или наоборот исключить. Для этого сравниваются результаты приведения двух матриц (Таблица 6,Таблица 7).

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Таблица - Путь исключающий (2;3).   |  |  |  |  |  | | --- | --- | --- | --- | --- | |  | 1 | 2 | 3 | 4 | | 1 | 999 | 0 | 24 | 3 | | 2 | 0 | 999 | 999 | 14 | | 3 | 15 | 16 | 999 | 0 | | 4 | 0 | 18 | 23 | 999 | | Таблица - Путь включающий (2;3).   |  |  |  |  |  | | --- | --- | --- | --- | --- | |  | 1 | 2 | 3 | 4 | | 1 | 999 | 0 | 999 | 3 | | 2 | 999 | 999 | 999 | 999 | | 3 | 15 | 999 | 999 | 0 | | 4 | 0 | 18 | 999 | 999 | |

Для выбранного ребра сумма минимальных значений, использованных при приведении матрицы, включающей ребро меньше, чем исключающей (0 против 33) поэтому далее в алгоритме, мы будем использовать её, а полученное значение прибавляем к нижней границе. Далее мы вновь ищем нулевые пути и суммы для них (1;2): 18 + 3 = 21, (3;4): 15 + 3 = 18, (4;1): 15 + 18 = 33. Следующим путем для рассмотрения возьмем (4;1).

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Таблица - Путь исключающий (4;1).   |  |  |  |  |  | | --- | --- | --- | --- | --- | |  | 1 | 2 | 3 | 4 | | 1 | 999 | 0 | 999 | 3 | | 2 | 999 | 999 | 999 | 999 | | 3 | 15 | 999 | 999 | 0 | | 4 | 999 | 18 | 999 | 999 | | Таблица - Путь включающий (4;1).   |  |  |  |  |  | | --- | --- | --- | --- | --- | |  | 1 | 2 | 3 | 4 | | 1 | 999 | 0 | 999 | 999 | | 2 | 999 | 999 | 999 | 999 | | 3 | 999 | 999 | 999 | 0 | | 4 | 999 | 999 | 999 | 999 | |

Для выбранного ребра сумма минимальных значений, использованных при приведении матрицы, включающей ребро меньше, чем исключающей (0 против 33) поэтому далее в алгоритме, мы будем использовать её, а полученное значение прибавляем к нижней границе. Далее мы вновь ищем нулевые пути и суммы для них (1;2): 0 + 0 = 3, (3;4): 0 + 0 = 0. Следующим путем для рассмотрения возьмем (1;2). Используя описанные выше шаги приводим матрицу к виду (Таблица 10).

Таблица - Конечная матрица.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 |
| 1 | 999 | 999 | 999 | 999 |
| 2 | 999 | 999 | 999 | 999 |
| 3 | 999 | 999 | 999 | 999 |
| 4 | 999 | 999 | 999 | 999 |

Выписав включенные пути получаем цикл 1 – 2 – 3 – 4 – 1.

## Структуры данных, используемые в программе

В данном алгоритме используются структуры данных:

* class Graph – класс для хранения и обработки графа
* std::vector< int> cycle – структура данных для хранения цикла жадного алгоритма
* std::vector<std::vector<int>> matrix – структура данных для хранения матрицы смежности
* std::vector<std::vector<int>> chosen\_way – структура данных для хранения цикла полученного методом ветвей и границ

## Псевдокод

Жадный алгоритм:

1. Создание массива visited для учета пройдённых узлов;
2. for(j = 0; j < размер матрицы; j++){
3. Поиск минимального не посещенного пути из точки min\_point;
4. Суммирование полученного пути с предыдущими total\_cost\_gready += matrix[point][min\_point];
5. Переход по найденному пути point = min\_point;
6. Отметка узла как пройденного в массиве visited;
7. Внесение узла в cycle;
8. }

Метод ветвей и границ:

1. Приводим матрицу смежности и запоминаем нижнюю границу;
2. total\_coast\_BaB = matrix\_reduction(tmp\_matrix);
3. While (! (chousen\_way.size() == size)) {size – число узлов, chousen\_way – вектор содержащий выбранные пути.
4. Поиск проверяемого пути {
5. Находим нулевой элемент матрицы с наибольшей суммой минимальных элементов в строки и столбца, при поиске минимальных элементов исключаем проверяемый нулевой.
6. }
7. Int wout\_way = without\_way(tmp\_matrix);
8. Int w\_way = with\_way(tmp\_matrix);
9. if (wout\_way <= w\_way) {
10. total\_cost\_BaB += wout\_way;
11. }
12. else {
13. total\_cost\_BaB += w\_way;
14. }
15. }

Функция matrix\_reduction(vector<vector<int>> &tmp\_m):

1. for (int i = 0; i < size; i++) {
2. int tmp\_bott = 0;
3. if (Строка не использована) {

Находим минимальный элемент строки, вычитаем его из элементов строки, и складываем его с нижней границей tmp\_bott += min;

1. }
2. }
3. for (int i = 0; i < size; i++) {
4. int tmp\_bott = 0;
5. if (Столбец не использован) {

Находим минимальный элемент столбца, вычитаем его из элементов столбца, и складываем его с нижней границей tmp\_bott += min;

1. }
2. }
3. return tmp\_bott;

# Инструкция пользователя

## Запуск программы

Запуск программы происходит из командной строки:

*Адрес\_к\_исполняемому\_файлу\Coursework*.exe *Адрес\_к\_входному\_файлу\название\_входного\_файла.txt Адрес\_к\_выходному\_файлу\_1\название\_выходного\_файла\_1.png Адрес\_к\_выходному\_файлу\_2\название\_выходного\_файла\_2.png*

Данные для входного и выходных фалов вводятся через пробел.

## Формат входного файла

Содержимое файла:

* В первой и второй строчке – начальная вершина и количество узлов в графе соответственно.
* В последующих строчках: вводится матрица смежности, где не существующие пути обозначаются числом 999
* Примеры содержимого входного файла в Таблица 11, Таблица 12, Таблица 13.

## Формат выходного файла

На выходе программы получаем вывод в консоль результатов работы алгоритмов и два файла, с заданными названиями, которые являются изображениями графов с выделенными конечными циклами, полученными жадным алгоритмом и методом ветвей и границ соответственно.

# Тестовые примеры

## Тест 1

Таблица . Тест 1

|  |  |
| --- | --- |
| До выполнения | После выполнения |
| “input.txt”:  start point = 1  Graph[5]:  999 | 2 | 35 | 5 | 20  11 | 999 | 20 | 25 | 2  18 | 19 | 999 | 3 | 20  2 | 20 | 34 | 999 | 2  2 | 18 | 3 | 37 | 999 | “OutputGready.png”: |
| “OutputBaB.png”: |
| Cmd:  Gready algorithm resalt: 12  BaB algorithm resalt: 12 |

## Тест 2

Таблица . Тест 2

|  |  |
| --- | --- |
| До выполнения | После выполнения |
| start point = 1  Graph[5]:  999 | 10 | 25 | 25 | 10  1 | 999 | 10 | 15 | 2  8 | 9 | 999 | 20 | 10  14 | 10 | 24 | 999 | 15  10 | 8 | 25 | 27 | 999 | “OutputGready.png”: |
| “OutputBaB.png”: |
| Cmd:  Gready algorithm resalt: 71  BaB algorithm resalt: 65 |

## Тест 3

Таблица . Тест 3

|  |  |
| --- | --- |
| До выполнения | После выполнения |
| start point = 3  Graph[7]:  999|90 |80 |40 |100|40 |80  60 |999|40 |50 |70 |40 |60  50 |30 |999|60 |20 |20 |40  10 |70 |20 |999|50 |10 |60  20 |40 |50 |20 |999|20 |10  60 |30 |40 |90 |60 |999|30  50 |30 |20 |60 |20 |20 |999 | “OutputGready.png”: |
| “OutputBaB.png”: |
| Cmd:  Gready algorithm resalt: 220  BaB algorithm resalt: 180 |

# Список литературы

1. Э. Рейнгольд , Ю. Нивергельт, Н. Део, Комбинаторные алгоритмы, Мир, 1980 г.